UNIVERSIDAD NACIONAL DE LOJA

FACULTAD DE LA ENERGÍA, LAS INDUSTRIAS Y LOS RECURSOS NATURALES NO RENOVABLES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

"ESTUDIO COMPARATIVO DE LAS MÉTRICAS DE CALIDAD DE SERVICIO(QoS) EMPLEANDO PROTOCOLOS DE ENCAMINAMIENTO REACTIVOS DSR (DYNAMIC SOURCE ROUTING) Y AODV (AD-HOC ON DEMAND DISTANCE VECTOR) EN REDES VANETS (VEHICULAR AD-HOC NETWORKS)"

TESIS DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTOR: CRISTIAN PAUL SAAVEDRA CAÑAR

DIRECTOR: ING. JOHN JOSSIMAR TUCKER YÉPEZ, MG. SC

LOJA – ECUADOR
2019

CERTIFICACIÓN

Ing. John Jossimar Tucker Yépez, Mg. Sc.

DIRECTOR DE TESIS

CERTIFICA:

Haber dirigido, asesorado, revisado y corregido el presente trabajo de tesis, en su proceso de investigación cuyo tema versa en **"ESTUDIO COMPARATIVO DE LAS MÉTRICAS DE CALIDAD DE SERVICIO(QoS) EMPLEANDO PROTOCOLOS DE ENCAMINAMIENTO REACTIVOS DSR (DYNAMIC SOURCE ROUTING) Y AODV (AD-HOC ON DEMAND DISTANCE VECTOR) EN REDES VANETS (VEHICULAR AD-HOC NETWORKS)",** previa a la obtención del título de **Ingeniero en Electrónica y Telecomunicaciones**, realizado por el señor: **Cristian Paul Saavedra Cañar** , la misma que cumple con la reglamentación y políticas de investigación, por lo que autorizo su presentación y posterior sustentación y defensa.

Loja, 28 de junio del 2019

.....

Ing. John Jossimar Tucker Yépez, Mg. Sc.

DIRECTOR DEL TRABAJO DE TESIS

AUTORÍA

CRISTIAN PAUL SAAVEDRA CAÑAR, declaro ser el autor del presente trabajo de titulación y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos de posibles reclamos o acciones legales por el contenido de la misma.

Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi tesis en el Repositorio Institucional – Biblioteca Virtual.

Firma: ----------

Cedula: 1104443898

Fecha: 12 de julio de 2019

CARTA DE AUTORIZACIÓN

CARTA DE AUTORIZACIÓN DE TESIS POR PARTE DEL AUTOR, PARA LA CONSULTA, REPRODUCCIÓN PARCIAL O TOTAL Y PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO.

CRISTIAN PAUL SAAVEDRA CAÑAR, declaro ser autor de la tesis titulada: **"ESTUDIO COMPARATIVO DE LAS MÉTRICAS DE CALIDAD DE SERVICIO (QoS) EMPLEANDO PROTOCOLOS DE ENCAMINAMIENTO REACTIVOS DSR (DYNAMIC SOURCE ROUTING) Y AODV (AD-HOC ON DEMAND DISTANCE VECTOR) EN REDES VANETS (VEHICULAR AD-HOC NETWORKS)",** como requisito para optar al grado de: **INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES**; autorizo al Sistema Bibliotecario de la Universidad Nacional de Loja para que con fines académicos, muestre al mundo la producción intelectual de la Universidad, a través de la visibilidad de su contenido de la siguiente manera en el Repositorio Digital Institucional:

Los usuarios pueden consultar el contenido de este trabajo en el RDI, en las redes de información del país y del exterior, con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia de la tesis que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los doce días del mes de julio del dos mil diecinueve.

Firma: ----------

Autor: Cristian Paul Saavedra Cañar

Cédula: 1104443898

Dirección: Loja, (Clodoveo Jaramillo) **Correo Electrónico:** cpsaavedrac@unl.edu.ec

Teléfono: 072613696 **Celular:**

Director de Tesis: Ing. John Jossimar Tucker Yépez, Mg. Sc.

