ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL



CONMUTACIÓN Y ENRUTAMIENTO

PARALELO 1

**Sistema de Configuración del protocolo de enrutamiento BGP para la interconexión entre ISP**

**MANUAL TECNICO**

DOCENTE:

Ing. Adriana Collaguazo

NÚMERO DE GRUPO:

4

INTEGRANTES:

Sophia Gómez

Alex Mendoza

Jamil Nagua

Andrés Vizueta

Nécker Espinosa

INTRODUCCION

Un sistema autónomo es la agrupación de redes IP que tienen una política de rutas propia e independiente. Esto significa que realiza su propia gestión del tráfico que fluye entre él y los diferentes Sistemas Autónomos que están en la red. Cada Sistema autónomo posee un número que lo identifica de manera única entre sus redes dentro de Internet, cada Sistema Autónomo es como una Internet en pequeño, ya que su rol se lleva a cabo por una sola entidad, típicamente un mismo Proveedor de Servicio de Internet (ISP)

Debido a la existencia del protocolo puerta de enlace de frontera1​ o BGP (del inglés Border Gateway Protocol), se logra intercambiar información de encaminamiento entre diferentes Sistemas Autónomos en este caso entre diferentes servidores, cada sistema autónomo posee su tabla de enrutamiento, pero gracias a BGP logran intercambiar dichas tablas, pero cabe recalcar que la comunicación se logra mediante los Routers externos o finales de cada sistema autónomo.

La forma de configurar y delimitar la información que contiene e intercambia el protocolo BGP es creando un sistema autónomo o AS. Cada uno tendrá conexiones o sesiones internas (iBGP), así como sesiones externas (eBGP).

Las relaciones que pueden existir entre distintos sistemas autónomos son principalmente de interconexión voluntaria (peering) y de tránsito. Básicamente una relación de tránsito es la que existe entre un proveedor y un cliente, debido a que el cliente pague por los recursos de Internet que le puede suministrar un proveedor. Las relaciones de peering no suelen ser pagadas y consisten en un enlace para comunicar dos sistemas autónomos con el fin de reducir costos, pérdida de paquetes y obtener caminos redundantes. Se recomienda hacer peering entre sistemas autónomos potencialmente similares en cuanto a tamaño.

Un escenario que se suele suceder es aquel llamado multiconexión, multi-ISP o multihoming. Estos nombres hacen referencia a un cliente que contrata a dos proveedores de tránsito, lo que implica que existen dos rutas de salida, de tal modo se deberá decidir entre un camino u otro dependiendo de ciertas especificaciones, necesidades o simples políticas que se impongan en el sistema autónomo.

Dentro de la comunicación BGP existen 4 tipos de mensajes que permiten la comunicación entre AS los cuales son:

OPEN: se utiliza para la inicialización de una sesión BGP una vez haya sido establecida la conexión TCP. Se suelen solicitar ciertos parámetros que caractericen a esa sesión. Por ejemplo, es muy probable que los miembros de la sesión no tengan la misma versión de BGP por lo que es importante indicar el número de versión en este mensaje.

UPDATE: es un mensaje de actualización, de gran importancia dentro de las operaciones de BGP ya que contiene los anuncios de nuevos prefijos. Se generarán mensajes de actualización cada vez que se determine una nueva mejor ruta para cierto destino o haya una modificación sobre alguna existente.

KEEPALIVE: una vez que la sesión BGP está activa se envía periódicamente un mensaje para mantener activa la conexión para confirmar que el otro extremo sigue estando activo en la sesión BGP. Generalmente se acuerda un tiempo máximo de espera durante el intercambio inicial de mensajes OPEN. El KEEPALIVE suele ser aproximadamente una vez cada tercio del tiempo de espera, pero no más de una vez cada segundo. Los mensajes KEEPALIVE no se deben generar si el tiempo de espera es cero ya que en ese caso se entiende que la sesión es completamente fiable.

NOTIFICATION: se envía a cerrar una sesión BGP y esto sucede cuando ocurre algún error que requiera el cierre de la misma. De modo que es un mensaje que permite informar nada.

