UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3067 Redes

Sección 10

Ing. Jorge Yass

Laboratorio 3 - Parte 2
Interconexión de nodos

SEBASTIAN ARISTONDO PEREZ 20880

JOSE DANIEL GONZALEZ CARRILLO 20293

GUATEMALA, 12 de Septiembre de 2023

Descripción de la práctica

Durante la presente práctica se realizó la implementación de tres algoritmos de enrutamiento los cuales fueron *Link State*, *Distance Vector* y *Flooding*. El objetivo principal fue implementarlos junto a un cliente de XMPP. Para esto se utilizó la librería *Slixmpp* de Python, la cual permitió realizar la conexión a un servidor y envío de mensajes. Distintas instancias del cliente simularon ser nodos de una topología de red. Esta topología se realizó para demostrar el funcionamiento de los algoritmos, de forma que se computaron las tablas de enrutamiento de cada nodo, dependiendo del algoritmo y se enviaron mensajes por una ruta dada por estas tablas. La implementación del cliente se hizo distinta para cada algoritmo.

En la implementación de Flooding, lo que se hizo fue solicitar el nombre del nodo y sus vecinos. Entonces al momento de mandar mensajes se definía el emisor, el receptor y cada nodo lo envió a todos sus vecinos. En el mensaje se agregó un header que guardó una lista de los nodos por los que ya había pasado el mensaje. De esta forma, se pudo identificar si el mensaje estaba rebotando. Una vez llegaba el mensaje al receptor, se mostraba y se dejaba de enviar a vecinos. Además, se usó un timestamp para determinar si un mensaje estaba repetido.

Para la implementación de Link State, primero se hizo un descubrimiento de la topología de red. Para esto se usó una etapa de “pre-flooding” en donde se envió un mensaje desde cada nodo indicando sus vecinos, a todos los nodos de la topología, por medio de sus vecinos. Posteriormente a esto, se determinó un cronómetro que esperaba cierto tiempo por más mensajes de topología. Cuando se acababa, realizaba el cálculo de su tabla de enrutamiento con el algoritmo descrito posteriormente y se podía empezar el envío de mensajes.

Finalmente, para la implementación de Distance Vector, fue necesaria una etapa del cálculo de las tablas de enrutamiento. Fue necesario determinar una manera para mandar las tablas, permitiendo que los demás nodos supieran que valor correspondía a qué nodo en la tabla del nodo orígen. Además, era necesario verificar que tablas atrasadas no sobreescribiendo tablas nuevas en el nodo destino. Para esto se usó un timestamp de los mensajes enviados. Se estableció un cronómetro que permitió determinar si el nodo ya no recibía mensajes durante cierto tiempo, de forma que se supiera cuando el sistema había convergido.

Descripción de los algoritmos

El algoritmo de "flooding" es un método utilizado en redes informáticas para enviar un mensaje o paquete de datos desde un origen a todos los nodos en una red. La característica principal de este algoritmo es que el mensaje se reenvía a todos los nodos vecinos sin ninguna consideración sobre si el nodo ya ha recibido el mensaje anteriormente. Esto asegura que el mensaje se propague por toda la red, pero puede dar lugar a problemas como bucles infinitos si no se gestionan correctamente.Para lograr esta mensajería se siguen los siguientes pasos: Nodo de origen decide enviar un mensaje, reenvío de mensajes a nodos vecinos, recepción y reenvío de mensajes.(Terrell Hanna, 2021)

El algoritmo de vector de distancia (Distance Vector) es un enfoque utilizado en el ámbito de las redes informáticas para determinar las rutas óptimas para la transferencia de datos entre nodos en una red. Este algoritmo se utiliza en protocolos de enrutamiento, que son conjuntos de reglas y procedimientos que permiten a los routers o conmutadores tomar decisiones sobre cómo dirigir el tráfico de red de manera eficiente. Para lograr esta coordinación es necesario seguir los siguientes pasos: Inicialización del nodo, intercambio de información, Actualización de distancias, Actualización de vecinos y convergencia y evitar bucles y recepción de mensajes. (Antoniou, 2016)

El algoritmo Link-State (Estado de Enlace) es un algoritmo utilizado en enrutamiento de redes para determinar las rutas óptimas entre nodos en una red. Su objetivo principal es crear una imagen precisa y actualizada de la topología completa de la red, lo que permite a cada nodo calcular las rutas más cortas hacia todos los demás nodos utilizando el algoritmo de Dijkstra.

