parsonsbots / dedicated-ip-checker

Add support for IPv6

Browse files

master v1.0.1 ··· v1.0.0.0

parsonsbots committed 6 days ago 1 parent fb58870 commit 01e8f123ed85817e54960304a6fa4a90bcf9fb69

Showing 4 changed files with 233 additions and 134 deletions.

Unified Split

173 README.md

```
@@ -1,7 +1,7 @@
  # Dedicated IP Checker

  ## Metadata
- * Version 1.0.0
+ * Version 1.0.1
  * Author: Will Parsons
  * Website: https://parsonsbots.com


  ## Changelog:
  * 1.0.0: Initial release for IPv4 only
+ * 1.0.1: Add support for IPv6

  ## What is a Dedicated IP Checker?
  This simple dedicated IP checker was built to be integrated with hosting and VPN / proxy services to provide a way for users to verify dedicated IP address exclusivity.
  ## Usage
  ### Define IP List to Check
        /**
-        * IPv4 list to check separated by new line
-        * This list should be submitted through a public HTML form textarea field
+        * IP list to check separated by new line
         *
-        * Reserved IPv4 addresses below are used for demo purposes
+        * Reserved IP addresses below are used for demo purposes
         *
         */
-              $ipv4ListToCheck = implode("\n", array(
+              $ipListToCheck = implode("\n", array(
+
+ '2001:db8:abcd:0008:847g:3e2:1088:8888',
+                      '2001:db8:abcd:0008::ffff:ffff',
+
+ '2001:db8:abcd:0008:0000:0000:8888:ffff',
+
+ '2001:db8:abcd:0008::1234:8888:FFFF',
                      '172.16.88.10',
                      '172.16.8.3',
                      '172.16.200.3',
                      '10.5.30.105',
                      '10.5.30.106',
                      '10.5.31.108',
-                     '_.10.5.30.199:80;*', // Invalid IP formats will be parsed
+                     '_.10.5.30.199_;*', // Invalid IP formats will be parsed
                      '10.5.89.44',
                      '10.8.8.100',
                      '10.8.8.101',
```

```
77
78    ### Define IPs That Belong to Users
79          /**
80  -        * List of dedicated IPv4 addresses that belong to each user (IPv4 => UUID format)
81           * Users should have access to see their own user ID
82           *
83  -        * Reserved IPv4 addresses below are used for demo purposes
84           *
85           */
86  -       $userIps = array(
87  -               '192.168.0.2' => '5c95396e-97c8-49af-abb3-17e04221ccee',
88  -               '192.168.0.3' => '5c95396e-97c8-49af-abb3-17e04221ccee',
89  -               '10.5.31.108' => '5c95354f-43c8-4976-b69d-12334221ccee',
90  -               '172.16.88.10' => '5c953514-e9e8-4cca-954c-12334221ccee',
91  -               '172.16.200.3' => '5c953514-e9e8-4cca-954c-12334221ccee',
92  -               '10.5.89.44' => '5c952170-f908-45e9-af43-6bf64221ccee',
93  -               '10.5.30.199' => '5c9518c1-0ad8-4e41-80a1-5bd54221ccee',
94  -               '172.16.8.3' => '5c951454-0418-4ca5-80f8-5b6b4221ccee',
95  -               '192.168.40.4' => '5c951242-56a9-401a-8d74-5b7b4221ccee',
96  -               '10.8.8.102' => '5c95396e-97c8-49af-abb3-17e04221ccee'
97  -       );
98
99  -       // Include formatted sample data (this file should be changed to your own database values and user input)
100 -       require_once('data.php');
101
102 -       // Include dedicated IP checker
103 -       require_once('checker.php');
104
105 -       $checker = new DedicatedIpChecker();
106 -       $verifiedIpv4List = $checker->verify($ipv4ListToCheck, $userIps);