Tribunal de Grado: Ing. Juan Gabriel Ochoa Aldeán, Mg. Sc.

Ing. Marianela del Cisne Carrión González, Mg. Sc.

Ing. Renato Benjamín Torres Carrión, Mg. Sc.

DEDICATORIA

A mis padres, Carlos y Clara A mi hermana, Ximena Cecibel. A ellos quienes con sus consejos y apoyo me alentaron a seguir adelante, aun cuando se me presentaron momentos difíciles, gracias por estar aquí.

A Erika Gianella quien estuvo junto a mi cada día, por su apoyo y cariño.

Cristian Paul Saavedra

AGRADECIMIENTO

Agradezco a Dios por la vida, por tener una familia maravillosa, y por el sin número de bendiciones recibidas a cada momento de mi vida, por ser la luz y guía que necesitaba a cada momento.

A mis padres Carlos y Clara por su apoyo incondicional durante mi formación académica, consejos oportunos y sobre todo por ser mi ejemplo a seguir.

A mi hermana Ximena, por motivarme a ser mejor cada día, y ser mi apoyo incondicional.

Al Ing. John Tucker, Mg. Sc. director del trabajo de titulación, por sus conocimientos, experiencia, paciencia, dedicación brindada y guía constante para culminar con satisfacción el presente proyecto.

A mi amigo y compañero Diego, a mi familia por su apoyo constante y desinteresado.

A la Universidad Nacional de Loja por darme la oportunidad de culminar mis estudios y así poder cumplir mis metas y sueños.

Gracias a todas las personas que con su apoyo y motivación contribuyeron a mi formación profesional.

Cristian Paul Saavedra

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ACRÓNIMOS

ACK	Acknowledgement
AODV	Ad-Hoc On Demand Distance Vector
AP	Access Point
BW	Band Width
DARPA	Defense Advanced Research Projects Agency
DSR	Dynamic Source Routing
DSSS	Direct Sequence Spread Spectrum
FHSS	Frequency Hopping Spread Spectrum
IBM	International Business Machines
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPV4	Internet Protocol Version 4
ITS	Intelligent Transportation Systems
LAN	Local Area Network
MAC	Media Access Control
MANET	Mobile Ad-Hoc Network
MG	Manhattan Grid
NS-3	Network Simulator 3
OBU	On Board Unit
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open system interconnection
PDF	Packet Delivery Funtion
PRN	Packet Radio Network
QoS	Quality of Service
RERR	Route Errors
RFC	Request for Comments

RIP	Routing Information Protocol
RREP	Route Replies
RREQ	Route Request
RSU	Road-Side Unit
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UHF	Ultra High Frecuency
V2I	Vehicle to Infraestructure
V2V	Vehicle to Vehicle
VANET	Vehicular Ad-Hoc Network
WG	Work Group
WLAN	Wireless Local Area Network

1. TÍTULO

“ESTUDIO COMPARATIVO DE LAS MÉTRICAS DE CALIDAD DE SERVICIO (QoS) EMPLEANDO PROTOCOLOS DE ENCAMINAMIENTO REACTIVOS DSR (DYNAMIC SOURCE ROUTING) Y AODV (AD-HOC ON DEMAND DISTANCE VECTOR) EN REDES VANETS (VEHICULAR AD-HOC NETWORKS)”

2. RESUMEN

En el presente trabajo se realiza el análisis y comparación de las métricas de calidad de servicio en los protocolos de encaminamiento reactivos AODV y DSR en redes VANETs. Mediante la revisión de documentación se estudia sus características, clasificación, aplicaciones, los distintos modelos de movilidad empleados para este tipo de redes, parámetros para el modelado del tráfico vehicular en el centro de la ciudad de Loja.

Para cumplir con las simulaciones se propuso que los nodos de encaminamiento sean implementados de manera virtual mediante la ayuda del software NS-3 (Network Simulator) que es de código abierto.

En cuanto a los modelos de movilidad se optó por el modelo Manhattan Grid que cumple con las características necesarias para las respectivas simulaciones, se utilizó la herramienta Bonnmotion para la creación de los diferentes escenarios donde se ha desplegado este tipo de redes, la cual tiene compatibilidad con la versión del simulador de redes (NS-2) permitiendo utilizar estos datos en Ns-3 mediante importación.