Los pasos para establecer las rutas son los siguientes: descubrimientos de vecinos y enlaces de cada nodo, compartir paquetes de estado de enlace a través de la red. Estos paquetes permiten que cada nodo calcule las rutas más cortas hacia todos los demás nodos utilizando el algoritmo de Dijkstra, formando así sus tablas de enrutamiento. Conforme cambia la topología de la red, los nodos actualizan sus paquetes y recalculan las rutas para mantener la precisión y eficiencia del enrutamiento.(WAN, 2023)

Resultados

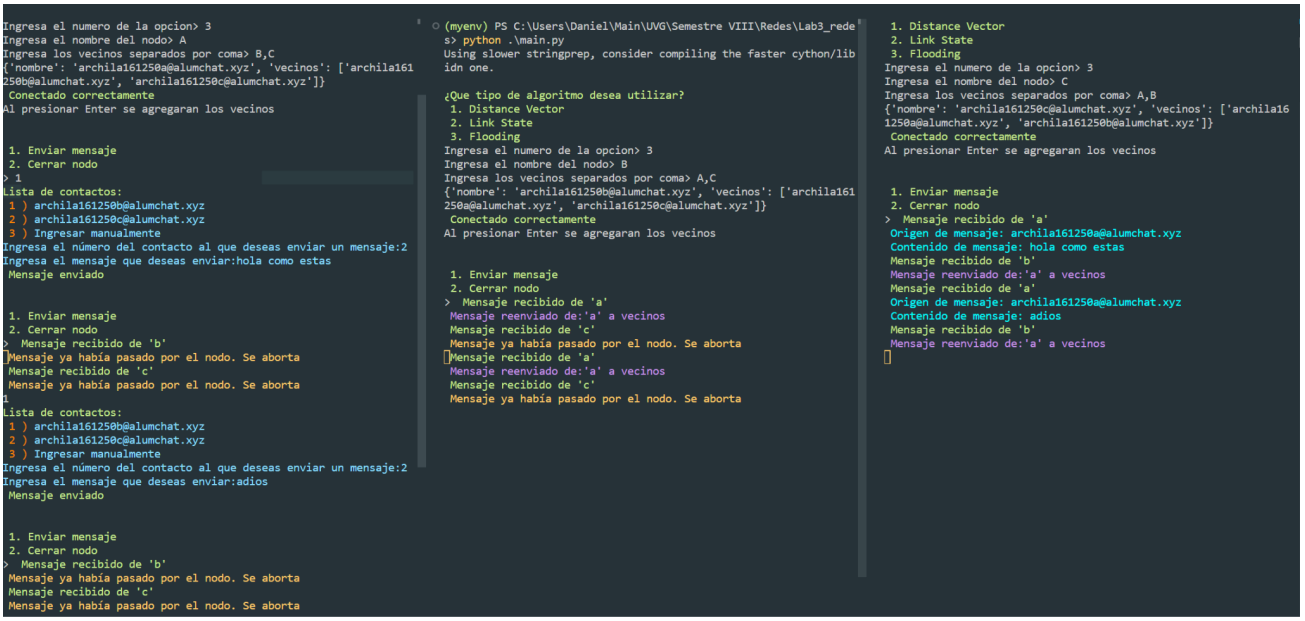

Figura 1 Envío de mensajes con Flooding

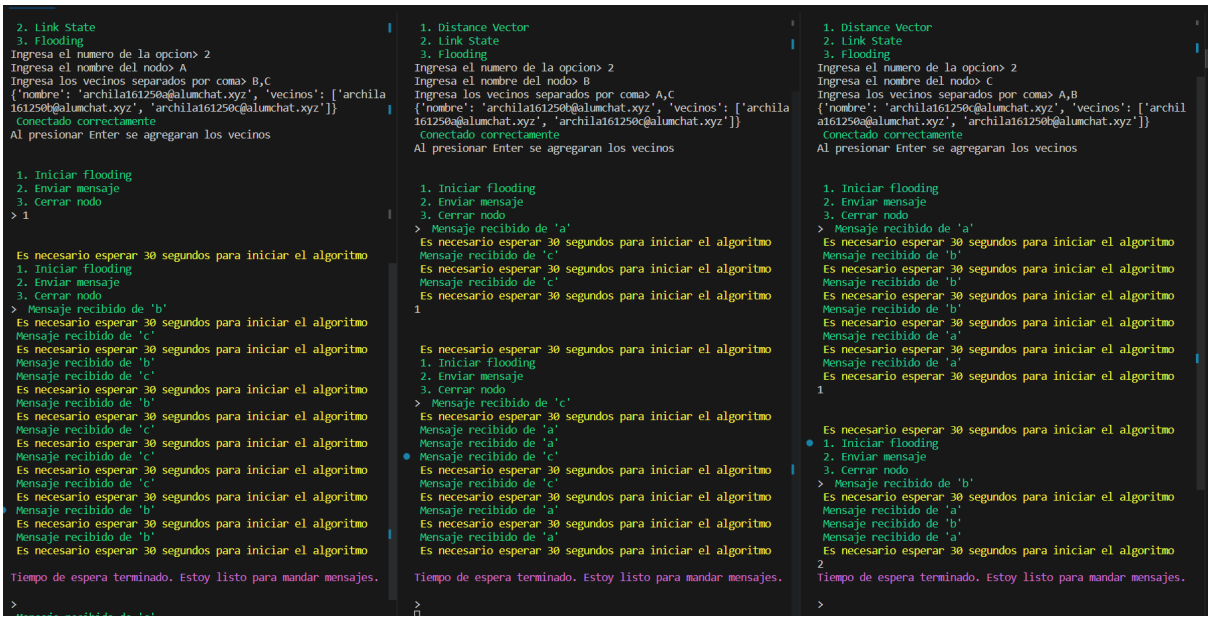

Figura 2 Mensajes de exploración Link State

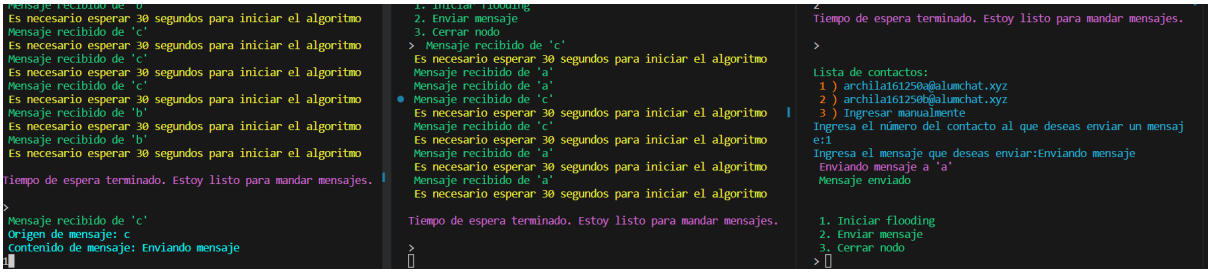

