



Initial v1.0.0 commit

[Browse files](#) master  v1.0.1 ... v1.0.0 parsonsbots committed on Mar 231 parent [09ef119](#)

commit df951b72722b5760383c2bdf1fa240bccb0171fc

 Showing 4 changed files with 371 additions and 0 deletions.

Unified

Split

▼ 112  checker.php 

... @@ -0,0 +1,112 @@

```
1 + <?php
2 + /**
3 +  * Dedicated IPv4 Address Checker
4 +  *
5 +  * Allow users to publicly verify dedicated IPv4
6 +  * addresses allocated to their unique, private account or
7 +  * user ID.
8 +  *
9 +  * @author Will Parsons
10 +  * @link https://parsonsbots.com
11 +  */
12 + class DedicatedIpChecker {
13 +
14 +     /**
15 +      * Get user ID for an IPv4 address
16 +      *
17 +      * @param string $ip IPv4 address
18 +      * @param array $userIps User IPs
19 +      * @param array $userIds User IDs
20 +      * @return string $userId User ID for dedicated IP (if
21 +      * user ID doesn't exist for IP, pick a UUID from existing
22 +      * user IDs)
23 +      */
24 +     protected function _getUserId($ip, $userIps,
25 + $userIds) {
26 +         if (!empty($userIps[$ip])) {
27 +             return $userIps[$ip];
28 +         }
29 +
30 +         $subnets = explode('.', $ip);
31 +         $key = ($subnets[0] != 0 ? $subnets[0] :
32 + 1) * ($subnets[3] != 0 ? $subnets[3] : 1) + $subnets[2] +
33 + $subnets[1];
34 +         return ($userIds[$key]);
35 +     }
36 +
37 +     /**
38 +      * Parse and filter IPv4 address list.
39 +      *
40 +      * @param array $ips Unfiltered IPv4 address list
41 +      * @return array $ips Filtered IPv4 address list
42 +      */
43 +     protected function _parseIps($ips = array()) {
44 +         $ips = implode("\n",
45 + array_map(function($ip) {
46 +             return trim($ip, '.');
47 +         }, array_filter(preg_split("/[
```

```

(\r\n|\n|\r) @#$,[:_-]/", $ips))));
40 +         $ips = $this->_validateIps($ips);
41 +         return explode("\n", $ips);
42 +     }
43 +
44 + /**
45 +  * Validate IPv4 address list.
46 +  *
47 +  * @param array $ips Filtered IPv4 address list
48 +  * @return array $ips Validated IPv4 address list
49 +  */
50 +     protected function _validateIps($ips) {
51 +         $ips =
array_values(array_filter(explode("\n", $ips)));
52 +
53 +         foreach ($ips as $key => $ip) {
54 +             $splitIpSubnets =
array_map('trim', explode('.', trim($ip)));
55 +
56 +             if (count($splitIpSubnets) != 4) {
57 +                 unset($ips[$key]);
58 +                 continue;
59 +             }
60 +
61 +             foreach ($splitIpSubnets as
$splitIpSubnet) {
62 +                 if (
63 +
!is_numeric($splitIpSubnet) ||
64 +
strlen($splitIpSubnet) > 3 ||
65 +
$splitIpSubnet >
255 ||
66 +
$splitIpSubnet < 0
67 +
) {
68 +                 unset($ips[$key]);
69 +                 continue;
70 +             }
71 +
72 +
73 +             $ips[$key] = $splitIpSubnets[0] .
'.' . $splitIpSubnets[1] . '.' . $splitIpSubnets[2] . '.'
. $splitIpSubnets[3];
74 +         }
75 +
76 +         return implode("\n", array_unique($ips));
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 +  *
85 +  * @return $userIds
86 +  */
87 +     public function verify($ipv4List, $userIps) {
88 +         $ipv4List = $this->_parseIps($ipv4List);
89 +         $userIds = array_values($userIps);
90 +

```

```

91 +         if (!empty($ipv4List)) {
92 +             $userIdsFormatted = false;
93 +
94 +             do {
95 +                 $userIds =
array_merge($userIds, $userIds);
96 +
97 +                 if (count($userIds) >
255255) {
98 +                     $userIdsFormatted
= true;
99 +                 }
100 +             } while ($userIdsFormatted ==
false);
101 +         }
102 +
103 +         foreach ($ipv4List as $key => $ip) {
104 +             $ips[$ip] = $this->_getUserId($ip,
$userIps, $userIds);
105 +             unset($ips[$key]);
106 +         }
107 +
108 +         return array_merge($ips, $userIps);
109 +     }
110 +
111 + }
112 + ?>

```

▼ 52 data.php

... @@ -0,0 +1,52 @@

```

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',
13 +         '172.16.201.4',
14 +         '192.168.0.0',
15 +         '192.168.0.1',
16 +         '192.168.0.2',
17 +         '192.168.0.3',
18 +         '192.168.0.4',
19 +         '192.168.40.4',
20 +         '192.168.49.4',
21 +         '10.5.30.103',
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',
29 +         '10.8.8.102',