Para la obtención de los resultados se han creado tres escenarios con distinta densidad de nodos, en los que de acuerdo a los datos recuperados se puede verificar cual es el protocolo de encaminamiento que mejor se adapta al tipo de redes Ah-Hoc propuestas en el presente trabajo.

Palabras claves: Ad-Hoc, AODV, Bonnmotion, Delay, DSR, Encaminamiento, Jitter, Manhattan Grid, Métricas, Movilidad, NS-3, QoS, Throughput, VANET.

ABSTRACT.

The present work, the analysis and comparison of quality of service metrics in the reactive routing protocols is performed AODV Y DSR in network VANETs. Through the review of documentation, its characteristics, classification, applications, the different mobility models used for this type of networks, parameters for the modeling of vehicular traffic in the center of the city of Loja are studied.

In order to achieved with the simulations, it was proposed that the routing nodes be implemented in a virtual way by means of the software NS-3 (Network Simulator) that is open source.

Regarding mobility models, the Manhattan Grid model was chosen, which complies with the necessary characteristics for the respective simulations. The Bonnmotion tool was used to create the different scenarios where this type of networks has been deployed, which has compatibility with the previous version of the network simulator (NS-2) allowing to use this data in Ns-3 through import.

Finally, to obtain the results, three scenarios with different density of nodes have been created, in which according to the recovered data it is possible to verify which is the routing protocol that best adapts to the type of Ah-Hoc networks proposed in the present work.

Keywords: Ad-Hoc, AODV, Bonnmotion, Delay, DSR, Routing, Jitter, Manhattan Grid, Metrics, Mobility, NS-3, QoS, Throughput, VANET.

3. INTRODUCCIÓN

El desarrollo de las redes inalámbricas y los sistemas de comunicación han aportado de manera positiva a la sociedad, principalmente la evolución de los dispositivos móviles, ya que prestan gran movilidad a los usuarios, esto complementado con la creciente oferta de los dispositivos de comunicación como son: smartphones, tablets y los diferentes terminales móviles que día a día transforman la manera de comunicarse de las personas.

Las redes conocidas tradicionalmente como inalámbricas cuentan con un dispositivo administrador de red e infraestructura para poder conectar a los diferentes dispositivos finales. En la actualidad se maneja otro concepto de redes inalámbricas conocidas como redes Ad-Hoc o redes sin infraestructura fija, las mismas que permiten la conexión de los diferentes terminales sin la existencia de un solo dispositivo administrador, esto quiere decir que cada usuario o nodo existente en la red puede actuar como encaminador, emisor y receptor de datos.

En redes VANETs, uno de los principales objetivos es el de proporcionar calidad de servicio (QoS) en el uso de aplicaciones en tiempo real, como por ejemplo la transmisión de datos o información, este tipo de redes son conocidas por ser altamente dinámicas, lo cual produce que se generen distintas topologías creando dificultades en descubrir y mantener caminos de comunicación en los escenarios propuestos, lo que causa pérdida de conectividad entre los vehículos teniendo una reducción en la calidad de servicio.

Debido a este problema se ha propuesto realizar el estudio de dos protocolos de encaminamiento utilizados en redes Ad-Hoc como son AODV y DSR, para lo que se ha realizado la medición, comparación y análisis de las métricas de calidad de servicio (QoS) mediante la simulación de los diferentes escenarios con la ayuda del software NS-3, dentro de los cuales se utilizó Manhattan Grid como el modelo de movilidad en los escenarios elegidos.