Figura 3 Envió de mensaje Link State

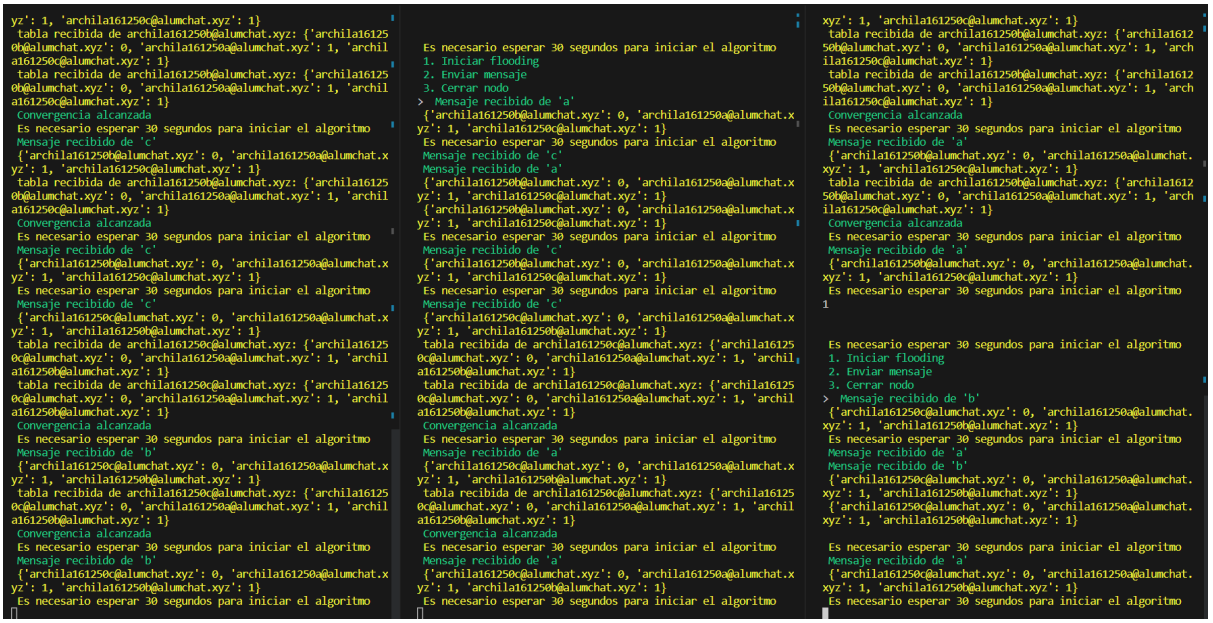

Figura 4 Intercambio de tablas Distance Vector

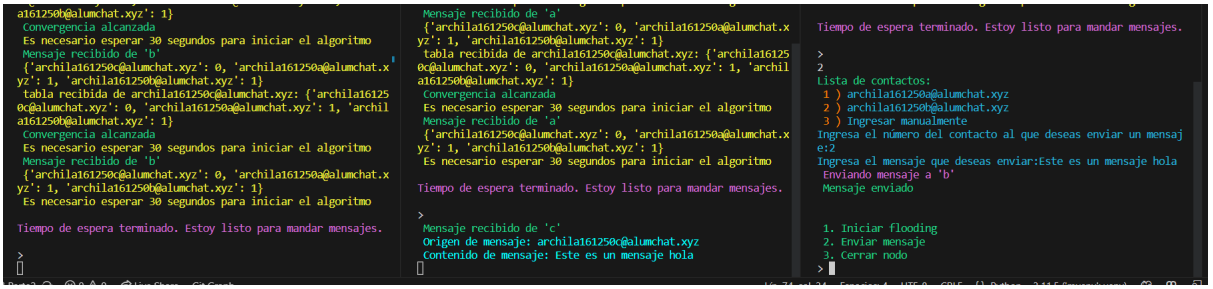

Figura 5 Envío de mensajes Distance Vector

Discusión

El algoritmo más sencillo en cuanto a implementación fue el algoritmo de Flooding, ya que su funcionamiento solo se basa en que los nodos replicarán los mensajes al momento de recibir un mensaje por medio de XMPP. Es importante destacar que se solventó uno de los mayores defectos del algoritmo de flooding, el cual es los mensajes repetidos. Para que un nodo no recibiera muchas veces el mismo mensaje en el json se agrego una sección de timestamp.Esto permitía que cuando un nodo recibía un mensaje dirigido a él, solo debía revisar si el último mensaje de dicho nodo tenía un timestamp menos o igual, si esta condición se cumplia significaba que era un mensaje repetido si no es un mensaje nuevo.

El algoritmo más complejo de implementar fue Distance Vector. Esto por la necesidad de estar mandando constantemente mensajes con la tabla correspondiente a cada nodo. En términos de carga de la red, este proceso fue bastante demandante, porque los nodos tardaban en converger. En términos de intensidad y complejidad computacional, también fue pesado, porque era necesario recalcular la tabla cada vez que llegaba un nuevo mensaje al nodo. Esto implicaba

realizar recorridos en la tabla para aplicar la ecuación de Bellman Ford. La convergencia de todos los nodos tardaba en llegar.