107
108 - ### Output
109 -       echo '<pre>';
110
111 -       foreach ($verifiedIpv4List as $ip => $userId) {
112 -               echo 'IP: ' . $ip . "\n";
113 -               echo 'User ID: ' . $userId . "\n\n";
114 -       }
115
116 -       echo '</pre>';
117
118 -       /* Output:
```

```
81
82    ### Define IPs That Belong to Users
83          /**
84  +        * List of dedicated IP addresses that belong to each user (IP => UUID format)
85           * Users should have access to see their own user ID
86           *
87  +        * Reserved IP addresses below are used for demo purposes
88           *
89           */
90  +               $userIps = array(
91  +                       '2001:db8:abcd:0008:0000:0000:8888:ffff' => '5c9518c1-0ad8-4e41-80a1-5bd54221ccee',
92  +                       '192.168.0.2' => '5c95396e-97c8-49af-abb3-17e04221ccee',
93  +                       '192.168.0.3' => '5c95396e-97c8-49af-abb3-17e04221ccee',
94  +                       '10.5.31.108' => '5c95354f-43c8-4976-b69d-12334221ccee',
95  +                       '172.16.88.10' => '5c953514-e9e8-4cca-954c-12334221ccee',
96  +                       '172.16.200.3' => '5c953514-e9e8-4cca-954c-12334221ccee',
97  +                       '10.5.89.44' => '5c952170-f908-45e9-af43-6bf64221ccee',
98  +                       '10.5.30.199' => '5c9518c1-0ad8-4e41-80a1-5bd54221ccee',
99  +                       '172.16.8.3' => '5c951454-0418-4ca5-80f8-5b6b4221ccee',
100 +                       '192.168.40.4' => '5c951242-56a9-401a-8d74-5b7b4221ccee',
101 +                       '10.8.8.102' => '5c95396e-97c8-49af-abb3-17e04221ccee'
102 +               );
103
104 +               // Include formatted sample data (this file should be changed to your own database values and user input)
105 +               require_once('data.php');
106
107 +               // Include dedicated IP checker
108 +               require_once('checker.php');
109
110 +               $checker = new DedicatedIpChecker();
111 +               $verifiedIpList = $checker->verify($ipListToCheck, $userIps);
112
113 + ### Results
114 +               echo '<pre>';
115
116 +               foreach ($verifiedIpList as $ip => $userId) {
117 +                       echo 'IP: ' . $ip . "\n";
118 +                       echo 'User ID: ' . $userId . "\n\n";
119 +               }
120
121 +               echo '</pre>';
122
123 +               /* Results:
```

```
-       IP: 172.16.88.10
-       User ID: 5c953514-e9e8-4cca-954c-12334221ccee

-       IP: 172.16.8.3
-       User ID: 5c951454-0418-4ca5-80f8-5b6b4221ccee

-       IP: 172.16.200.3
-       User ID: 5c953514-e9e8-4cca-954c-12334221ccee

-       IP: 172.16.201.4
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee

-       IP: 192.168.0.0
-       User ID: 5c95396e-97c8-49af-abb3-17e04221ccee

-       IP: 192.168.0.1
-       User ID: 5c95396e-97c8-49af-abb3-17e04221ccee

-       IP: 192.168.0.2
-       User ID: 5c95396e-97c8-49af-abb3-17e04221ccee

-       IP: 192.168.0.3
-       User ID: 5c95396e-97c8-49af-abb3-17e04221ccee

-       IP: 192.168.0.4
-       User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee

-       IP: 192.168.40.4
-       User ID: 5c951242-56a9-401a-8d74-5b7b4221ccee

-       IP: 192.168.49.4
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee

-       IP: 10.5.30.103
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee

-       IP: 10.5.30.105
