xibosignage / **xibo-cms** Public

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

Commit `87e0a26`

dasgarner committed on Mar 5 Verified

```
DataSet: improve sanitization

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

`hotfix/wolf`

1 parent `2caec88` commit `87e0a26`

3 files changed +119 -7

Top

Filter files...

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

Search within code

lib/Controller/DataSetData.php

```
@@ -123,7 +123,7 @@ public function grid($dataSetId)
123  123          $filter = trim($filter, 'AND');
124  124
125  125          // Work out the limits
```

```
126      -          $filter = $this->gridRenderFilter(['filter' => $this->getSanitizer()->getParam('filter', $filter)]);
         126 +      $filter = $this->gridRenderFilter(['filter' => $filter]);
127      127
128      128        try {
129      129            $data = $dataSet->getData([
```

lib/Entity/DataSet.php

```
         @@ -19,6 +19,7 @@
19       19     use Xibo\Factory\DataSetFactory;
20       20     use Xibo\Factory\DisplayFactory;
21       21     use Xibo\Factory\PermissionFactory;
         22  +  use Xibo\Helper\Sql;
22       23     use Xibo\Service\ConfigServiceInterface;
23       24     use Xibo\Service\DateServiceInterface;
24       25     use Xibo\Service\LogServiceInterface;
         @@ -207,9 +208,6 @@ class DataSet implements \JsonSerializable
207      208
208      209        private $countLast = 0;
209      210
210      -      /** @var array Blacklist for SQL */
211      -      private $blackList = array(';', 'INSERT', 'UPDATE', 'SELECT', 'DELETE', 'TRUNCATE', 'TABLE', 'FROM', 'WHERE');
212      -
213      211        /** @var  SanitizerServiceInterface */
214      212        private $sanitizer;
215      213
         @@ -364,7 +362,7 @@ public function getUniqueColumnValues($columns)
364      362                    if ($column->heading == $heading) {
365      363                        // Formula column?
366      364                        if ($column->dataSetColumnTypeId == 2) {
367      -                          $select .= str_replace($this->blackList, '', htmlspecialchars_decode($column->formula, ENT_QUOTES)) . ' AS `' . $column->heading . '`,';
         365  +                  $select .= Sql::cleanup(htmlspecialchars_decode($column->formula, ENT_QUOTES)) . ' AS `' . $column->heading . '`,';
```

```
368  366                        }
369  367                        else {
370  368                            $select .= '`' . $column->heading . '`,';
     ↕  @@ -445,7 +443,7 @@ public function getData($filterBy = [], $options = [])
445  443                        continue;
446  444                    }
447  445
448       -                $formula = str_ireplace($this->blackList, '', htmlspecialchars_decode($column->formula, ENT_QUOTES));
          446 +                $formula = Sql::cleanup(htmlspecialchars_decode($column->formula, ENT_QUOTES));
449  447                    $formula = str_replace('[DisplayId]', $displayId, $formula);
450  448
451  449                    $heading = str_replace('[DisplayGeoLocation]', $displayGeoLocation, $formula) . ' AS `' . $column->heading . '`';
     ↕  @@ -463,7 +461,7 @@ public function getData($filterBy = [], $options = [])
463  461            if ($filter != '') {
464  462                // Support display filtering.
465  463                $filter = str_ireplace('[DisplayId]', $displayId, $filter);
466       -                $filter = str_replace($this->blackList, '', $filter);
          464 +                $filter = Sql::cleanup($filter);
467  465
468  466                $body .= ' AND ' . $filter;
469  467            }
     ↓
```

lib/Helper/Sql.php

```
     ⋯  @@ -0,0 +1,114 @@
     1  + <?php
     2  + /*
     3  +  * Copyright (C) 2026 Xibo Signage Ltd
     4  +  *
     5  +  * Xibo - Digital Signage - http://www.xibo.org.uk
     6  +  *
     7  +  * This file is part of Xibo.
     8  +  *
     9  +  * Xibo is free software: you can redistribute it and/or modify
    10  +  * it under the terms of the GNU Affero General Public License as published by
    11  +  * the Free Software Foundation, either version 3 of the License, or
```

