Universidad del Valle de Guatemala
Facultad de Ingeniería
Departamento de Ciencias de la Computación
Redes

Laboratorio 3 y 4: algoritmos de enrutamiento

Link del repositorio: https://github.com/alegudiel/Lab3y4_Redes

Alejandra Gudiel 19232
María Isabel Montoya 19169
María José Morales 19145

Antecedentes

Conociendo a dónde enviar los mensajes para cualquier router se vuelve trivial el envío de mensajes.
Únicamente es necesario conocer el destino final y se reenvía al vecino que puede proveer la mejor ruta al destino. Toda esa información es almacenada en las tablas de enrutamiento. No obstante, con el dinamismo con el que se espera que pueda funcionar el Internet es necesario que dichas tablas puedan actualizarse y acomodarse a cambios en la infraestructura. Los algoritmos con los que se actualizan estas tablas son conocidos como algoritmos de enrutamiento.

- Flooding: envía cada paquete a todas las salidas posibles, excepto por la que entró. Mientas exista un mensaje, siempre se logrará enviar a un destino. Su mayor desventaja es que puede compartir mensajes previamente enviados.
- Distance Vector Routing: es iterativo, asíncrono y distribuido. Utiliza un algoritmo de enrutamiento en el que los routers envían periódicamente actualizaciones de enrutamiento a todos los vecinos mediante la difusión de sus tablas de enrutamiento completas
- Link State Routing: se basa en el algoritmo SPF (Shortest Path First). Calcula el mejor camino en base al menor costo acumulado a lo largo de una ruta. Cada nodo debe conocer los nodos adyacentes y su costo de vínculo para intercambiar paquetes de saludo y luego informar a los nodos vecinos sobre el estado de cada enlace directamente conectado. Cada nodo se encarga de propagar y mantener un DB de la información topológica y finalmente calcular el camino con costo más bajo hacia la red destino.

Objetivos

- Conocer los algoritmos de enrutamiento utilizados en las implementaciones actuales de Internet.
- Comprender cómo funcionan las tablas de enrutamiento.

Resultados

Protocolo usado:
- Emisor/nodo inicio: usuario @ dominio (alumchat.fun)
- Receptor/nodo final: usuario @ dominio (alumchat.fun)
- Nodos recorridos
- Distancia recorrida
- Listado de nodos: cuando se envía y recibe un mensaje.
- Mensaje enviado

```
C:\Users\Isabel\Lab3y4_Redes\src>python main.py
Please enter your username - user@alumchat.fun: marisamv@alumchat.fun
Please enter your password:
Please enter the algorithm to use:
1. Flooding
2. Distance Vector
3. Link State
1

        ------------------------------------------------------------
        |                                                          |
        |                                                          |
        |                    Welcome to alumchat                   |
        |                                                          |
        |                                                          |
        ------------------------------------------------------------
        1. Send a message
        2. How to use?
        3. Exit

Please select an option: 1
With whom do you want to start a chat? ale@alumchat.fun
Type your message: hola
Type your message: que tal
Type your message:
```

2|marisamv@alumchat.fun|ale@alumchat.fun|4||B|[('A', {'jid': 'ale@alumchat.fun'}), ('B', {'jid': 'marisamv@alumchat.fun'}), ('C', {'jid': 'majo@alumchat.fun'}), ('D', {'jid': 'test@alumchat.fun'})]-[('A', 'C', 1), ('B', 'A', 1), ('C', 'D', 1), ('D', 'B', 1)]

En este primer ejercicio, se ingresa la sesión y se le pide al usuario con qué algoritmo desea probar el chat. En este caso podemos ver que se usa *Flooding*. Por tanto, la imagen de abajo es la información que esta manejando el cliente para poder enviar el mensaje correspondiente. Se muestran los nodos que no han sido usados para llegar al destino final y las rutas hacia donde puede llegar con otros usuarios. Por tanto, como buscamos los caminos que puede tomar una conexión entre marisamv y ale, el algoritmo nos devuelve los que están disponibles para una futura ruta.

```
C:\Users\Isabel\Downloads\Lab3y4_Redes-main\Lab3y4_Redes-main\src>python main.py
Please enter your username - user@alumchat.fun: marisamv@alumchat.fun
Please enter your password:
Please enter the algorithm to use:
1. Flooding
2. Distance Vector
3. Link State
2

        ------------------------------------------------------------
        |                                                          |
        |                                                          |
        |                    Welcome to alumchat                   |
        |                                                          |
        |                                                          |
        ------------------------------------------------------------
        1. Send a message
        2. How to use?
        3. Exit

Please select an option: 1
With whom do you want to start a chat? ale@alumchat.fun
Type your message: Distance
Type your message:
```

En este segundo ejercicio hacemos uso del Distance Vector Routing donde cuenta la cantidad de saltos que debe realizar para poder llegar a nuestro destino final. En este caso, se nos muestra los caminos disponibles para llegar al nodo final y enviar el mensaje una vez haya considerado la mejor ruta..