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee

-       IP: 10.5.30.106
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee

-       IP: 10.5.31.108
-       User ID: 5c95354f-43c8-4976-b69d-12334221ccee

-       IP: 10.5.30.199
-       User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
```

```
+               IP: 2001:db8:abcd:0008:847g:3e2:1088:8888
+               User ID: 5c951454-0418-4ca5-80f8-
5b6b4221ccee

+               IP: 2001:db8:abcd:0008:0000:0000:ffff:ffff
+               User ID: 5c953514-e9e8-4cca-954c-
12334221ccee

+               IP: 2001:db8:abcd:0008:0000:0000:8888:ffff
+               User ID: 5c9518c1-0ad8-4e41-80a1-
5bd54221ccee

+               IP: 2001:db8:abcd:0008:0000:1234:8888:FFFF
+               User ID: 5c953514-e9e8-4cca-954c-
12334221ccee

+               IP: 172.16.88.10
+               User ID: 5c953514-e9e8-4cca-954c-
12334221ccee

+               IP: 172.16.8.3
+               User ID: 5c951454-0418-4ca5-80f8-
5b6b4221ccee

+               IP: 172.16.200.3
+               User ID: 5c953514-e9e8-4cca-954c-
12334221ccee

+               IP: 172.16.201.4
+               User ID: 5c953514-e9e8-4cca-954c-
12334221ccee

+               IP: 192.168.0.0
+               User ID: 5c951242-56a9-401a-8d74-
5b7b4221ccee

+               IP: 192.168.0.1
+               User ID: 5c951242-56a9-401a-8d74-
5b7b4221ccee

+               IP: 192.168.0.2
+               User ID: 5c95396e-97c8-49af-abb3-
17e04221ccee

+               IP: 192.168.0.3
+               User ID: 5c95396e-97c8-49af-abb3-
17e04221ccee

+               IP: 192.168.0.4
+               User ID: 5c95396e-97c8-49af-abb3-
17e04221ccee

+               IP: 192.168.40.4
+               User ID: 5c951242-56a9-401a-8d74-
5b7b4221ccee

+               IP: 192.168.49.4
+               User ID: 5c9518c1-0ad8-4e41-80a1-
5bd54221ccee

+               IP: 10.5.30.103
+               User ID: 5c95396e-97c8-49af-abb3-
```

```
-
-       IP: 10.5.89.44
-       User ID: 5c952170-f908-45e9-af43-6bf64221ccee
-
-       IP: 10.8.8.100
-       User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
-
-       IP: 10.8.8.101
-       User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
-
-       IP: 10.8.8.102
-       User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
-
-       IP: 10.8.8.103
-       User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
-
-       */
-
- ## Coming Soon
- * 1.0.1: IPv6 support
```

```
17e04221ccee
+
+       IP: 10.5.30.105
+       User ID: 5c951454-0418-4ca5-80f8-
5b6b4221ccee
+
+       IP: 10.5.30.106
+       User ID: 5c9518c1-0ad8-4e41-80a1-
5bd54221ccee
+
+       IP: 10.5.31.108
+       User ID: 5c95354f-43c8-4976-b69d-
12334221ccee
+
+       IP: 10.5.30.199
+       User ID: 5c9518c1-0ad8-4e41-80a1-
5bd54221ccee
+
+       IP: 10.5.89.44
+       User ID: 5c952170-f908-45e9-af43-
6bf64221ccee
+
+              IP: 10.8.8.100
+              User ID: 5c953514-e9e8-4cca-954c-
12334221ccee
+
+              IP: 10.8.8.101
+              User ID: 5c953514-e9e8-4cca-954c-
12334221ccee
+
+              IP: 10.8.8.102
+              User ID: 5c95396e-97c8-49af-abb3-
17e04221ccee
+
+              IP: 10.8.8.103
+              User ID: 5c95396e-97c8-49af-abb3-
17e04221ccee
+
+              */
+
  ## More From ParsonsBots
  - https://github.com/parsonsbots
  - https://gist.github.com/parsonsbots
  - https://parsonsbots.com
+ - https://eightomic.com
```

