Aluno:
Thomaz Tavares Trize

Trabalho de Laboratório de Desenvolvimento de Redes de Computadores III, desenvolvimento de algoritmo para captura de inforamações sobre host em linguagem python

Trabalho de Desenvolvimento da disciplina Laboratório de Desenvolvimento III, do Curso Superior de Redes de Computadores da Fatec Bauru.

Professor Luis Alexandre Da Silva

BAURU/SP
2021

Resumo

O projeto apresenta com base o desenvolvimento em python um código que análisa os dispositivos ativos na rede e captura os nomes de hosts das maquinas positivas da rede. O programa é voltado a uma ferramenta para análise de dispositivos na rede e retorno dos ambiêntes disponives na rede, assim coletando os hosts livres e utilizados.

Sumário

O Programa:

Desenvolvido com a função de capturar dados de hosts e ip’s ativos na rede o programa em python tem como base algumas bibliotecas de integração em redes e sistemas operacionais, sua interface é de modo simples onde o único valor a ser inserido é o ip da rede em que o utilizador pretende análisar.

Observações:

1. O usúario deve estar conectado a rede que deseja fazer a coleta.
2. O programa faz a adaptação automaticamente de acordo com o sistema em que esta sendo executado Windows/Linux/MacOS.
3. Nomes de maquinas são exibidas de acordo com o definido no sistema, alguns exemplos:
    - Desktop-10934
    - user
    - Windows
    - Debian
    - pop-os

Como utilizar:

Linux:

1. Instale o python

```
sudo apt install python3
```

2. Salve o arquivo do código fonte com extensão .py
3. Execução:

```
python3 scan.py
```

Windows:

1. Instalação do python

link: https://www.python.org/downloads/

2. Executar o código por meio da IDE python

a) abra o aquivo salvo

b) execute o programa

Código Fonte:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import os
import socket
import platform
import subprocess

#############################
def verified_os(distro, plat):
    global op
    if distro == "windows":
        op = subprocess.run(["ping", "-c", "1", plat], stdout=subprocess.PIPE)
    elif distro == "linux":
        op = os.system("ping -c 1 -w 1 " + plat + ">/dev/null")
    else:
        op = os.system("ping -c 1 -w 1 " + plat + ">/dev/null")
#############################

print("|----------------------------|")
print("|   Software developer por 3T  |")
print("|       NETWORK Scanner        |")
print("|----------------------------|\n\n")

print("*****")
print("Hello " + socket.gethostname())
print("*****\n\n")

distro = platform.system().lower()
print("Your distro system is: " + distro)

ip = input("Insert the IP address: ")
print("\n\n------------\n\n")
ipstr = ip.split(".")
p = "."
ipaddr = ipstr[0] + p + ipstr[1] + p + ipstr[2] + p

################################
for net in range(1,255):
    machine = ipaddr + str(net)

    verified_os(distro, machine)

    if op == 0:
        print(machine + " is up")
        try:
            hostname = socket.gethostbyaddr(machine)
            print(hostname)
            print("\n------------\n")
        except:
            hostname = socket.gethostbyname_ex(machine)
```

```
            print(hostname)
            print("\n----------\n")
    else:
        print(machine + " down\n")

print("Program END...")
##############################
```

Exemplo de saída:

```
|-----------------------------------------|
|   Software developer por 3T  |
|        NETWORK Scanner              |
|-----------------------------------------|


*****
Hello zano <- nome do utilizador
*****


Your distro system is: linux
Insert the IP address: 192.168.0.8 <- IP de maquina inserido


-------------


192.168.0.1 is up
('_gateway', [], ['192.168.0.1'])

------------

192.168.0.2 down

192.168.0.3 is up
('192.168.0.3', [], ['192.168.0.3'])

------------

192.168.0.4 is up
('192.168.0.4', [], ['192.168.0.4'])

------------

192.168.0.5 down

192.168.0.6 down

192.168.0.7 down

192.168.0.8 is up
('zano', [], ['192.168.0.8']) <- Maquina principal

------------

192.168.0.9 down

192.168.0.10 down
```

O valor de ip inserido foi 192.168.0.8, este é o endereço da maquina em ipv4, usando ipconfig (windows) e ifconfig (linux) é retornado o valor ipv4 da maquina na rede.

Código Comentado:

1. Bibliotecas

