xibosignage / **xibo-cms** Public

Code | Pull requests 17 | Actions | Security and quality 18 | Insights

Commit ed213cb

dasgarner committed on Mar 5

```
DataSet: improve sanitization

 relates to xibosignageltd/xibo-private#1244
```

hotfix/borrelly

1 parent f646204 commit ed213cb

3 files changed +102 -12

Top

Filter files...

- lib
  - Controller
    - DataSetData.php
  - Entity
    - DataSet.php
  - Helper
    - Sql.php

Search within code

lib/Controller/DataSetData.php

```
@@ -147,7 +147,7 @@ public function grid(Request $request, Response
$response, $id)
147  147          $filter = trim($filter, 'AND');
148  148
149  149          // Work out the limits
```

```
150     -         $filter = $this->gridRenderFilter(['filter' => $request->getParam('filter', $filter)], $sanitizedParams);
        150 +     $filter = $this->gridRenderFilter(['filter' => $filter], $sanitizedParams);
151     151
152     152           try {
153     153               $data = $dataSet->getData(
```

lib/Entity/DataSet.php

```
        @@ -31,6 +31,7 @@
31      31    use Xibo\Factory\DataSetFactory;
32      32    use Xibo\Factory\PermissionFactory;
33      33    use Xibo\Helper\SanitizerService;
        34  + use Xibo\Helper\Sql;
34      35    use Xibo\Service\ConfigServiceInterface;
35      36    use Xibo\Service\DisplayNotifyServiceInterface;
36      37    use Xibo\Service\LogServiceInterface;
        @@ -265,9 +266,6 @@ class DataSet implements \JsonSerializable
265     266
266     267       private $countLast = 0;
267     268
268     -     /** @var array Blacklist for SQL */
269     -     private $blackList = array(';', 'INSERT', 'UPDATE', 'SELECT', 'DELETE', 'TRUNCATE', 'TABLE', 'FROM', 'WHERE');
270     -
271     269       /** @var  \Xibo\Helper\SanitizerService */
272     270       private $sanitizerService;
273     271
        @@ -442,7 +440,7 @@ public function getUniqueColumnValues($columns)
442     440                   if ($column->heading == $heading) {
443     441                       // Formula column?
444     442                       if ($column->dataSetColumnTypeId == 2) {
445     -                         $select .= str_replace($this->blackList, '', htmlspecialchars_decode($column->formula, ENT_QUOTES)) . ' AS `' . $column->heading . '`,';
        443 +                     $select .= Sql::cleanup(htmlspecialchars_decode($column->formula, ENT_QUOTES)) . ' AS `' .
```

```
                $column->heading . '`,';
446 444                     }
447 445                     else {
448 446                         $select .= '`' . $column->heading . '`,';
    @@ -527,12 +525,7 @@ public function getData($filterBy = [], $options = [],
    $extraParams = [])
527 525                 }
528 526
529 527                 $count = 0;
530     -               $formula = str_ireplace(
531     -                   $this->blackList,
532     -                   '',
533     -                   htmlspecialchars_decode($column->formula, ENT_QUOTES),
534     -                   $count
535     -               );
    528 +               $formula = Sql::cleanup(htmlspecialchars_decode($column-
    >formula, ENT_QUOTES), $count);
536 529
537 530                 if ($count > 0) {
538 531                     $this->getLog()->error(
    @@ -559,7 +552,7 @@ public function getData($filterBy = [], $options = [],
    $extraParams = [])
559 552             if ($filter != '') {
560 553                 // Support display filtering.
561 554                 $filter = str_replace('[DisplayId]', $displayId, $filter);
562     -               $filter = str_ireplace($this->blackList, '', $filter);
    555 +               $filter = Sql::cleanup($filter);
563 556
564 557                 $body .= ' AND ' . $filter;
565 558             }
```

