PROJETO DE BLOCO

ALUNO: MARCELO DA SILVA OLIVEIRA

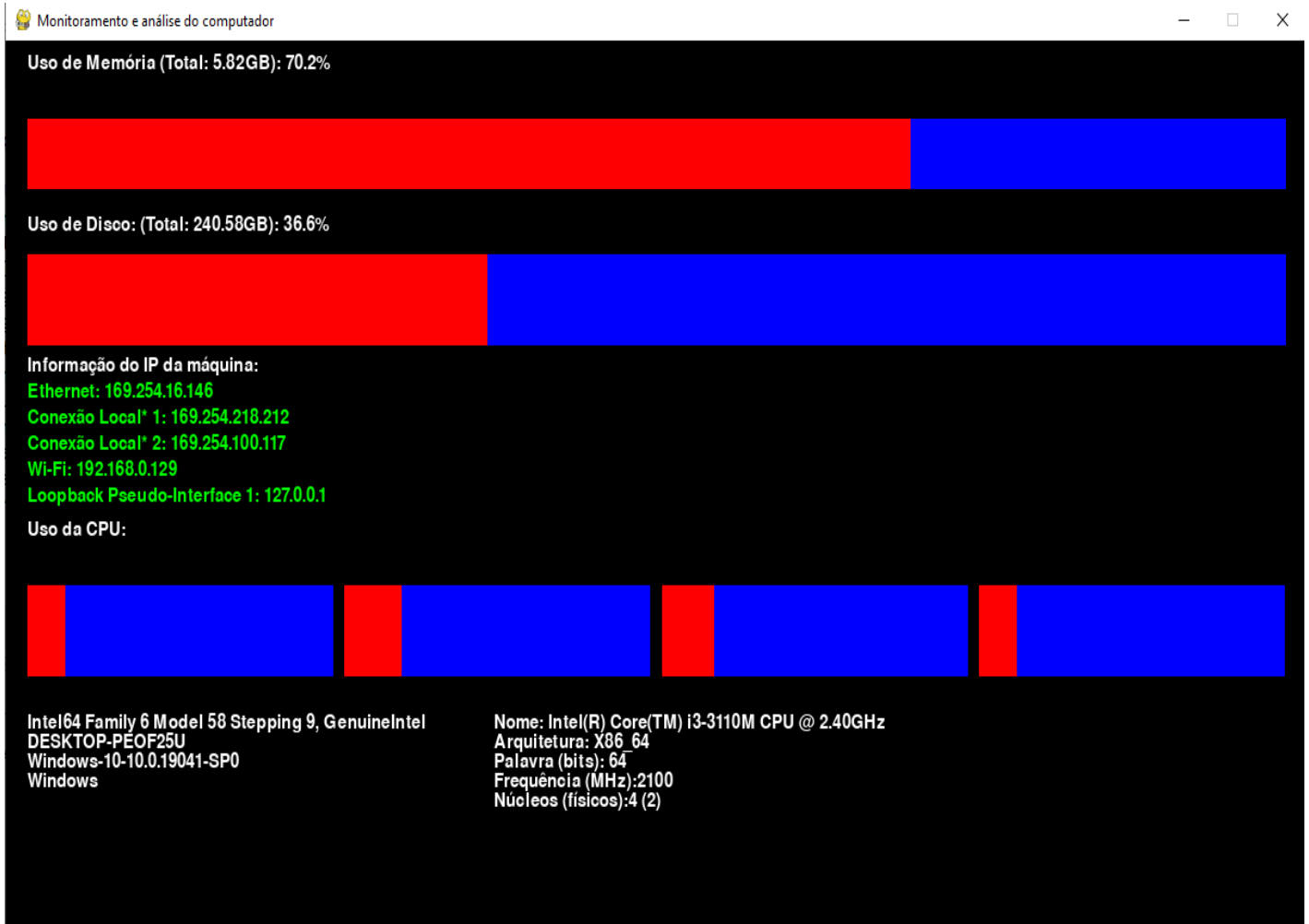
PROFESSOR: ALCIONE DOLAVALE

SOBRE O PROJETO

O projeto consiste num aplicativo simples de apresentação textual do monitoramento e análise de computadores em rede. Ele foi desenvolvido em linguagem Python usando módulos como psutil (para capturar dados do sistema computacional) e sockets (para criar cliente e servidor) e foi desenvolvido de forma incremental durante o curso.