Además, el protocolo para este algoritmo fue más cargado, porque debía llevar un historial de los nodos por los que ya había pasado un mensaje para que no hubiera ciclos. También era necesario un timestamp, de forma que no se sobreescribieran tablas de nodos viejas en lugar de las nuevas en el nodo destino. Sin embargo, al momento de mandar mensajes, estos no debían pasar por todos los nodos, si no que seguían una ruta específica y en el receptor no se lidió con mensajes repetidos y no se sobrecargaba la red.

En cuanto a Link State no presentó mayores dificultades en la fase de exploración, pues ya se tenía la experiencia de Distance Vector. Pero si se tuvo problemas al momento de plantear en qué momento debe un nodo dejar de esperar completar de forma interna la topología. Debido a que cada nodo no sabe cuántos nodos o como es la topología no es posible saber en qué momento un nodo debe dejar de esperar mensajes de tipo info e informar que ya se pueden enviar mensajes. Para solucionar este inconveniente de implementación los nodos tienen un temporizador que se reinicia cada vez que recibe un mensaje de tipo info, si no se recibe otro mensaje de ese tipo en los siguientes 30 segundos el nodo informa que la topología está terminada y que ya es posible mandar mensajes.

En este laboratorio se evidenció la ineficiencia de Flooding, debido a su necesidad de mandar el mensaje a todos los nodos. Esto hizo que se debieran tener métodos para verificar mensajes ya recibidos. Link State tuvo su dificultad en el descubrimiento de la topología, pero computar la tabla de enrutamiento fue sencillo y el envío de mensajes no requiere tanta complejidad. Finalmente, Distance Vector tiene más flexibilidad, porque sería posible ingresar nuevos nodos en la parte de envío de tablas y se podrían volver a calcular las distancias en todos los nodos. En términos de implementación de los algoritmos en la parte 1 del laboratorio, hizo fácil adaptarlos al protocolo XMPP y envío de mensajes.

Comentario grupal

El usar XMPP para realizar la comunicación entre nodos aclaró más el funcionamiento e importancia de los algoritmos. En la parte 1 del laboratorio no se pudo ver claramente su uso, debido a la limitación de realizar el proceso manual de comunicación. Sin embargo, al usar el protocolo, fue más fácil visualizar y entender cómo se da el paso de mensajes en una red. Así como las ventajas y desventajas de cada algoritmo. Fue un laboratorio bastante interesante y un poco exigente, sin embargo, al ya tener el código de envío de mensajes del proyecto y la implementación de los algoritmos, fue casi solo usar un enfoque de “building blocks”, en donde unimos las diferentes partes que estaban dispersas.

Conclusiones

- El algoritmo de flooding es un buen método para implementar en la fase de exploración de para otras topologías como Link State y Distance Vector. Ya que permite que los nodos convergen y conozcan la distribución de la topología

- De los 3 algoritmos trabajados, Distance Vector es el algoritmo más flexible gracias a su capacidad de incorporar nuevos nodos a una topología que ya está funcionando

- Al momento de elegir qué algoritmo utilizar, se debe evaluar la complejidad y tamaño de la red, así como su capacidad de carga. También se debe entender la dinámica de la red para verificar necesidades de flexibilidad, en cuanto adición o eliminación de nodos en el tiempo.

Referencias

Terrell Hanna, K. (2021, June 1). *What is network flooding and how does it work?* SearchNetworking. https://www.techtarget.com/searchnetworking/definition/flooding

Antoniou, S. (2016). *Dynamic Routing Protocols: Distance Vector and Link State Protocols*. Pluralsight.com. https://www.pluralsight.com/blog/it-ops/dynamic-routing-protocol

WAN, L. (2023, August 3). *What are the advantages and disadvantages of distance vector and link state algorithms?* Www.linkedin.com. https://www.linkedin.com/advice/1/what-advantages-disadvantages-distance-vector-link-state