```
12  +  * any later version.
13  +  *
14  +  * Xibo is distributed in the hope that it will be useful,
15  +  * but WITHOUT ANY WARRANTY; without even the implied warranty of
16  +  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
17  +  * GNU Affero General Public License for more details.
18  +  *
19  +  * You should have received a copy of the GNU Affero General Public License
20  +  * along with Xibo.  If not, see <http://www.gnu.org/licenses/>.
21  +  */
22  +
23  + namespace Xibo\Helper;
24  +
25  + class Sql
26  + {
27  +     const DISALLOWED_KEYWORDS = [
28  +         ';', '@@', // Reduced symbols, handling comments via regex now
29  +         'INSERT', 'UPDATE', 'SELECT', 'FROM', 'WHERE', 'DELETE', 'TRUNCATE',
30  +         'TABLE', 'ALTER', 'GRANT', 'REVOKE', 'CREATE', 'DROP', 'UNION',
31  +         'HAVING', 'GROUP', 'INTO', 'OUTFILE', 'DUMPFILE', 'PROCEDURE',
32  +         'SLEEP', 'BENCHMARK', 'INFORMATION_SCHEMA', 'LOAD_FILE', 'LOCK',
33  +         'EXECUTE', 'PREPARE', 'DEALLOCATE', 'SHOW', 'DESCRIBE', 'EXPLAIN',
34  +         'CALL', 'HANDLER', 'RENAME', 'SHUTDOWN', 'SET', 'USE', 'FLUSH',
35  +         'KILL', 'OPTIMIZE', 'REPAIR', 'ANALYZE', 'CHECK', 'CHECKSUM',
36  +         'GET_LOCK', 'RELEASE_LOCK', 'IS_FREE_LOCK', 'IS_USED_LOCK',
37  +         'MASTER_POS_WAIT', 'PASSWORD', 'USER', 'SYSTEM_USER', 'SESSION_USER',
38  +         'CURRENT_USER', 'DATABASE', 'SCHEMA', 'VERSION', 'CONNECTION_ID',
39  +         'LAST_INSERT_ID', 'ROW_COUNT', 'FOUND_ROWS', 'LOAD_XML', 'NAME_CONST',
40  +         'DO', 'EXTRACTVALUE', 'UPDATEXML', 'XMLTYPE', 'DBMS_PIPE', 'PG_SLEEP',
41  +         // Added String Builders & Encoders
42  +         // we have specific use cases for 'CONCAT', so we keept that
43  +         'CONCAT_WS', 'CHAR', 'UNHEX', 'HEX', 'ASCII', 'BIN', 'ORD', 'BASE64'
44  +     ];
45  +
46  +     /**
47  +      * Cleanup SQL (Maximum Paranoia for Legacy Code)
48  +      * @param string $sql the SQL to clean
49  +      * @param int $total the total number of replacements
50  +      * @return string
51  +      */
```

```
52  +     public static function cleanup(string $sql, int &$total = 0): string
53  +     {
54  +         // 1. EXTRACT AND PROTECT STRING LITERALS (Preserve user data)
55  +         $strings = [];
56  +         $placeholderPrefix = '__SQL_STR_';
57  +         $stringPattern = '/(\'(?:\\\\.|[^\'\\\\])*\'|"(?:\\\\.|[^"\\\\])*")/';
58  +
59  +         $sqlCleaned = preg_replace_callback($stringPattern, function ($matches)
      use (&$strings, $placeholderPrefix) {
60  +             $id = count($strings);
61  +             $strings[] = $matches[0];
62  +             return $placeholderPrefix . $id . '__';
63  +         }, $sql);
64  +
65  +         // 2. STRIP COMMENTS & ENCODINGS (Before keyword checks)
66  +         $preCleanupCount = 0;
67  +         $preCleanupPatterns = [
68  +             '/(?:\/\*.*?\*\/|--[ \t].*?(?:\n|$)|#[^\n]*?(?:\n|$))/s', //
      Standard & Executable Comments
69  +             '/\b0x[0-9a-fA-F]+\b/',                                    // Hex
      literals (e.g., 0x7e)
70  +             '/\bb\'[01]+\'/i'                                          // Binary
      literals
71  +         ];
72  +         $sqlCleaned = preg_replace($preCleanupPatterns, '', $sqlCleaned, -1,
      $preCleanupCount);
73  +         $total += $preCleanupCount;
74  +
75  +         // 3. PREPARE KEYWORD PATTERNS
76  +         $wordKeywords = [];
77  +         $symbolKeywords = [];
78  +
79  +         foreach (self::DISALLOWED_KEYWORDS as $keyword) {
80  +             if (ctype_alnum(str_replace('_', '', $keyword))) {
81  +                 $wordKeywords[] = preg_quote($keyword, '/');
82  +             } else {
83  +                 $symbolKeywords[] = $keyword;
84  +             }
85  +         }
86  +
```

```
87  +         $wordPattern = empty($wordKeywords) ? null : '/\b(' . implode('|', $wordKeywords) . ')\b/i';
88  +
89  +         // 4. RECURSIVE CLEANUP
90  +         $count = 0;
91  +         do {
92  +             $symbolCount = 0;
93  +             $wordCount = 0;
94  +
95  +             $sqlCleaned = str_ireplace($symbolKeywords, '', $sqlCleaned, $symbolCount);
96  +
97  +             if ($wordPattern) {
98  +                 $sqlCleaned = preg_replace($wordPattern, '', $sqlCleaned, -1, $wordCount);
99  +             }
100 +
101 +             $count = $symbolCount + $wordCount;
102 +             $total += $count;
103 +         } while ($count > 0);
104 +
105 +         // 5. RESTORE STRING LITERALS
106 +         if (!empty($strings)) {
107 +             foreach ($strings as $id => $originalString) {
108 +                 $sqlCleaned = str_replace($placeholderPrefix . $id . '__', $originalString, $sqlCleaned);
109 +             }
110 +         }
111 +
112 +         return trim($sqlCleaned);
113 +     }
114 + }
```

Comments 0

Please sign in to comment.