ETAPAS 1, 2 E 3

- ***Apresentação gráfica do monitoramento e análise do computador usando módulos como psutil (para capturar dados do sistema computacional) e Pygame (para exibir graficamente os dados).***



ETAPA 4

- ***Nesta etapa foi utilizado o módulo 'os' para fazer o monitoramento de arquivos e diretórios e processos em python***

```python
import os, time
# Obtém lista de arquivos e diretórios do diretório corrente:
lista = os.listdir()
dic = {} # cria dicionário
for i in lista: # Varia na lista dos arquivos e diretórios
    if os.path.isfile(i): # checa se é um arquivo
        # Cria uma lista para cada arquivo. Esta lista contém o
        # tamanho, data de criação e data de modificacão.
        dic[i] = []
        dic[i].append(os.stat(i).st_size) # Tamanho
        dic[i].append(os.stat(i).st_atime) # Tempo de criação
        dic[i].append(os.stat(i).st_mtime) # Tempo de modificação

titulo = '{:11}'.format("Tamanho") # 10 caracteres + 1 de espaço
# Concatenar com 25 caracteres + 2 de espaços
titulo = titulo + '{:27}'.format("Data de Modificação")
# Concatenar com 25 caracteres + 2 de espaços
titulo = titulo + '{:27}'.format("Data de Criação")
titulo = titulo + "Nome"
print(titulo)
for i in dic:
    kb = dic[i][0]/1024
    tamanho = '{:10}'.format(str('{:.2f}'.format(kb)+' KB'))
    print(tamanho, str(time.ctime(dic[i][2])), " ", str(time.ctime(dic[i][1])),"  ", str(i))
```

```python
import subprocess, psutil, time
lista = psutil.pids()
lista_proc = []
titulo = '{:^7}'.format("PID")
titulo = titulo + '{:^11}'.format("# THREADS")
titulo = titulo + '{:^26}'.format("CRIAÇÃO")
titulo = titulo + '{:^9}'.format("T. USU.")
titulo = titulo + '{:^9}'.format("T. SIS.")
titulo = titulo + '{:^12}'.format("MEM. (%)")
titulo = titulo + '{:^12}'.format("RSS")
titulo = titulo + '{:^12}'.format("VMS")
titulo = titulo + " EXECUTÁVEL"
lista_proc.append(titulo)
def mostra_info(pid):
    try:
        p = psutil.Process(pid)
        texto = '{:6}'.format(pid)
        texto = texto + '{:11}'.format(p.num_threads())
        texto = texto + "  " + time.ctime(p.create_time()) + "  "
        texto = texto + '{:8.2f}'.format(p.cpu_times().user)
        texto = texto + '{:8.2f}'.format(p.cpu_times().system)
        texto = texto + '{:10.2f}'.format(p.memory_percent()) + " %"
        rss = p.memory_info().rss/1024/1024
        texto = texto + '{:10.2f}'.format(rss) + " MB"
        vms = p.memory_info().vms/1024/1024
        texto = texto + '{:10.2f}'.format(vms) + " MB"