OBJETIVOS

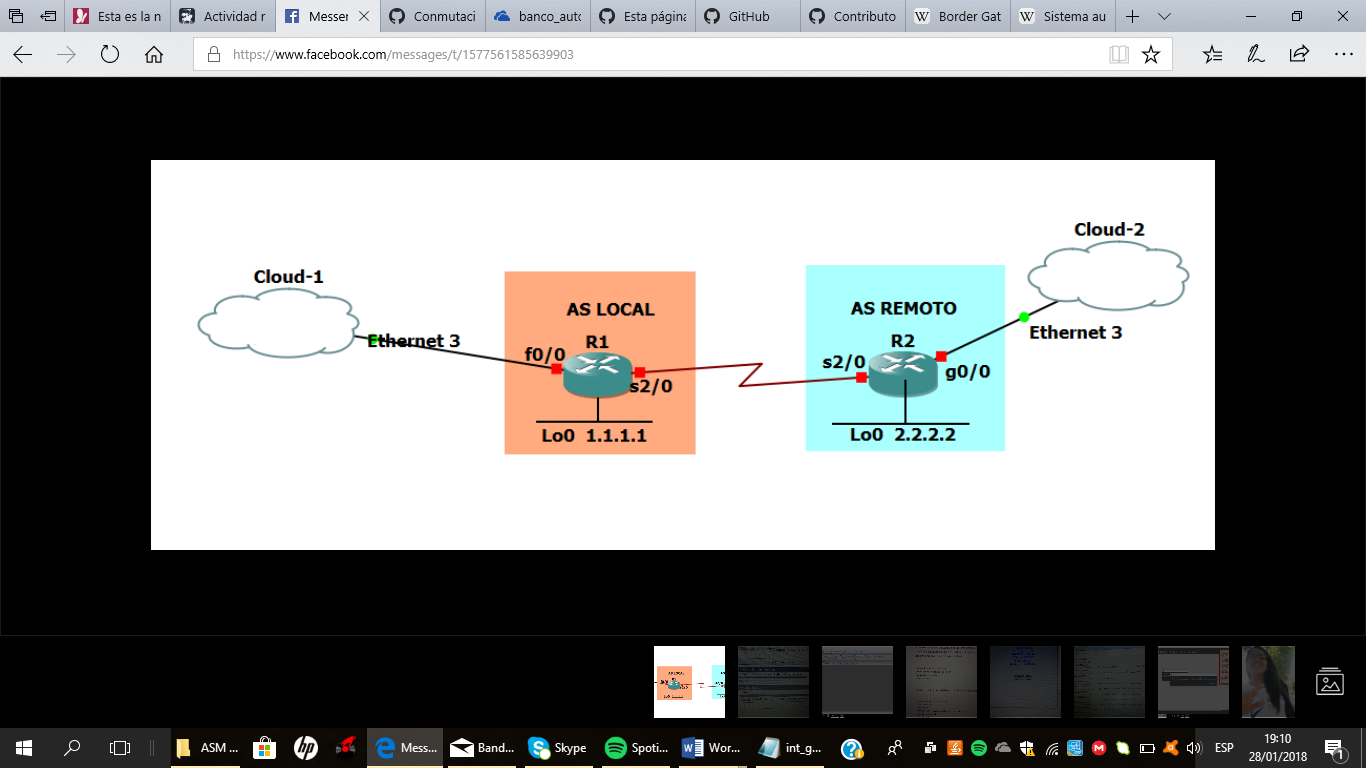
OBJETIVO GENERAL

Crear un sistema con el lenguaje de programación Python para lograr la conexión entre distintos proveedores de servicio de internet con diferentes sistemas autónomos, mediante el protocolo de enrutamiento BGP.

OBJETIVO ESPECIFICOS

* Realizar la configuración básica de un router directamente conectado con la aplicación creada.
* Lograr el intercambio de paquetes entre los routers de distintos sistemas autónomos, mediante el protocolo de enrutamiento de BGP.

Topología de la red



ESCENARIO DE RED

En este sistema se logrará la comunicación entre varios sistemas autónomos distintos, local y remoto, utilizando el protocolo BGP.

De acuerdo al diagrama de red se observa las diferentes nubes cloud-1 y cloud-2 las cuales son distintas computadoras que poseen el programa creado en Python, dichas computadoras se conectan a routers ( R1 y R2 ) que se encuentran vacíos, y para continuar el proceso se deben activar las interfaces de cada router y levantar la conexión telnet y el transporte de los protocolos con el comando line transport input.