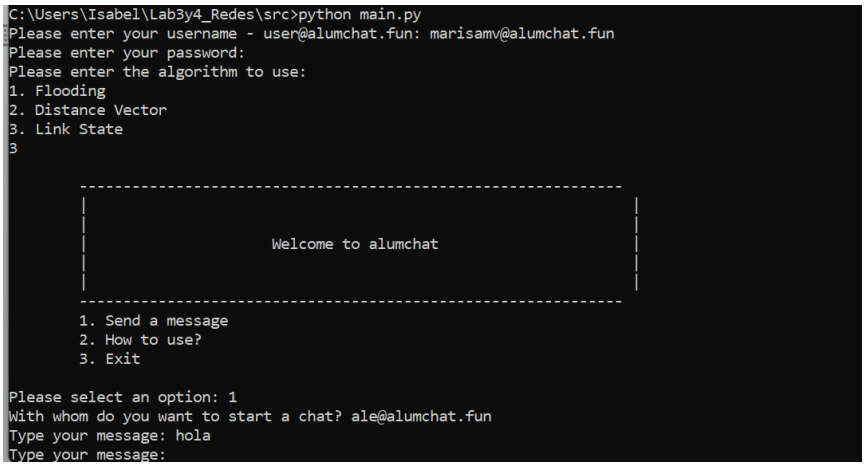
marisamv 16:41

2|marisamv@alumchat.fun|ale@alumchat.fun|4||B|[('A', {'jid': 'ale@alumchat.fun'}), ('B', {'jid': 'marisamv@alumchat.fun'}), ('C', {'jid': 'majo@alumchat.fun'}), ('D', {'jid': 'test@alumchat.fun'})]-[('A', 'C', 1), ('B', 'A', 1), ('C', 'D', 1), ('D', 'B', 1)]

3|marisamv@alumchat.fun|ale@alumchat.fun||1662072089.974315|A|

3|marisamv@alumchat.fun|ale@alumchat.fun||1662072094.984593|A|

--marisamv@alumchat.fun--ale@alumchat.fun--4||B--Distance



En este tercer ejercicio, se usa Link State Routing para poder chatear. Nuevamente, el usuario del lado derecho envía los mensajes al usuario con quien desea hablar y del lado derecho se muestran el estado inicial y final del vecino, que en este caso es con quien esta estableciendo la conversación; para así evitar enviar toda la información existente.

Discusión

En el presente laboratorio se creó un chat que sigue el protocolo definido de XMPP, en donde los usuarios se usaron con dominio de @alumchat.fun, los cuales para fines de aprendizaje acerca del enrutamiento de mensajes, fueron los que representan los nodos en las rutas a tomar en la comunicación del sistema. Se implementan las funcionalidades de poder recibir mensajes, actualizar un mensaje y hacer un echo, usando los algoritmo de enrutamiento de Distance Vector Routing y Link State Routing.

Se utilizó la librería networkx para la creación de tablas que logra que todos los nodos vecinos y no vecinos, para así encontrar la mejor ruta para enviar el mensaje al nodo destino. El protocolo definido se presenta en formato json (figura 1) para poder contener la información que se enviará a los nodos del grafo. Definiendo que el nodo recibirá los mensajes siguiendo el protocolo mostrado en la figura 2. Así pudiendo extraer el tipo de algoritmos a utilizar, el nodo destino, los vecinos, el tipo de mensaje, el delay, entre otros.

El hecho de que flooding sea un algoritmo que no usa el peso de las aristas para llegar a un nodo a otro (entre los otros dos algoritmos implementados), lo que hizo que haya sido el algoritmo menos complicado a desarrollar. Añadiendo que de todos modos implementar el LSR y DVR fue una etapa complicada cuando se realizó el laboratorio.

De las pruebas realizadas pudimos darnos cuenta que al no aplicar algoritmos para encontrar el camino más corto para moverse de un nodo a otro es el algoritmo que más recursos y tiempo ocupa. Encontrando que el LSR es el algoritmo más efectivo ya que en todo el proceso, los nodos conocen a toda la red completa y puede usar shortest path para encontrar la ruta más eficiente.

Conclusiones

- Sabemos que el DVR solamente conoce el siguiente paso y la tabla de distancias hacia un nodo específico, mientras que el LSR es el que conoce a toda la red completa. Lo que desde la parte de la teoría nos da un insight acerca de que este será el algoritmo más eficiente para el enrutamiento.
- Flooding es el algoritmo más fácil de entender pero el algoritmo más complicado de usar ya que será ineficiente.
- Con este laboratorio logramos entender las diferentes maneras de enrutamiento que pueden tomar los mensajes al momento de entablar una conversación. Así mismo podemos decir que comprendemos cómo funcionan las tablas de enrutamiento que son utilizadas para la mayoría de aplicaciones de mensajería.

Referencias

Computer Network | Distance Vector (2022) https://www.javatpoint.com/distance-vector-routing-algorithm#:~:text=The%20Distance%20vector%20algorithm%20is,result%20back%20to%20its%20neighbors.

Computer Netwrok | Link State Routing(2022 https://www.javatpoint.com/link-state-routing-algorithm

GeeksForGeeks (2022) https://www.geeksforgeeks.org/fixed-and-flooding-routing-algorithms/

Tanenbaum, A., & Wetherall, A. (2011). Redes De Computadoras (5ta ed.). Pearson Educación.