        texto = texto + " " + p.exe()
        lista_proc.append(texto)
    except:
        pass
for i in lista:
    mostra_info(i)
for i in range(len(lista_proc)):
    print(lista_proc[i])
```

```
Tamanho    Data de Modificação        Data de Criação            Nome
1.90 KB    Thu Sep 23 07:15:32 2021   Fri Nov 12 12:07:08 2021   Botão.py
37.35 KB   Thu Sep 23 07:21:42 2021   Tue Nov 16 23:07:41 2021   Capa TP_AT - 2021.docx
0.02 KB    Fri Sep 10 07:51:43 2021   Fri Nov 12 12:07:08 2021   conta.py
0.34 KB    Fri Aug 13 08:11:05 2021   Fri Nov 12 12:07:08 2021   CPU INFO.py
1.89 KB    Fri Sep 10 11:50:57 2021   Fri Nov 12 12:07:08 2021   CPU pygame.py
0.19 KB    Fri Jul 23 10:55:13 2021   Fri Nov 12 12:07:08 2021   CUMSUM.py
0.40 KB    Fri Nov 12 11:26:42 2021   Fri Nov 12 12:07:08 2021   Etapa 4 Ex 3 PB.py
0.08 KB    Thu Aug 26 12:11:06 2021   Fri Nov 12 12:07:08 2021   IP.py
3.36 KB    Tue Aug 17 09:49:15 2021   Fri Nov 12 12:07:08 2021   Jogo do quadradinho.py
239.70 KB  Mon Sep 20 07:11:12 2021   Fri Nov 12 12:07:08 2021   Marcelo_Oliveira_DR1_AT.pdf
0.65 KB    Thu Aug 26 11:33:49 2021   Fri Nov 12 12:07:08 2021   Memória - 1.py
0.61 KB    Fri Nov 12 11:40:30 2021   Mon Nov 29 17:29:11 2021   OS Estapa 4 Ex 2 PB.py
0.11 KB    Thu Sep  9 09:00:10 2021   Fri Nov 12 12:07:08 2021   pandas.py
1.05 KB    Thu Sep  9 19:51:40 2021   Fri Nov 12 12:07:08 2021   q14.py
1.42 KB    Thu Sep  9 20:02:51 2021   Fri Nov 12 12:07:08 2021   Questão 13.py
1.17 KB    Thu Sep  9 20:28:25 2021   Fri Nov 12 12:07:08 2021   Questão 14.py
0.38 KB    Fri Jul 23 10:40:23 2021   Fri Nov 12 12:07:08 2021   Rasc.py
0.04 KB    Fri Nov 12 11:36:37 2021   Mon Nov 29 17:29:11 2021   rasc2.py
1.90 KB    Fri Aug 13 09:02:55 2021   Fri Nov 12 12:07:08 2021   Simple gui.py
0.02 KB    Fri Nov  5 08:40:49 2021   Fri Nov 12 12:07:08 2021   test5.py
0.05 KB    Wed Sep  1 19:05:15 2021   Fri Nov 12 12:07:08 2021   teste.py
0.71 KB    Thu Sep  9 20:39:16 2021   Fri Nov 12 12:07:08 2021   teste3.py
0.49 KB    Tue Oct  5 08:59:53 2021   Fri Nov 12 12:07:08 2021   teste4.py
4.45 KB    Thu Aug 26 23:04:39 2021   Fri Nov 12 12:07:08 2021   TP2.py
5.51 KB    Thu Nov  4 08:16:00 2021   Wed Dec  8 22:57:40 2021   TP3 PB.py
2.09 KB    Fri Sep 10 10:06:05 2021   Fri Nov 12 12:07:08 2021   TP3 Questão 10.py
2.15 KB    Fri Sep  3 11:17:37 2021   Fri Nov 12 12:07:09 2021   TP3 Questão 11.py
2.44 KB    Fri Sep 10 10:06:10 2021   Fri Nov 12 12:07:09 2021   TP3 Questão 12.py