Al ejecutar el programa este cargara toda la plantilla básica correspondiente a la asignación de nombre, dirección de red, mascara de red.

Alcance del sistema

El sistema fue creado con el propósito de conectar dos sistemas autónomos, un sistema autónomo local y otro remoto, para esto se trabaja con el protocolo de enrutamiento BGP.

Esta aplicación debe ser ejecutada en ambos routers que serán parte de la sesión bgp, tomando en cuenta, que el numero AS local es el AS perteneciente al router conectado a la PC donde se está ejecutando la aplicación, y el AS remoto es el AS perteneciente al router con el que se quiere establecer conexión (neighbor).

El sistema es útil para una red escalable. Pero debe ser ejecutado en cada router que quiere ser parte de la sesión BGP.

Además recordar, que se debe de identificar cada ip de los routers vecinos como peer BGP.

Tabla de direccionamiento

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dispositivo | Interfaz | Dirección IP | Mascara de subred | Puerta de enlace predeterminada |
| ISP1 | S2/0 | 209.168.14.1 | 255.255.255.252 | N/A |
|  | G0/0 | 192.168.137.10 | 255.255.255.0 | N/A |
|  | Lo0 | 1.1.1.1 | 255.255.255.255 | N/A |
| ISP2 | S2/0 | 209.168.14.2 | 255.255.255.252 | N/A |
|  | G0/0 | 192.168.137.2 | 255.255.255.0 | N/A |
|  | Lo0 | 2.2.2.2 | 255.255.255.255 | N/A |
| PC1 | Ethernet | 192.168.137.6 | 255.255.255.0 | 192.168.137.10 |
| PC2 | ethernet | 192.168.137.12 | 255.255.255.0 | 192.168.137.2 |

PC1: Equipo con la aplicación a ser ejecutada en el primer router. Se encuentra directamente conectada al router ISP1.

PC2: Equipo con la aplicación a ser ejecutada en el segundo router. Se encuentra directamente conectada al router ISP2.

CODIGO FUNCIONES

FUNCIONES TELNET

import getpass

import sys

import telnetlib

import time

**Pide usuario y contraseña para generar la conexión telnet con el router**

def conexion\_telnet(ip,user,password):

HOST = ip

# user = raw\_input("Enter your telnet username: ")

# password = raw\_input("Enter your telnet password: ")

# password = getpass.getpass()

tn = telnetlib.Telnet(HOST,port='23')

tn.read\_until("Username: ")

tn.write(user + "\n")

if password:

# tn.read\_until("Password: ")

tn.write(password + "\n")

print("Conexion telnet realizada con exito")

return tn

**Permite que se asigne la configuración básica en el router**

def configuracionBasica(tn, hostname):

# tn.write("enable \n")

# tn.write("admin\n")

tn.write("conf t \n")

tn.write("hostname "+ hostname+" \n")

tn.write("ip domain-name fiec.espol.edu.ec \n")

#tn.write("crypto key generate rsa \n")

#tn.write("1024 \n")

tn.write("username monitoreo privilege 5 secret monitoreo \n")

tn.write("banner motd # ACCESO SOLO A PERSONAL AUTORIZADO# \n")

tn.write("line vty 0 4 \n")

tn.write("transport input all \n")

tn.write("login local \n")

tn.write("exec-timeout 3 3 \n")

tn.write("logging synchronous \n")

tn.write("line console 0 \n")

tn.write("transport output all \n")

tn.write("login local \n")

tn.write("exec-timeout 3 3 \n")

tn.write("logging synchronous \n")

configurarInterfaces(tn)

tn.write("end \n")

# tn.write("wr \n")

print("configuracion realizada con exito!!!")

