```
#!/usr/bin/python

'''
author: Hopeless
task: final!!! ping scanner and port banners
        The software can perform:
        - Test ip-is-online
        - Checking whether the port is open at the IP-online
        - If an open port system shows banner
        - The system exports data file with all of the scan

        You can enter at the Ip, ip / sider, ip-range, http
        You can enter a list of ports, port manually or scan the list of ports are recommended.
        You can select the file name to export the data or the system automatically saves

'''
import sys, socket , getopt ,subprocess
from scapy.all import *
from netaddr import *
from datetime import datetime

# clear the screen
subprocess.call('clear',shell=True)
socket.setdefaulttimeout(2)
TIMEOUT = 2
conf.verb = 0
portlist = [20,21,22,25,3389,80,443]
argv = sys.argv[1:]
subnet = []
output = 'save.txt'
answers = []
liveIPs = []

#Checking what has been entered
try:
        opts,args = getopt.getopt(argv,'hi:p:o:')
except getopt.GetoptError:
        print 'portscanner.py -i ip/sider or ip-range or http \nfor more exmple use : portscanner.py -h'
        sys.exit(2)
for option,arg in opts:
        if option in '-h':
                print '\nexmple:'
                print 'portscanner.py -i 10.0.0.1/24 or 10.0.0.1-15 or www.debka.co.il \n'
                print '-p 20,22,100,101  \n(if this option is not selected, automatically are who chose the recommended port)\n'
                print '-o save.txt \n(if this option is not selected, automatically this file save in save.txt)\n'
                print 'for exmple:\nportscanner.py -i 10.0.0.1-3 -p 22,3389 -o ports.txt'
                sys.exit()
        if option in '-i':
                subnet = arg
        elif option in '-o':
                output = arg
        elif option in ('-p'):
                portadd = []
                portcheck = arg.split(',')
                for portsplit in portcheck:
                        portadd.append(int(portsplit))
                portlist = portadd
        else:
                print 'portscanner.py -i ip/sider or ip-range or http \nfor more exmple use : portscanner.py -h'
                sys.exit()

#checks if ip/sider or ip-range or http
if subnet:
        try:
                subnet = IPNetwork(subnet)
        except Exception, e:
                if '-' in subnet:
                        ipall =[]
```

```
                        subnet = subnet.split('-')
                        ipfirst = subnet[0]
                        ips = ipfirst.split('.')
                        iptmp = ips[0]+'.'+ips[1]+'.'+ips[2]+'.'
                        for i in range(int(ips[3]),(int(subnet[1])+1)):
                                ip = iptmp+str(i)
                                ipall.append(ip)
                        subnet = ipall
                elif subnet[0].isalpha():
                        subnethost = socket.gethostbyname(arg)
                        subnet = IPNetwork(subnethost)
                else:
                        print 'bad ip or url , for help (portscanner.py -h)'
                        sys.exit()

while not subnet:
        print 'portscanner.py -i ip/sider or ip-range or http'
        print 'for exmple:\nportscanner.py -i 10.0.0.1-3 -p 22,3389 -o ports.txt'
        print 'for more exmple use : portscanner.py -h'
        sys.exit(2)
#If the initial request was incorrect ,checks if ip/sider or ip-range or http
#not need now
'''
while not subnet:
        try:
                subnet = raw_input('Give a correct ip/sider or ip-range or http to run the scan\n')
                subnet = IPNetwork(subnet)
        except Exception as e:
                if '-' in subnet:
                        ipall =[]
                        subnet = subnet.split('-')
                        ipfirst = subnet[0]
                        ips = ipfirst.split('.')
                        iptmp = ips[0]+'.'+ips[1]+'.'+ips[2]+'.'
                        for i in range(int(ips[3]),(int(subnet[1])+1)):
                                ip = iptmp+str(i)
                                ipall.append(ip)
                        subnet =ipall
                elif subnet[0].isalpha():
                        subnethost = socket.gethostbyname(arg)
                        subnet = IPNetwork(subnethost)
                else:
                        print 'bad ip or url , for help (portscanner.py -h)'
'''

# Checking reply address from ip
def isAlive(ip):
        ppkt = IP(dst=ip,ttl=64)/ICMP()
        # this will recieve the answer (if...)
        reply = sr1(ppkt,timeout=TIMEOUT)
        # check if answered
        if not (reply is None):
                print reply[IP].src, '[+] is online'
                liveIPs.append(reply[IP].src)
        else:
                print ip,'[-] timed out'

        return liveIPs

#Checking reply port, if it gives banner
def getBanner(ip,portlist):
        for port in portlist:
                try:
                        s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
                        result = s.connect_ex((ip,port))
                        if result ==0:
                                ans = s.recv(1024)
                                print "Port {}:   Open".format(port)
```

```
                                print 'found ->','ip:', ip, 'port:',port, 'banner:', str(ans)
                                answ ='ip:', ip, 'port:',port, 'banner:', str(ans)
                                answers.append(answ)
                        s.close()
                except Exception, e:
                        print 'bad port',port,', please try again...'
                        pass
                        s.close()

# main func
if __name__ == '__main__':
        # Check the time at started
        t1 = datetime.now()
        #Loop ip-live from the address entered
        for ip in subnet:
                alive = isAlive(str(ip))
        #Loop banner from addresses are available
        for ip in liveIPs:
                banners = getBanner(ip,portlist)
        # Saves the file
        save = open(output,"w")
        save.write(str(answers))
        save.close()

        # Checking the time again
        t2 = datetime.now()
        # Calculates the difference of time
        total =  t2 - t1
        print 'scanning completed in:',total
```