lib/Helper/Sql.php

```
    @@ -0,0 +1,97 @@
1   + <?php
2   +
3   + namespace Xibo\Helper;
4   +
5   + /**
6   +  * SQL definitions
```

```
7  +  */
8  + class Sql
9  + {
10 +     const DISALLOWED_KEYWORDS = [
11 +         ';', '@@', // Reduced symbols, handling comments via regex now
12 +         'INSERT', 'UPDATE', 'SELECT', 'FROM', 'WHERE', 'DELETE', 'TRUNCATE',
13 +         'TABLE', 'ALTER', 'GRANT', 'REVOKE', 'CREATE', 'DROP', 'UNION',
14 +         'HAVING', 'GROUP', 'INTO', 'OUTFILE', 'DUMPFILE', 'PROCEDURE',
15 +         'SLEEP', 'BENCHMARK', 'INFORMATION_SCHEMA', 'LOAD_FILE', 'LOCK',
16 +         'EXECUTE', 'PREPARE', 'DEALLOCATE', 'SHOW', 'DESCRIBE', 'EXPLAIN',
17 +         'CALL', 'HANDLER', 'RENAME', 'SHUTDOWN', 'SET', 'USE', 'FLUSH',
18 +         'KILL', 'OPTIMIZE', 'REPAIR', 'ANALYZE', 'CHECK', 'CHECKSUM',
19 +         'GET_LOCK', 'RELEASE_LOCK', 'IS_FREE_LOCK', 'IS_USED_LOCK',
20 +         'MASTER_POS_WAIT', 'PASSWORD', 'USER', 'SYSTEM_USER', 'SESSION_USER',
21 +         'CURRENT_USER', 'DATABASE', 'SCHEMA', 'VERSION', 'CONNECTION_ID',
22 +         'LAST_INSERT_ID', 'ROW_COUNT', 'FOUND_ROWS', 'LOAD_XML', 'NAME_CONST',
23 +         'DO', 'EXTRACTVALUE', 'UPDATEXML', 'XMLTYPE', 'DBMS_PIPE', 'PG_SLEEP',
24 +         // Added String Builders & Encoders
25 +         // we have specific use cases for 'CONCAT', so we keept that
26 +         'CONCAT_WS', 'CHAR', 'UNHEX', 'HEX', 'ASCII', 'BIN', 'ORD', 'BASE64'
27 +     ];
28 +
29 +     /**
30 +      * Cleanup SQL (Maximum Paranoia for Legacy Code)
31 +      * @param string $sql the SQL to clean
32 +      * @param int $total the total number of replacements
33 +      * @return string
34 +      */
35 +     public static function cleanup(string $sql, int &$total = 0): string
36 +     {
37 +         // 1. EXTRACT AND PROTECT STRING LITERALS (Preserve user data)
38 +         $strings = [];
39 +         $placeholderPrefix = '__SQL_STR_';
40 +         $stringPattern = '/(\'(?:\\\\.|[^\'\\\\])*\'|"(?:\\\\.|[^"\\\\])*")/';
41 +
42 +         $sqlCleaned = preg_replace_callback($stringPattern, function ($matches)
   use (&$strings, $placeholderPrefix) {
43 +             $id = count($strings);
44 +             $strings[] = $matches[0];
45 +             return $placeholderPrefix . $id . '__';
```

```
46 +         }, $sql);
47 +
48 +         // 2. STRIP COMMENTS & ENCODINGS (Before keyword checks)
49 +         $preCleanupCount = 0;
50 +         $preCleanupPatterns = [
51 +             '/(?:\/\*.*?\*\/|--[ \t].*?(?:\n|$)|#[^\n]*?(?:\n|$))/s', // Standard & Executable Comments
52 +             '/\b0x[0-9a-fA-F]+\b/',                                    // Hex literals (e.g., 0x7e)
53 +             '/\bb\'[01]+\'/i'                                          // Binary literals
54 +         ];
55 +         $sqlCleaned = preg_replace($preCleanupPatterns, '', $sqlCleaned, -1, $preCleanupCount);
56 +         $total += $preCleanupCount;
57 +
58 +         // 3. PREPARE KEYWORD PATTERNS
59 +         $wordKeywords = [];
60 +         $symbolKeywords = [];
61 +
62 +         foreach (self::DISALLOWED_KEYWORDS as $keyword) {
63 +             if (ctype_alnum(str_replace('_', '', $keyword))) {
64 +                 $wordKeywords[] = preg_quote($keyword, '/');
65 +             } else {
66 +                 $symbolKeywords[] = $keyword;
67 +             }
68 +         }
69 +
70 +         $wordPattern = empty($wordKeywords) ? null : '/\b(' . implode('|', $wordKeywords) . ')\b/i';
71 +
72 +         // 4. RECURSIVE CLEANUP
73 +         $count = 0;
74 +         do {
75 +             $symbolCount = 0;
76 +             $wordCount = 0;
77 +
78 +             $sqlCleaned = str_ireplace($symbolKeywords, '', $sqlCleaned, $symbolCount);
79 +
```

```
80  +             if ($wordPattern) {
81  +                 $sqlCleaned = preg_replace($wordPattern, '', $sqlCleaned, -1,
    $wordCount);
82  +             }
83  +
84  +             $count = $symbolCount + $wordCount;
85  +             $total += $count;
86  +         } while ($count > 0);
87  +
88  +         // 5. RESTORE STRING LITERALS
89  +         if (!empty($strings)) {
90  +             foreach ($strings as $id => $originalString) {
91  +                 $sqlCleaned = str_replace($placeholderPrefix . $id . '__',
    $originalString, $sqlCleaned);
92  +             }
93  +         }
94  +
95  +         return trim($sqlCleaned);
96  +     }
97  + }
```

Comments 0

Please sign in to comment.