



1146 lines (1016 loc) · 46.7 KB

CodeBlame

RawCopyDownloadCode

```
1  #
2  # License:
3  #
4  #   Copyright (c) 2003-2006 ossim.net
5  #   Copyright (c) 2007-2014 AlienVault
6  #   All rights reserved.
7  #
8  #   This package is free software; you can redistribute it and/or modify
9  #   it under the terms of the GNU General Public License as published by
10 #   the Free Software Foundation; version 2 dated June, 1991.
11 #   You may not use, modify or distribute this program under any other version
12 #   of the GNU General Public License.
13 #
14 #   This package 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 General Public License for more details.
18 #
19 #   You should have received a copy of the GNU General Public License
20 #   along with this package; if not, write to the Free Software
21 #   Foundation, Inc., 51 Franklin St, Fifth Floor, Boston,
22 #   MA 02110-1301 USA
23 #
24 #
25 # On Debian GNU/Linux systems, the complete text of the GNU General
26 # Public License can be found in `/usr/share/common-licenses/GPL-2'.
```

```
27     #
28     # Otherwise you can read it here: http://www.gnu.org/licenses/gpl-2.0.txt
29     #
30
31     #
32     # GLOBAL IMPORTS
33     #
34     import datetime
35     import json
36     import os
37     import pickle
38     import re
39     import socket
40     import time
41     from hashlib import md5
42
43     import GeoIP
44
45     GEOIPDB = GeoIP.open("/usr/share/geoip/GeoLiteCity.dat", GeoIP.GEOIP_STANDARD)
46     #
47     # LOCAL IMPORTS
48     #
49
50     from SiteProtectorMap import *
51     from NetScreenMap import *
52     from Logger import Logger
53
54     logger = Logger.logger
55
56     #
57     # GLOBAL VARIABLES
58     #
59     DEFAULT_ID = '99999'
60     DATE_FORMAT_FILE_PATH = '/etc/ossim/agent/plugins/date_config/date_formats.json'
61     CUSTOM_FORMATS_LOADED = False
62     HOST_RESOLV_CACHE = {}
63
64     ✓ PROTO_TABLE = {
65         '1': 'icmp',
66         '6': 'tcp',
67         '17': 'udp',
68     }
69
70     HOST_BLACK_LIST = {}
71
72     ✓ FIXED_MONTH_TRANSLATE = {
```

```
73     # ENGLISH
74     'jan': 1,
75     'feb': 2,
76     'mar': 3,
77     'apr': 4,
78     'may': 5,
79     'jun': 6,
80     'jul': 7,
81     'aug': 8,
82     'sep': 9,
83     'oct': 10,
84     'nov': 11,
85     'dec': 12,
86     'january': 1,
87     'february': 2,
88     'march': 3,
89     'april': 4,
90     # 'May': 5,
91     'june': 6,
92     'july': 7,
93     'august': 8,
94     'september': 9,
95     'october': 10,
96     'november': 11,
97     'december': 12,
98     # SPANISH
99     'ene': 1,
100    # 'feb': 2,
101    # 'mar': 3,
102    'abr': 4,
103    # 'may': 5,
104    # 'jun': 6,
105    # 'jul': 7,
106    'ago': 8,
107    # 'sep': 9,
108    # 'oct': 10,
109    # 'nov': 11,
110    'dic': 12,
111    'enero': 1,
112    'febrero': 2,
113    'marzo': 3,
114    'abril': 4,
115    'mayo': 5,
116    'junio': 6,
117    'julio': 7,
118    'agosto': 8
```

110 agent = Agent(),


```
1073     HOST_RESOLV_DYNAMIC_CACHE = {}
1074
1075     # Dynamic host-ip cache.
1076     def refreshCache(data):
1077         """ Refresh the HOST dynamic cache """
1078         # action="refresh_asset_list" list={ossim-unstable-pro=192.168.2.18,crosa=192.168.2.130} id
1079         logger.debug("Updating dynamic host cache... %s" % data)
1080         """ Update HOST_RESOLV_DYNAMIC_CACHE """
```



```
1080         # HostResolv.HOST_RESOLV_DYNAMIC_CACHE.clear()
1081         pattern = "action=\"refresh_asset_list\"\\s+list={(?P<list>.*)}"
1082         ipv4_reg = "\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}"
1083         hostname_valid = "(([a-zA-Z]|[a-zA-Z][a-zA-Z0-9\\-]*[a-zA-Z0-9])\\.)*([A-Za-z]|[A-Za-z][A-Za-0-9\\-]*[A-Za-z0-9])"
1084         reg_comp = re.compile(pattern)
1085         res = reg_comp.match(data)
1086         host_list = []
1087         new_cache = {}
1088         if res is not None:
1089             tmp_list = res.group('list')
1090             if tmp_list is not None:
1091                 host_list = tmp_list.split(';')
1092                 logger.debug("HOST_LIST: %s" % host_list)
1093                 for asset in host_list:
1094                     if asset == '':
1095                         continue
1096                     ip, hostnames = asset.split('=')
1097                     hostname_list = hostnames.split(',')
1098                     logger.debug("IP = %s , hostnamelist: %s" % (ip, hostname_list))
1099                     for hostname in hostname_list:
1100                         hostname = hostname.strip()
1101                         hostname = hostname.lower()
1102                         if re.match(ipv4_reg, ip) and re.match(hostname_valid, hostname):
1103                             if new_cache.has_key(hostname):
1104                                 if ip not in new_cache[hostname]:
1105                                     new_cache[hostname].append(ip)
1106                             else:
1107                                 new_cache[hostname] = []
1108                                 new_cache[hostname].append(ip)
1109
1110         HostResolv.HOST_RESOLV_DYNAMIC_CACHE = new_cache
1111
1112         HostResolv.printCache()
1113         HostResolv.saveHostCache()
1114
1115         refreshCache = staticmethod(refreshCache)
1116
1117         def saveHostCache():
1118             logger.info("Saving dynamic host cache in /etc/ossim/agent/host_cache.dic")
1119             pickle.dump(HostResolv.HOST_RESOLV_DYNAMIC_CACHE, open("/etc/ossim/agent/host_cache.dic", 'w'))
1120
1121         saveHostCache = staticmethod(saveHostCache)
1122
1123         def loadHostCache():
1124             if os.path.isfile("/etc/ossim/agent/host_cache.dic"):
1125                 try:
```

```
1126         logger.debug("Loading dynamic host cache from '/etc/ossim/agent/host_cache.dic'")
1127         HostResolv.HOST_RESOLV_DYNAMIC_CACHE = pickle.load(open("/etc/ossim/agent/host_cach
1128         HostResolv.printCache()
1129     except:
1130         logger.warning("Deleting corrupt file host_cache_pro.dic")
1131         os.remove("/etc/ossim/agent/host_cache_pro.dic")
1132         return False
1133     else:
1134         return False
1135
1136     return True
1137
1138 loadHostCache = staticmethod(loadHostCache)
1139
1140 ✓ def printCache():
1141     logger.debug("----- Dynamic cache -----")
1142     for host, ip in HostResolv.HOST_RESOLV_DYNAMIC_CACHE.items():
1143         logger.debug("%s ----->> %s" % (host, ip))
1144     logger.debug("-----")
1145
1146 printCache = staticmethod(printCache)
```