171 ■■■■□ checker.php

```
...  @@ -1,93 +1,171 @@
 1    <?php
 2    /**
-  * Dedicated IPv4 Address Checker
 3  + * Dedicated IP Address Checker
 4     *
-  * Allow users to publicly verify dedicated IPv4
addresses allocated to their unique, private account or
user ID.
 5  + * Allow users to publicly verify dedicated IPv4 and
IPv6 addresses allocated to their unique, private account
or user ID.
 6     *
-  * @author Will Parsons
-  * @link   https://parsonsbots.com
 7  + * @copyright 2019 Will Parsons
 8  + * @license   https://github.com/parsonsbots/dedicated-
ip-checker/blob/master/LICENSE // MIT License
 9  + * @link      https://parsonsbots.com
10  + * @link      https://eightomic.com
 9     */
11  +
```

```
 class DedicatedIpChecker {

 /**
-  * Get user ID for an IPv4 address
+  * Get user ID for an IP address
+  *
+  * @param string $ip
+  * @param array $userIps
+  * @param array $userIds
   *
-  * @param string $ip IPv4 address
-  * @param array $userIps User IPs
-  * @param array $userIds User IDs
-  * @return string $userId User ID for dedicated IP (if user ID doesn't exist for IP, pick a UUID from existing user IDs)
+  * @return string $response User ID for dedicated IP (if user ID doesn't exist for IP, pick a UUID from existing user IDs)
   */
     protected function _getUserId($ip, $userIps, $userIds) {
         if (!empty($userIps[$ip])) {
             return $userIps[$ip];
         }

-        $subnets = explode('.', $ip);
-        $key = ($subnets[0] != 0 ? $subnets[0] : 1) * ($subnets[3] != 0 ? $subnets[3] : 1) + $subnets[2] + $subnets[1];
-        return ($userIds[$key]);
+        if (strpos($ip, ':') !== false) {
+            $characters = 'abcdefghijklmnopqrstuvwxyzbcdfghjklmnpqrstvwxyz';
+
+            $ipCharacters = str_replace(array(0), '', strtolower($ip));
+
+            for ($i = 0; $i < strlen($ipCharacters); $i++) {
+                if (($numericValue = strpos($characters, $ipCharacters[$i])) !== false) {
+                    $ipCharacters[$i] = $numericValue + 1;
+                }
+            }
+
+            $subnets = array_filter(explode(':', $ipCharacters));
+            $key = array_sum($subnets) * count($subnets);
+        } else {
+            $subnets = explode('.', $ip);
+            $key = ($subnets[0] != 0 ? $subnets[0] : 1) * ($subnets[3] != 0 ? $subnets[3] : 1) + $subnets[2] + $subnets[1];
+        }
+
+        $response = ($userIds[$key]);
+        return $response;
     }

 /**
-  * Parse and filter IPv4 address list.
+  * Parse and filter IP address list
+  *
+  * @param mixed [array/string] $ips
   *
-  * @param array $ips Unfiltered IPv4 address list
-  * @return array $ips Filtered IPv4 address list
+  * @return array $response
```

```
   */
        protected function _parseIps($ips = array()) {
-               $ips = implode("\n",
array_map(function($ip) {
-                       return trim($ip, '.');
-               }, array_filter(preg_split("/[]
(\r\n|\n|\r) @#$+,[;:_-]/", $ips))));
-               $ips = $this->_validateIps($ips);
-               return explode("\n", $ips);
        }

  /**
-  * Validate IPv4 address list.
   *
-  * @param array $ips Filtered IPv4 address list
-  * @return array $ips Validated IPv4 address list
   */
-       protected function _validateIps($ips) {
-               $ips =
array_values(array_filter(explode("\n", $ips)));
-
-               foreach ($ips as $key => $ip) {
-                       $splitIpSubnets =
array_map('trim', explode('.', trim($ip)));
-
-                       if (count($splitIpSubnets) != 4) {
-                               unset($ips[$key]);
-                               continue;
-                       }

                        foreach ($splitIpSubnets as
$splitIpSubnet) {
                                if (
!is_numeric($splitIpSubnet) ||
strlen($splitIpSubnet) > 3 ||
                                        $splitIpSubnet >
255 ||
                                        $splitIpSubnet < 0
                                ) {
-                                       unset($ips[$key]);
-                                       continue;
                                }
                        }

-                       $ips[$key] = $splitIpSubnets[0] .
'.' . $splitIpSubnets[1] . '.' . $splitIpSubnets[2] . '.'
. $splitIpSubnets[3];
                }

-               return implode("\n", array_unique($ips));
```

```
   */
        protected function _parseIps($ips = array()) {
+               if (!is_array($ips)) {
+                       $ips =
array_filter(preg_split("/[](\r\n|\n|\r) <>()~
{}|`\"'=?!*&@#$+,[;_-]/", $ips));
+               }
+
+               $response = $this->_validateIps($ips);
+               return $response;
        }

  /**
+  * Validate IPv4 address
+  *
+  * @param string $ip
   *
+  * @return mixed [boolean/string] $response
   */
+       protected function _validateIpv4($ip) {