```
###############
import os
import socket
import platform
import subprocess
###############
```

A biblioteca “os” teve como função fazer a requisição do terminal do sistema para realização do comando ping e averiguar as maquinas ativas.

O “socket” é a biblioteca de função comunicativa entre a rede assim fazendo request aos dispositivos, foi usado para requisitar os hosts e ip’s das maquinas.

Platform é uma biblioteca de sistema onde podemos realizar coleta de valores da “plataforma”/sistema operacional da maquina assim sendo usado para portábilidade do código.

O Subprocess foi utilizado para a melhor performance em sistemas windows, por possuir uma integração com a command line do sistema, sendo assim facilitando a utilização do ping.

2.Função: Validação de Sistema

```
#############################
def verified_os(distro, plat):
    global op
    if distro == "windows":
        op = subprocess.run(["ping", "-c", "1", plat], stdout=subprocess.PIPE)
    elif distro == "linux":
        op = os.system("ping -c 1 -w 1 " + plat + ">/dev/null")
    else:
        op = os.system("ping -c 1 -w 1 " + plat + ">/dev/null")
#############################
```

A função “verified_os” recebe o tipo de sistema da distribuição e o ip da plataforma, efetuando uma condicional com o retorno global ao sistema, com as respostas da distribuição e o comando ping, sendo realiazado de acordo com a portábilidade do sistema operante. Utilizando a biblioteca “subprocess” para windows e “os” para linux/derivados e MacOS.

3.“Main”

```
##############################
print("|----------------------------|")
print("|   Software developer por 3T  |")
print("|       NETWORK Scanner        |")
print("|----------------------------|\n\n")
```

```
print("*****")
print("Hello " + socket.gethostname()) <- captura do nome da maquina
print("*****\n\n")

distro = platform.system().lower() <- coleta da plataforma do sistema.
print("Your distro system is: " + distro)

ip = input("Insert the IP address: ")
print("\n\n----------\n\n")
ipstr = ip.split(".")
p = "."
ipaddr = ipstr[0] + p + ipstr[1] + p + ipstr[2] + p
################################
```

Nesta estapa do código é impresso uma mensagem de introdução, onde a biblioteca "socket" utiliza a função "socket.gethostname()", está chamada de função retorna o nome da maquina de execução assim na linha a baixo temos a captura da plataforma para a portabilidade com a função "verified_os".

No primeiro input é requisitado o valor de endereço ip que será a faixa a ser análisada. Assim realizando a fragmentação das partes separando-as pelos "."(pontos) e as reagrupando em formatos de array, neste reagrupamento é realizado a conversão do input de formato string para um array, deixando neste array a última faixa/range ip ausente.

4.Verificação dos dispositivos

```
####################################
for net in range(1,255):
    machine = ipaddr + str(net)

    verified_os(distro, machine)

    if op == 0:
        print(machine + " is up")
        try:
            hostname = socket.gethostbyaddr(machine)
            print(hostname)
            print("\n----------\n")
        except:
            hostname = socket.gethostbyname_ex(machine)
            print(hostname)
            print("\n----------\n")
    else:
        print(machine + " down\n")

print("Program END...")
####################################
```

Na última etapa do processo é realizado um loop para a validação dos dispositivos ativos em rede, subistituindo a parte ausente do ip e realocando o valor do loop em um range de 1-255, portando fazendo uma ampla análise e locando os ativos e disponiveis em rede. Quando efetuado a troca de valor do range é acionado a função verified_os para realizar o ping na maquina alvo, caso o valor retorne ativo será imprimido os

resultados de “nome da maquina” e o “ip da maquina alvo”, na exceção de não ser realizado a conversão por ip ela será realizada pelo valor de host em rede. Sendo assim no sistema demonstrando em rede os hosts ativos e desativados das intefaces conectadas, em valor de falha será impresso o “ip da maquina alvo” e o marcador “down”, finalizando a varredura o programa é encerrado.