**Permite establecer la configuración de las interfaces dentro del router**

def configurarInterfaces(tn):

f = open("R2\_interfaces.txt", 'r')

a = f.readlines()

for linea in a:

lista = linea.split(',')

print(lista)

interfaz = lista[0]

ip = lista[1]

mascara = lista[2]

mascara2 = (mascara[0:(len(mascara) - 1)])

tn.write("interface " + interfaz + "\n")

tn.write("ip address " + ip + " " + mascara2 + "\n")

tn.write("no shutdown \n")

tn.write("exit \n")

f.close()

**Asigna la ip del vecino**

#def Defininir\_IPActual(ipNeighbor):

# ipSplit=ipNeighbor.split(".")

# cuartoOcteto=ipSplit[3]

# ipActual=""

# for i in range(3):

# ipActual=ipActual+ipSplit[i]+"."

# if(cuartoOcteto=="1"):

# ipActual=ipActual+"2"

# else:

# ipActual = ipActual + "1"

# print (ipActual)

# return ipActual

**Asigna el sistema autónomo local**

def SesionBGP(tn, ASlocal):

tn.write("configure terminal\n")

tn.write("router BGP " + ASlocal+"\n")

tn.write("end \n")

**Asigna las ip del sistema autónomo vecino**

def configurarVecino(ipNeighbor, ASlocal, ASremoto,tn):

tn.write("conf terminal \n")

tn.write("ip route 0.0.0.0 0.0.0.0 lo0 name to\_core\_isp \n")

tn.write("router bgp "+ASlocal+"\n")

tn.write("neighbor " +ipNeighbor+ " remote-as " +ASremoto+"\n")

tn.write("end \n")

def configurarNetwork(tn, network, ASlocal):

tn.write("conf terminal \n")

tn.write("router bgp " + ASlocal + "\n")

if(network=="0.0.0.0"):

tn.write("network "+ network+" \n")

else:

tn.write("network " + network + " mask 255.255.255.0" +"\n")

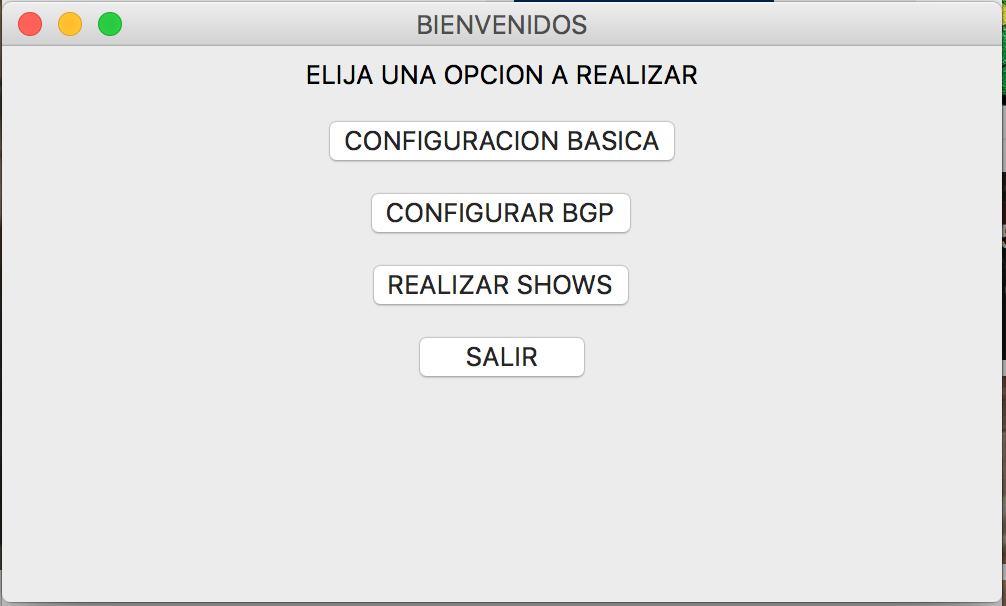
tn.write("end \n")

Interfaz gráfica:



Esta pantalla sirve para logear un usuario. Al darle click en el botón Validar, se revisa que los campos de usuario y password no estén vacíos.