+               $response = false;
+               $splitIpSubnets = explode('.', $ip);

+               if (count($splitIpSubnets) === 4) {
                        foreach ($splitIpSubnets as
$splitIpSubnet) {
                                if (
!is_numeric($splitIpSubnet) ||
strlen($splitIpSubnet) > 3 ||
                                        $splitIpSubnet >
255 ||
                                        $splitIpSubnet < 0
                                ) {
+                                       return false;
                                }
                        }

+                       $response = $ip;

                }

+               return $response;
+       }
+
+ /**
+  * Validate IPv6 address
+  *
```

```
+  * @param string $ip
+  *
+  * @return mixed [boolean/string] $response
+  */
+       protected function _validateIpv6($ip) {
+               $response = false;
+
+               if (strpos($ip, '::') !== false) {
+                       $ip = str_replace('::', str_repeat(':0000', 7 - (substr_count($ip, ':') - 1)) . ':', $ip);
+               }
+
+               $splitIpSubnets = explode(':', $ip);
+
+               if (count($splitIpSubnets) === 8) {
+                       foreach ($splitIpSubnets as $splitIpSubnet) {
+                               if (strlen($splitIpSubnet) > 4) {
+                                       return false;
+                               }
+                       }
+
+                       $response = $ip;
+               }
+
+               return $response;
+       }
+
+ /**
+  * Validate IP address list
+  *
+  * @param array $ips
+  *
+  * @return array $response
+  */
+       protected function _validateIps($ips) {
+               foreach ($ips as $key => $ip) {
+                       if (
+                               empty($ip) ||
+                               strlen($ip) < 7 ||
+                               !($ip = trim($ip, '.')) ||
+                               !($ip = trim($ip, ':')) ||
+                               (
+                                       strpos($ip, ':::') === false &&
+                                       substr_count($ip, ':') > 4 &&
+                                       ($ips[$key] = $this->_validateIpv6($ip)) === false
+                               ) ||
+                               (
+                                       strpos($ip, ':') === false &&
+                                       ($ips[$key] = $this->_validateIpv4($ip)) === false
+                               )
+                       ) {
+                               unset($ips[$key]);
+                       }
+               }
+
```

```
77          }
78
79      /**
80       * Verify dedicated IPs
81       *
82      -  * @param array $ipv4List List of IPv4 addresses separated by new line
83      -  * @param array $userIps List of dedicated IPv4 addresses that belong to each unique user ID (IPv4 => UUID format)
84      -  * @return $userIds
85       */
86      -       public function verify($ipv4List, $userIps) {
87      -               $ipv4List = $this->_parseIps($ipv4List);
88                      $userIds = array_values($userIps);
89
90      -               if (!empty($ipv4List)) {
91                              $userIdsFormatted = false;
92
93                              do {
99                              } while ($userIdsFormatted == false);
100                     }
101
102     -               foreach ($ipv4List as $key => $ip) {
103                             $ips[$ip] = $this->_getUserId($ip, $userIps, $userIds);
104                             unset($ips[$key]);
105                     }
106
107     -               return array_merge($ips, $userIps);
108
109         }
110
111     }
```

```
152     +               $response = implode("\n", array_unique($ips));
153     +               return $response;
154         }
155
156     /**
157      * Verify dedicated IPs
158      *
159     +  * @param array $ips
160     +  * @param array $userIps
161     +  *
162     +  * @return $response
163      */
164     +       public function verify($ips, $userIps) {
165     +               $ips = $this->_parseIps($ips);
166                     $userIds = array_values($userIps);
167
168     +               if (!empty($ips)) {
169                             $userIdsFormatted = false;
170
171                             do {
177                             } while ($userIdsFormatted == false);
178                     }
179
180     +               $ips = explode("\n", $ips);
181     +
182     +               foreach ($ips as $key => $ip) {
183                             $ips[$ip] = $this->_getUserId($ip, $userIps, $userIds);
184                             unset($ips[$key]);
185                     }
186
187     +               $response = array_merge($ips, $userIps);
188     +               return $response;
189         }
190
191     }
```

