

egix@karmainsecurity ~ \$

~/research ~/blog ~/about

# MetInfo CMS <= 8.1 (weixinreply.class.php) PHP Code Injection Vulnerability

More Link:

[www.metinfo.cn](http://www.metinfo.cn)

Affected Versions:

8.9, 8.0, and 8.1.

Vulnerability Description:

Arbitrary code is located into the `/app/system/weixin/include/class/weixinreply.class.php` script.

Specifically, within the `weixinreply::wxAdminLogin()` method:

```
public function wxAdminLogin($data = array(), $code = '')
{
    global $_M;
    $weixinapi = load::mod_class('weixin/weixinapi', 'new');
    $login_code = cache::get("weixin/" . $code);
    if ($login_code) {
        cache::put("weixin/" . $login_code, $data['FromUserName']);
    }
    return;
}
```

The `$code` parameter is passed through the `EventKey` and `FromUserName` XML tags from the HTTP request body when dispatching `weixin` API. The `$code` parameter is sanitized before being used in a call to the `cache::get()` and `cache::put()` methods respectively.

By abusing the `$code` parameter, the `cache::get()` method can be used to include arbitrary files. Specifically, the `$code` parameter may include Path Traversal sequences, making the `cache::get()` method include arbitrary files. Subsequently, the `cache::put()` method will write the `FromUserName` parameter into the `/cache/weixin/Array.php` file, embedding it within double quotes:

```
public static function put($file, $data, $type = 'php')
{
    global $_M;

    load::sys_func('file');
    $save = PATH_CACHE . $file . '.' . $type;
    makefile($save);
    $data = str_replace(array("\n", "\\n"), array("\\n", "\\n"), $data);
    if (!is_array($data)) {
        file_put_contents($save, "<?php\nundefined('IN_MET') or exit('No permission');\n\n$cache=\"{$data}\"");
    } else {
        $info = var_export($data, true);
        $info = "<?php\nundefined('IN_MET') or exit('No permission');\n\n$cache = {$info};\n?>";
        file_put_contents($save, $info);
    }
}
```

This vulnerability can be exploited by remote, unauthenticated attackers to inject and execute arbitrary PHP code by abusing PHP's complex caching mechanism, resulting in an unauthenticated Remote Code Execution (RCE).

When MetInfo is running on non-Windows servers, successful exploitation of this vulnerability requires the `/cache/weixin/` directory to be created when installing and configuring the official WeChat plugin.

### Proof of Concept:

<https://karmainsecurity.com/pocs/CVE-2026-29014.php>

### Conclusion:

A patch or solution is currently available.

### Disclosure Timeline:

- [6] – Vendor contacted through several `@metinfo.cn` and `@mituo.cn` email addresses, no response
- [6] – Tried to reach out to the vendor again, no response
- [6] – Tried to reach out to the vendor once again, no response
- [6] – Tried to reach out to the vendor through [Weibo](#), no response
- [6] – CVE identifier requested
- [6] – CVE identifier assigned
- [6] – Public disclosure

### Reference:

[CVE-2026-29014](#) has been assigned to this vulnerability.

### Credits:

Vulnerability discovered by Egidio Romano.

Karma(In)Security © 2026