0.84 KB    Fri Sep  3 10:54:05 2021   Fri Nov 12 12:07:09 2021   TP3 Questão 9.py
25.15 KB   Thu Sep  9 20:54:05 2021   Fri Nov 12 12:07:09 2021   __rzi_0.514
```

```
PID  # THREADS        CRIAÇÃO          T. USU.  T. SIS.  MEM. (%)     RSS        VMS      EXECUTÁVEL
4      157 Wed Dec 31 21:00:00 1969    0.00  2076.08    0.45 %    27.12 MB    0.25 MB
100      4 Mon Dec  6 13:06:27 2021    0.00     6.77    0.04 %    38.13 MB    6.48 MB Registry
380      5 Mon Dec  6 13:07:33 2021    0.27     0.56    0.04 %     2.49 MB    1.68 MB C:\Windows\System32\fontdrvhost.exe
388      5 Mon Dec  6 13:07:33 2021    2.19     2.17    0.11 %     6.75 MB    5.59 MB C:\Windows\System32\fontdrvhost.exe
500      2 Mon Dec  6 13:06:48 2021    0.00     0.20    0.02 %     1.04 MB    1.11 MB C:\Windows\System32\smss.exe
516      7 Wed Dec  8 22:57:25 2021    0.28     0.58    0.25 %    14.83 MB    4.96 MB C:\Windows\System32\svchost.exe
620    239 Mon Dec  6 13:08:06 2021  389.61  1019.58    2.65 %   158.02 MB  215.17 MB C:\Windows\explorer.exe
636     11 Mon Dec  6 13:07:28 2021    1.53     8.16    0.09 %     5.15 MB    2.42 MB C:\Windows\System32\csrss.exe
728     12 Mon Dec  6 13:07:33 2021  120.91   121.12    0.33 %    19.95 MB   13.48 MB C:\Windows\System32\svchost.exe
732      1 Mon Dec  6 13:07:29 2021    0.00     0.09    0.10 %     5.99 MB    1.34 MB C:\Windows\System32\wininit.exe
748     15 Mon Dec  6 13:07:29 2021    2.38   253.59    0.09 %     5.61 MB    2.72 MB C:\Windows\System32\csrss.exe
888      3 Mon Dec  6 13:07:30 2021    0.38     1.47    0.16 %     9.63 MB    2.57 MB C:\Windows\System32\winlogon.exe
880      6 Mon Dec  6 13:07:33 2021    5.55     8.91    0.13 %     7.73 MB    2.79 MB C:\Windows\System32\svchost.exe
884      7 Mon Dec  6 13:07:30 2021   17.77    36.48    0.14 %     8.51 MB    5.11 MB C:\Windows\System32\services.exe
892      8 Mon Dec  6 13:07:31 2021   26.28    24.05    0.36 %    21.47 MB    9.42 MB C:\Windows\System32\lsass.exe
904      1 Thu Dec  9 01:19:34 2021    0.02     0.00    0.03 %     2.01 MB    0.88 MB C:\Windows\System32\vrcrleakdiag.exe
928     14 Mon Dec  6 13:07:51 2021   16.77    48.48    0.66 %    39.45 MB   10.77 MB C:\Windows\System32\sihost.exe
1016    14 Mon Dec  6 13:07:33 2021   42.39    81.42    0.61 %    36.35 MB   16.99 MB C:\Windows\System32\svchost.exe
1060     5 Mon Dec  6 13:07:34 2021    0.12     0.34    0.12 %     6.92 MB    1.91 MB C:\Windows\System32\svchost.exe
```