18 data.php

```
...  @@ -1,12 +1,15 @@
1    <?php
2    /**
3    -  * IPv4 list to check separated by new line
4    -  * This list should be submitted through a public HTML form textarea field
5     *
6    -  * Reserved IPv4 addresses below are used for demo purposes
7     *
8     */
9    -       $ipv4ListToCheck = implode("\n", array(
10                  '172.16.88.10',
11                  '172.16.8.3',
12                  '172.16.200.3',
```

```
1    <?php
2    /**
3    +  * IP list to check separated by new line
4     *
5    +  * Reserved IP addresses below are used for demo purposes
6     *
7     */
8    +       $ipListToCheck = implode("\n", array(
9    +               '2001:db8:abcd:0008:847g:3e2:1088:8888',
10   +               '2001:db8:abcd:0008::ffff:ffff',
11   +               '2001:db8:abcd:0008:0000:0000:8888:ffff',
12   +               '2001:db8:abcd:0008::1234:8888:FFFF',
13                  '172.16.88.10',
14                  '172.16.8.3',
15                  '172.16.200.3',
```

```
22                '10.5.30.105',
23                '10.5.30.106',
24                '10.5.31.108',
25 -              '_.10.5.30.199:80;*', // Invalid IP formats will be parsed
26                '10.5.89.44',
27                '10.8.8.100',
28                '10.8.8.101',
31        ));
32
33    /**
34 -   * List of dedicated IPv4 addresses that belong to each user (IPv4 => UUID format)
35    * Users should have access to see their own user ID
36    *
37 -   * Reserved IPv4 addresses below are used for demo purposes
38    *
39    */
40        $userIps = array(
41                '192.168.0.2' => '5c95396e-97c8-49af-abb3-17e04221ccee',
42                '192.168.0.3' => '5c95396e-97c8-49af-abb3-17e04221ccee',
43                '10.5.31.108' => '5c95354f-43c8-4976-b69d-12334221ccee',
```

```
25                '10.5.30.105',
26                '10.5.30.106',
27                '10.5.31.108',
28 +              '_.10.5.30.199_;*', // Invalid IP formats will be parsed
29                '10.5.89.44',
30                '10.8.8.100',
31                '10.8.8.101',
34        ));
35
36    /**
37 +   * List of dedicated IP addresses that belong to each user (IP => UUID format)
38    * Users should have access to see their own user ID
39    *
40 +   * Reserved IP addresses below are used for demo purposes
41    *
42    */
43        $userIps = array(
44 +              '2001:db8:abcd:0008:0000:0000:8888:ffff' => '5c9518c1-0ad8-4e41-80a1-5bd54221ccee',
45                '192.168.0.2' => '5c95396e-97c8-49af-abb3-17e04221ccee',
46                '192.168.0.3' => '5c95396e-97c8-49af-abb3-17e04221ccee',
47                '10.5.31.108' => '5c95354f-43c8-4976-b69d-12334221ccee',
```

5 ■■■■ demo.php

```
...  @@ -1,13 +1,12 @@
1     <?php
2         require_once('data.php');
3         require_once('checker.php');
4 -
5         $checker = new DedicatedIpChecker();
6 -       $verifiedIpv4List = $checker->verify($ipv4ListToCheck, $userIps);
7
8         echo '<pre>';
9
10 -      foreach ($verifiedIpv4List as $ip => $userId) {
11                echo 'IP: ' . $ip . "\n";
12                echo 'User ID: ' . $userId . "\n\n";
13        }
```

```
1     <?php
2         require_once('data.php');
3         require_once('checker.php');
4         $checker = new DedicatedIpChecker();
5 +       $verifiedIpList = $checker->verify($ipListToCheck, $userIps);
6
7         echo '<pre>';
8
9 +       foreach ($verifiedIpList as $ip => $userId) {
10                echo 'IP: ' . $ip . "\n";
11                echo 'User ID: ' . $userId . "\n\n";
12        }
```

0 comments on commit `01e8f12`