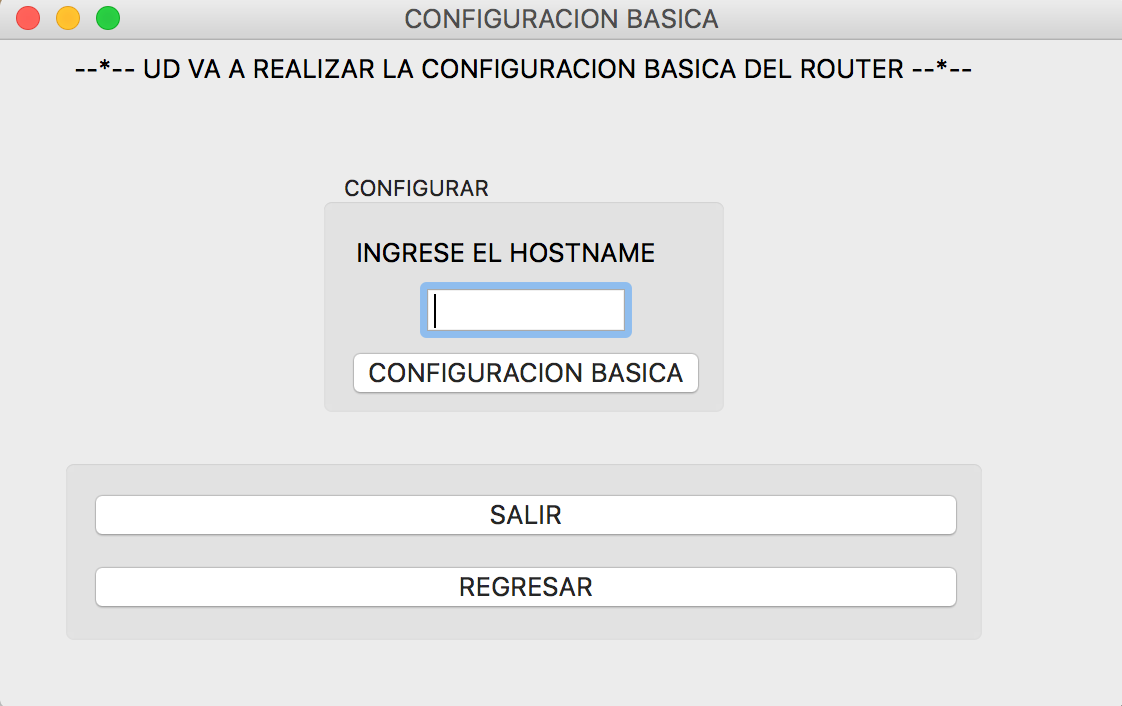
Después de esto, se procede a comparar en una base de datos donde se encuentran los usuarios con sus respectivas contraseñas. Si coincide, el usuario ingresa al sistema para hacer la conexión BGP. En caso de que no coincida con ninguno, no podrá ingresar al sistema y mostrará un mensaje de error.