ETAPA 5

- ***Nesta etapa, foi utilizado o módulo 'sched' para fazer o monitoramento e o gerenciamento de processos.***

```python
def Long_event_1(name):
    lista_dir = []
    p_dir = ''
    entrada = input()
    if os.path.isdir(entrada):
        lista_dir.append(entrada)
        somador = 0
        while lista_dir:
            diretorio = lista_dir[0]
            p_dir = os.path.join(p_dir, diretorio)
            lista = os.listdir(p_dir)
            for i in lista:
                p = os.path.join(p_dir, i)
                if os.path.isdir(p):
                    lista_dir.append(i)
                elif os.path.isfile(p):
                    somador = somador + os.stat(p).st_size
            lista_dir.remove(diretorio)
        print('tamanho: ' + str(round(somador/1024, 2)) + ' KB')
    else:
        print("O diretório", '\''+ entrada + '\'', 'não existe.')
    print ('FIM DO EVENTO:', time.ctime(), time.process_time())
    print()
import sched, time, os, psutil, subprocess

scheduler = sched.scheduler(time.time, time.sleep)

def Long_event_1(name): ···

def Long_event_2(name): ···

print('INICIO:', time.ctime())
scheduler.enter(2, 3, Long_event_1, ('LISTA DE INFORMAÇÕES DE ARQUIVOS E DIRETÓRIOS',))
scheduler.enter(8, 2, Long_event_2, ('LISTA DE INFORAÇÕES DE UM PROCESSO',))
print('CHAMADAS ESCALONADAS DA FUNÇÃO:', time.ctime(), time.process_time())

scheduler.run()
```

```python
def Long_event_2(name):
    print ('EVENTO:', time.ctime(), time.process_time(), name)
    pid = subprocess.Popen('notepad.exe').pid

    p = psutil.Process(pid)
    print('Nome: ', p.name())
    print('Executável: ', p.exe())
    print('Tempo de criação: ', time.ctime(p.create_time()))
    print('Tempo de usuário: ', p.cpu_times().user, 's')
    print('Tempo de sistema: ', p.cpu_times().system, 's')
    print('Percentual de uso de CPU:: ', p.cpu_percent(interval = 1.0), '%')
    perc_mem = '{:.2f}'.format(p.memory_percent())
    print('Percentual de uso de memória: ', perc_mem, '%')
    mem = '{:.2f}'.format(p.memory_info().rss/1024/1024)
    print('Uso de memória: ', mem, 'MB')
    print('Número de threads: ', p.num_threads())
    print ('FIM DO EVENTO:', time.ctime(), time.process_time())
    print()
```

```
INICIO: Thu Dec  9 02:45:19 2021
CHAMADAS ESCALONADAS DA FUNÇÃO: Thu Dec  9 02:45:19 2021 1.078125
C:\Users\win10\Thonny
tamanho: 377.84 KB
FIM DO EVENTO: Thu Dec  9 02:45:26 2021 1.109375

EVENTO: Thu Dec  9 02:45:27 2021 1.109375 LISTA DE INFORAÇÕES DE UM PROCESSO
Nome:  notepad.exe
Executável:  C:\Windows\System32\notepad.exe
Tempo de criação:  Thu Dec  9 02:45:27 2021
Tempo de usuário:  0.0 s
Tempo de sistema:  0.015625 s
Percentual de uso de CPU::  12.3 %
Percentual de uso de memória:  0.24 %
Uso de memória:  14.09 MB
Número de threads:  7
FIM DO EVENTO: Thu Dec  9 02:45:28 2021 1.125
```

ETAPA 6

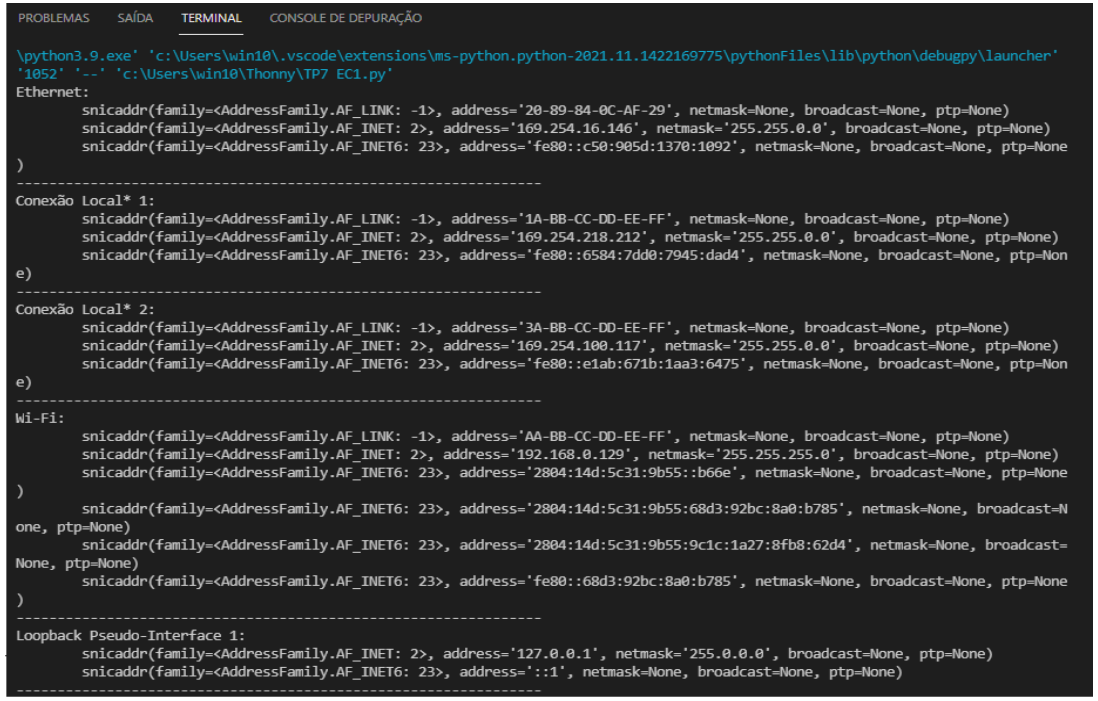
- ***Nesta etapa, foi utilizado o módulo 'nmap' para captar informações sobre redes e sub-redes***
- ***Estrturas de código na Etapa 8***