```

```

30 +         '10.8.8.103'
31 +     ));
32 +
33 + /**
34 +  * List of dedicated IPv4 addresses that belong to each
35 +  * user (IPv4 => UUID format)
36 +  * Users should have access to see their own user ID
37 +  * Reserved IPv4 addresses below are used for demo
38 +  * purposes
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',
44 +         '172.16.88.10' => '5c953514-e9e8-4cca-954c-
12334221ccee',
45 +         '172.16.200.3' => '5c953514-e9e8-4cca-954c-
12334221ccee',
46 +         '10.5.89.44' => '5c952170-f908-45e9-af43-
6bf64221ccee',
47 +         '10.5.30.199' => '5c9518c1-0ad8-4e41-80a1-
5bd54221ccee',
48 +         '172.16.8.3' => '5c951454-0418-4ca5-80f8-
5b6b4221ccee',
49 +         '192.168.40.4' => '5c951242-56a9-401a-8d74-
5b7b4221ccee',
50 +         '10.8.8.102' => '5c95396e-97c8-49af-abb3-
17e04221ccee'
51 +     );
52 + ?>

```

▼ 16 demo.php

... @@ -0,0 +1,16 @@

```

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 +     }
14 +
15 +     echo '</pre>';
16 + ?>

```

▼ 191 readme.md

... @@ -0,0 +1,191 @@

```

1 + # Dedicated IP Checker
2 + * Version 1.0.0
3 + * Author: Will Parsons

```

```
4 + * Website: https://parsonsbots.com
5 +
6 + # Repository
7 + * GIT: git@github.com:parsonsbots/dedicated-ip-
  checker.git
8 +
9 + # Required:
10 + PHP 5.x+
11 +
12 + # Changelog:
13 + * 1.0.0: Initial release for IPv4 only
14 +
15 + # What is a Dedicated IP Checker?
16 + 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.
17 +
18 + A dedicated IP is an IP address that's only provided to
  a single user as a usage preference and for security
  purposes. A semi-dedicated IP is an IP address that's
  provided to multiple users for affordability.
19 +
20 + If a user purchases a dedicated IP for their hosting,
  VPN or proxy server, they should simply be able to verify
  that the IP is exclusive to their account / user ID.
21 +
22 + This dedicated IP checker is improving the standard of
  transparency where dedicated IP addresses are purchased
  and will help prevent them from being shared between
  users (intentionally or unintentionally).
23 +
24 + Why are dedicated IPs important and why should users be
  able to verify dedicated IP exclusivity?
25 +
26 + - Websites may eventually block your dedicated IP if
  it's shared with other users and too many connections are
  coming from the same shared network IP.
27 + - When using a dedicated IP to whitelist access to
  sensitive systems (through a VPN, proxy server, etc),
  multiple users should never have access to that same IP
  for security purposes.
28 + - Users can lower the possibility of frequent CAPTCHAs
  and IP blacklisting when browsing.
29 +
30 + # How Does it Work?
31 + You can enable users to search and verify dedicated IP
  exclusivity for a list of dedicated IPs without them
  being logged-in to their account.
32 +
33 + Each user should already have their own unique user ID
  to correspond with their purchased dedicated IPs.
34 +
35 + Here's an example of the verification process:
36 +
37 + 1. User logs into your website's hosting portal or VPN /
  proxy control panel.
38 + 2. User retrieves their unique user account ID and
  dedicated IPs you've provided them.
39 + 3. User logs out of their account.
40 + 4. User visits the dedicated IP checker page on your
  website.
41 + 5. User inputs the list of dedicated IPs to see if they
```

```

correspond with their unique user account ID.
42 +
43 + Inactive IPs, semi-dedicated IPs, shared IPs, reserved
  IPs or IPs that aren't related to your website will be
  masked with random user IDs for security and privacy
  purposes.
44 +
45 + For an example of an existing implementation, please
  visit the GhostProxies dedicated proxy IP checker at
  https://ghostproxies.com/dedicated-proxy-checker
46 +
47 + # Usage
48 + ## Define IP List to Check
49 + /**
50 +     * IPv4 list to check separated by new line
51 +     * This list should be submitted through a public
  HTML form textarea field
52 +     *
53 +     * Reserved IPv4 addresses below are used for demo
  purposes
54 +     *
55 +     */
56 +     $ipv4ListToCheck = implode("\n", array(
57 +         '172.16.88.10',
58 +         '172.16.8.3',
59 +         '172.16.200.3',
60 +         '172.16.201.4',
61 +         '192.168.0.0',
62 +         '192.168.0.1',
63 +         '192.168.0.2',
64 +         '192.168.0.3',
65 +         '192.168.0.4',
66 +         '192.168.40.4',
67 +         '192.168.49.4',
68 +         '10.5.30.103',
69 +         '10.5.30.105',
70 +         '10.5.30.106',
71 +         '10.5.31.108',
72 +         '_.10.5.30.199:80;*'; // Invalid
  IP formats will be parsed
73 +         '10.5.89.44',
74 +         '10.8.8.100',
75 +         '10.8.8.101',
76 +         '10.8.8.102',
77 +         '10.8.8.103'
78 +     ));
79 +
80 + ## Define IPs That Belong to Users
81 + /**
82 +     * List of dedicated IPv4 addresses that belong to
  each user (IPv4 => UUID format)
83 +     * Users should have access to see their own user
  ID
84 +     *
85 +     * Reserved IPv4 addresses below are used for demo
  purposes
86 +     *
87 +     */
88 +     $userIps = array(
89 +         '192.168.0.2' => '5c95396e-97c8-49af-abb3-
  17e04221ccee',
90 +         '192.168.0.3' => '5c95396e-97c8-49af-abb3-
  17e04221ccee',