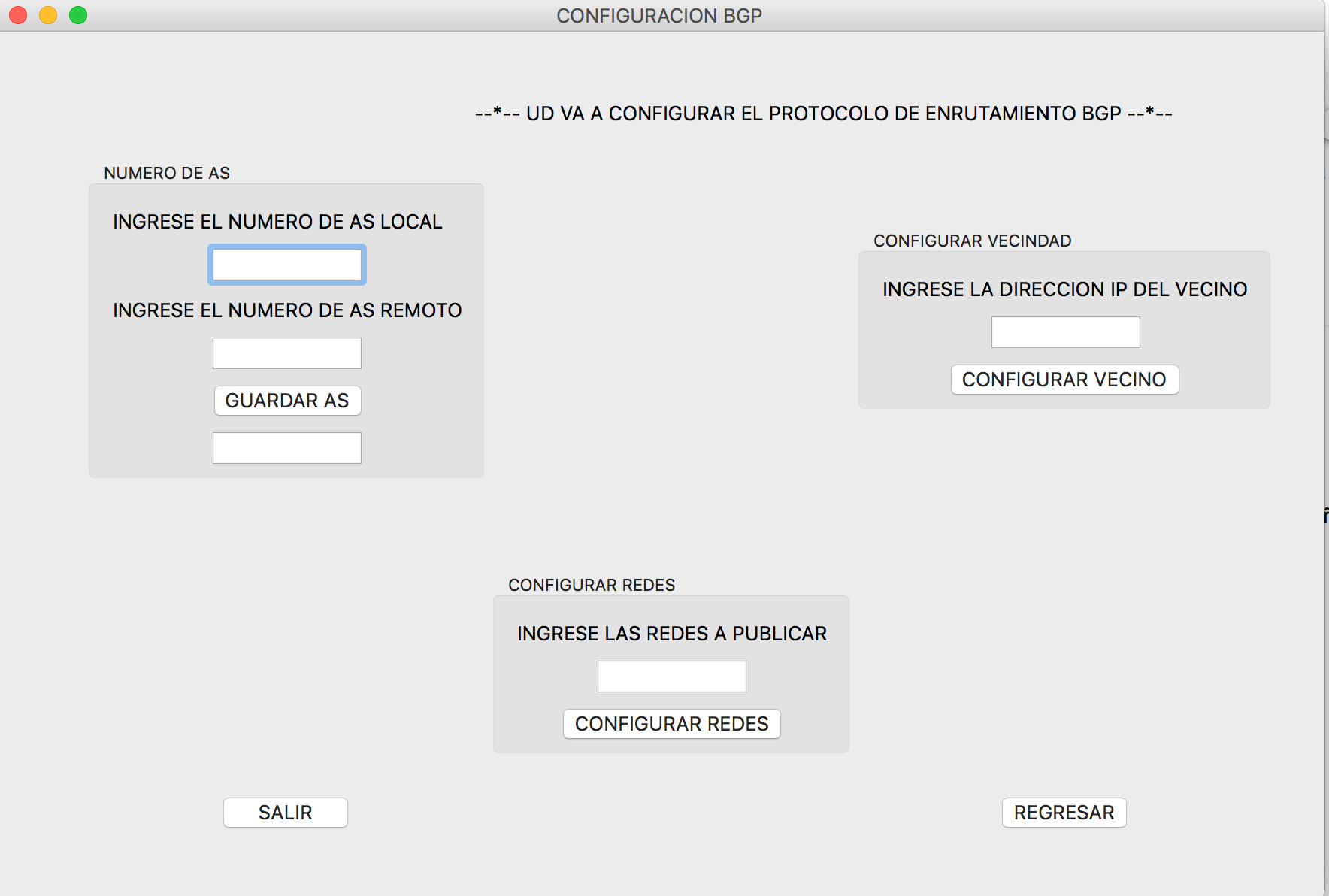
Al ingresar al sistema, muestra una pantalla de menú en la que, en caso de ser la primera vez que se conecta al router se deberá ir a configuración básica para configurar el router.

En caso de que el router este configurado, se procede a dar click en Configurar BGP para realizar la configuración del protocolo de enrutamiento.

Al dar click en el botón Realizar Shows aparecerá otra pantalla, en la que se podrá realizar diferentes shows relacionados a la sesión BGP.

Al dar click en Salir, se cierra la pantalla y el sistema completo.

Pantalla donde se configurará el router al dar click en Configuración Básica. Se procede a asignarle como hostname el nombre que se ingresa en el cuadro de texto. Además, se configura las interfaces con sus respectivas ip, esto lo hace al leer un archivo de texto con sus ips y mascara de subred respectivas.



Esta pantalla es la encargada de realizar la comunicación BGP. Primero se debe ingresar el número de AS local y remoto. En caso de que no se ingrese, mostrara un mensaje de error. Y ninguna de las otras opciones podrá ser ejecutada. Una vez que se hayan ingresado los sistemas autónomos, se procede a ingresar la ip del neighbor al dar click en Configurar Vecino.

Luego de esto, se debe ingresar las redes a publicar al dar click en configurar redes.

Las dos opciones anteriores están validadas para que no funcionen antes de ingresar los sistemas autónomos. Así mismo para que bote error si los campos se encuentran vacíos.

Una vez realizado todo esto, la configuración BGP en un router está completa. Pero para que haya comunicación entre los dos sistemas autónomos, el sistema debe ser ejecutado a su vez en el router vecino.

Recomendaciones

* Se recomienda que el computador tenga instalado el interpretador de Python, con sus respectivas librerías instaladas.
* Se recomienda que para cualquier modificación de la interfaz gráfica usar el programa wxformbuilder
* Se recomienda la utilización de routers de la misma marca (Recomendación cisco)

Conclusiones:

* Se concluye que la red es re escalable, es decir puede expandirse tanto como se lo desee, siempre y cuando se obtenga ordenadores que puedan interpretar la aplicación de Python
* Se concluye que la utilización del protocolo de enrutamiento BGP facilito la comunicación entre los sistemas autónomos establecidos.