ETAPA 7

- ***Nesta etapa, foi utilizado as funções do módulo 'psutil' de Python para capturar informações das interfaces de redes do computador.***

```python
import psutil
import time
interfaces = psutil.net_if_addrs()
nomes = []
# Obtém os nomes das interfaces primeiro
for i in interfaces:
    nomes.append(str(i))
# Depois, imprimir os valores:
for i in nomes:
    print(i+":")
    for j in interfaces[i]:
        print("\t"+str(j))
    print('--------------------------------------------------------------------')

```



ETAPA 8

- ***Nesta etapa, foi utilizado a função 'socket' para desenvolver serviços de rede para apresentar informações de um computador remoto.***

SERVIDOR

```
# Servidor
import socket, psutil, platform, os, time, subprocess, sched, nmap, pickle

# Cria o socket
socket_servidor = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# Obtem o nome da máquina
host = socket.gethostname()
porta = 9999
# Associa a porta
socket_servidor.bind((host, porta))
# Escutando...
socket_servidor.listen()
print("Servidor de nome", host, "esperando conexão na porta", porta)
# Aceita alguma conexão
(socket_cliente,addr) = socket_servidor.accept()
print("Conectado a:", str(addr))

while True:
    msg = socket_cliente.recv(100000)
    # Decodifica mensagem em UTF-8:

def retorna_codigo_ping(hostname):
    """Usa o utilitario ping do sistema operacional para encontrar   o host

    plataforma = platform.system()
    args = []
    if plataforma == "Windows":
        args = ["ping", "-n", "1", "-l", "1", "-w", "100", hostname]

    else:
        args = ['ping', '-c', '1', '-W', '1', hostname]

    ret_cod = subprocess.call(args,
                              stdout=open(os.devnull, 'w'),
                              stderr=open(os.devnull, 'w'))
    return ret_cod

# Chamadas
lis = []
ip_string = msg.decode('utf-8')
ip_lista = ip_string.split('.')
base_ip = ".".join(ip_lista[0:3]) + '.'
print("O teste será feito na sub rede: ", base_ip)
host_validos = verifica_hosts(base_ip)
print ("Os host válidos são: ", host_validos)

print('Iniciando nmap.PortScanner')
obter_hostnames(host_validos)
print()
print('FIM.')
bytes = pickle.dumps(lis)
socket_cliente.send(bytes)
elif '4' == msg.decode('utf-8'):
    msg = 'Ok... Entre com um IP alvo(o processo pode levar alguns minutos): '
    socket_cliente.send(msg.encode('utf-8'))
    msg = socket_cliente.recv(100000)
    def obter_hostnames(host_validos):
        resposta = "O teste foi feito na sub rede: " + base_ip + '\n'
        resposta = resposta + "Os hosts válidos são: " + str(host_validos) + '\n'
        nm = nmap.PortScanner()
        for i in host_validos:
            try:
                nm.scan(i)
                resposta = resposta + '\n' + 'O IP possui o nome ' + str(nm[i].hostname()) + '\n'

                for proto in nm[i].all_protocols():
                    resposta = resposta + '----------' + '\n'
                    resposta = resposta + 'Protocolo : ' + proto + '\n'

                    lport = nm[i][proto].keys()

                    for port in lport:
                        resposta = resposta + 'Porta: ' + str(port) + '     ' + 'Estado: '  +  str(nm[i][prot
                msg = 'O endereço IP' + str(i) + 'foi verificado'
                print(msg)
                print(resposta)
            except:
                resposta = resposta + 'O IP deu problema.'
                pass

            lis.append(resposta)

def verifica_hosts(base_ip):
    """Verifica todos os host com a base_ip entre 1 e 255 retorna uma lista com todos os host que t
    print("Mapeando\r")
    host_validos = []
    return_codes = dict()
    for i in range(1, 255):

        return_codes[base_ip + '{0}'.format(i)] =   retorna_codigo_ping(base_ip + '{0}'.format(i))
        if i %20 ==0:
            print(".", end = "")
        if       return_codes[base_ip + '{0}'.format(i)] == 0:
            host_validos.append(base_ip + '{0}'.format(i))
    print("\nMapping ready...")

    return host_validos
```