```

```

91 +             '10.5.31.108' => '5c95354f-43c8-4976-b69d-
12334221ccee',
92 +             '172.16.88.10' => '5c953514-e9e8-4cca-
954c-12334221ccee',
93 +             '172.16.200.3' => '5c953514-e9e8-4cca-
954c-12334221ccee',
94 +             '10.5.89.44' => '5c952170-f908-45e9-af43-
6bf64221ccee',
95 +             '10.5.30.199' => '5c9518c1-0ad8-4e41-80a1-
5bd54221ccee',
96 +             '172.16.8.3' => '5c951454-0418-4ca5-80f8-
5b6b4221ccee',
97 +             '192.168.40.4' => '5c951242-56a9-401a-
8d74-5b7b4221ccee',
98 +             '10.8.8.102' => '5c95396e-97c8-49af-abb3-
17e04221ccee'
99 +         );
100 +
101 +         // Include formatted sample data (this file should
be changed to your own database values and user input)
102 +         require_once('data.php');
103 +
104 +         // Include dedicated IP checker
105 +         require_once('checker.php');
106 +
107 +         $checker = new DedicatedIpChecker();
108 +         $verifiedIpv4List = $checker-
>verify($ipv4ListToCheck, $userIps);
109 +
110 + ## Output
111 +         echo '<pre>';
112 +
113 +         foreach ($verifiedIpv4List as $ip => $userId) {
114 +             echo 'IP: ' . $ip . "\n";
115 +             echo 'User ID: ' . $userId . "\n\n";
116 +         }
117 +
118 +         echo '</pre>';
119 +
120 +         /* Output:
121 +
122 +         IP: 172.16.88.10
123 +         User ID: 5c953514-e9e8-4cca-954c-12334221ccee
124 +
125 +         IP: 172.16.8.3
126 +         User ID: 5c951454-0418-4ca5-80f8-5b6b4221ccee
127 +
128 +         IP: 172.16.200.3
129 +         User ID: 5c953514-e9e8-4cca-954c-12334221ccee
130 +
131 +         IP: 172.16.201.4
132 +         User ID: 5c952170-f908-45e9-af43-6bf64221ccee
133 +
134 +         IP: 192.168.0.0
135 +         User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
136 +
137 +         IP: 192.168.0.1
138 +         User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
139 +
140 +         IP: 192.168.0.2
141 +         User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
142 +
143 +         IP: 192.168.0.3

```

```
144 + User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
145 +
146 + IP: 192.168.0.4
147 + User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
148 +
149 + IP: 192.168.40.4
150 + User ID: 5c951242-56a9-401a-8d74-5b7b4221ccee
151 +
152 + IP: 192.168.49.4
153 + User ID: 5c952170-f908-45e9-af43-6bf64221ccee
154 +
155 + IP: 10.5.30.103
156 + User ID: 5c952170-f908-45e9-af43-6bf64221ccee
157 +
158 + IP: 10.5.30.105
159 + User ID: 5c952170-f908-45e9-af43-6bf64221ccee
160 +
161 + IP: 10.5.30.106
162 + User ID: 5c952170-f908-45e9-af43-6bf64221ccee
163 +
164 + IP: 10.5.31.108
165 + User ID: 5c95354f-43c8-4976-b69d-12334221ccee
166 +
167 + IP: 10.5.30.199
168 + User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
169 +
170 + IP: 10.5.89.44
171 + User ID: 5c952170-f908-45e9-af43-6bf64221ccee
172 +
173 + IP: 10.8.8.100
174 + User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
175 +
176 + IP: 10.8.8.101
177 + User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
178 +
179 + IP: 10.8.8.102
180 + User ID: 5c95396e-97c8-49af-abb3-17e04221ccee
181 +
182 + IP: 10.8.8.103
183 + User ID: 5c9518c1-0ad8-4e41-80a1-5bd54221ccee
184 +
185 + */
186 +
187 + # Coming Soon
188 + * 1.1.0: IPv6 support
189 +
190 + # More From ParsonsBots
191 + https://github.com/parsonsbots
```

0 comments on commit df951b7