CLIENTE

```
# Cliente
import socket, sys, pickle
# Cria o socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
try:
    # Tenta se conectar ao servidor
    s.connect(("192.168.0.129", 9999))
except Exception as erro:
    print(str(erro))
    sys.exit(1) # Termina o programa
msg = input("Menu: \n [1] Informações de arquivos e dir
                    # Envia mensagem codificada em byte
s.send(msg.encode('utf-8'))
while msg != '9':
    if msg == '4':
        bytes = s.recv(100000)
        if 'Ok... Entre com um IP alvo(o processo pode levar alguns minutos): ' == bytes.decode('utf-8'):
            print(bytes)
            msg = input()
            s.send(msg.encode('utf-8'))
            bytes = s.recv(100000)
            lista_proc = pickle.loads(bytes)
            for i in range(len(lista_proc)):
                print(lista_proc[i])
            msg = input("Menu: \n [1] Informações de arquivos e diretórios \n [2] Informações de processo
            # Envia mensagem codificada em bytes ao servidor
            s.send(msg.encode('utf-8'))
```

RESRPOSTA DO SERVIDOR AO CLIENTE

```
Menu:
 [1] Informações de arquivos e diretórios
 [2] Informações de processo
 [3] Escalonamento de Chamadas
 [4] Redes e Subredes
 [5] Informação de interfaces de rede
 [9] Sair
4
b'Ok... Entre com um IP alvo(o processo pode levar alguns minutos): '
192.168.0.129
O teste foi feito na sub rede: 192.168.0.
Os hosts válidos são: ['192.168.0.1', '192.168.0.31', '192.168.0.129', '192.168.0.130']

O IP possui o nome cable.box
----------
Protocolo : tcp
Porta: 22    Estado: filtered
Porta: 23    Estado: filtered
Porta: 53    Estado: open
Porta: 80    Estado: open
Porta: 111    Estado: filtered
Porta: 443    Estado: open
Porta: 5000    Estado: open

O IP possui o nome
----------
Protocolo : tcp
Porta: 5555    Estado: open
Porta: 32769    Estado: open
```

```
O IP possui o nome DESKTOP-PEOF25U.box
----------
Protocolo : tcp
Porta: 25    Estado: filtered
Porta: 110    Estado: filtered
Porta: 119    Estado: filtered
Porta: 125    Estado: filtered
Porta: 135    Estado: open
Porta: 139    Estado: open
Porta: 143    Estado: filtered
Porta: 445    Estado: open
Porta: 465    Estado: filtered
Porta: 548    Estado: filtered
Porta: 563    Estado: filtered
Porta: 587    Estado: filtered
Porta: 800    Estado: filtered
Porta: 903    Estado: filtered
Porta: 993    Estado: filtered
Porta: 995    Estado: filtered
Porta: 1025    Estado: filtered
Porta: 1122    Estado: filtered
Porta: 1433    Estado: filtered
Porta: 1688    Estado: open
Porta: 5357    Estado: open
Porta: 9999    Estado: open
O IP deu problema.
```

MOSTRANDO NA PRÁTICA...

OBRIGADO