4. REVISIÓN DE LITERATURA

4.1. Redes Inalámbricas

En el año de 1968 se da el nacimiento de la red ALOHA en la Universidad de Hawái la misma que utilizó nodos o estaciones fijas para poder realizar la comunicación entre centros universitarios. Ya en el año de 1979 el equipo de investigación de IBM publicó en el volumen 67 de la revista científica de IEEE (Institute of Electrical and Electronics Engineers) los resultados obtenidos en la implementación de una red de área local en una fábrica ubicada en suiza mediante la utilización de sensores infrarrojos, con este experimento se da el inicio de las redes inalámbricas.(Frodigh & Larsson, 2000)

Las redes inalámbricas utilizan ondas electromagnéticas para propagarse y conectar los diferentes dispositivos móviles sin la necesidad de utilizar un medio guiado, cada día toman más fuerza e importancia en la implementación de nuevas tecnologías móviles, ya que uso favorece el desarrollo de las actividades diarias en todos los ámbitos tecnológicos, en los campos laborales, académicos, de esparcimiento, el comportamiento social que día a día gana más fuerza gracias a la ayuda de la internet.

Estas redes se dividen en dos tipos como lo son: las redes inalámbricas que están formadas por una infraestructura, las cuales conectan los dispositivos a la red por medio de un Access Point (AP)/Puntos de Acceso que normalmente se utilizan routers/encaminadores y switches/conmutadores que son los llevan la información de un lugar a otro permitiendo la movilidad de los nodos por los que está constituida la red. (Curay Cuastumal, 2016)

El segundo tipo de red inalámbrica es el modo AD-HOC, esta red no utiliza una infraestructura preestablecida, ya que tiene la capacidad de recibir los datos o paquetes y reenviarlos a toda la red.

4.2. Estándar IEEE 802.11x

Estándar internacional desarrollado en el año 1997 por el grupo de trabajo (WG11) de la familia 802 perteneciente a IEEE, que trabaja con velocidades de transmisión de 2Mbps, es un conjunto de normas creadas para las comunicaciones inalámbricas, se implementa en las capas inferiores del modelo de referencia OSI (OPEN SYSTEM INTERCONNECTION) como lo son capa física y enlace de datos sobre medios inalámbricos no guiados. (Viscaino, 2018)

El estándar 802.11x se puede definir como la comunicación de los terminales de datos ya sean en medios aéreos o en ondas radio las cuales trabajan a frecuencias de 2.4 y a 5 GHz, se usan en redes de área local (LAN) la cuales utilizan diferentes técnicas de multiplexación como por ejemplos: FHSS (Espectro Ensanchado por Saltos de Frecuencia), DSSS (Espectro Ensanchado por Secuencia Directa), OFDM (Multiplexación por División de Frecuencia Ortogonal). Estas son algunas de las técnicas utilizadas para compartir el medio o canal de comunicación. (Viscaino, 2018)

A continuación, en la Tabla 1 se muestran los diferentes estándares en los que se divide la familia 802.11x:

Estándar	Denominación	Tasa Transmisión	Frecuencia de trabajo	Potencia
802.11a	WIFI5	54 Mbps	5 GHz	100 mW
802.11b	WIFI	11 Mbps	2.4 GHz	100 mW
802.11g	-	54 Mbps	2.4 GHz	100 mW
802.11h	-	54 Mbps	5 GHz	
802.11n	-	600 Mbps	2.4 GHz y 5 GHz	100 mW
802.11p	-	600 Mbps	5.90 y 6.20 GHz	
802.11s	Open80211s	600 Mbps	5GHz	100 mW
802.11ac	-	6.9 Mbps	2.5 GHz	160 mW

802.11af	-	26.7 Mbps	0.054GHz	100 mW
802.11ay	-	3 Gbps	2,4 GHz	100 mW
802.11ad	-	700 Mbps	2.4 GHz y 5 GHz	10 mW

Tabla 1 Clasificación estándares IEEE 802.11x
Fuente: (Hirai & Murase, 2018)

4.3. AD-HOC

Las primeras redes Ad-hoc nacieron en los años 70 fueron desarrolladas por DARPA (Agencia de Investigación de Proyectos Avanzados de Defensa) de los Estado Unidos, se conocían como Packet Radio Networks y operaban en bandas de Ultra alta Frecuencia UHF, este tipo de red no posee nodos fijos ni mantiene una administración centralizada. Todos los nodos son altamente dinámicos y pueden formar enlaces con sus diferentes miembros teniendo en cuenta el alcance o rango de transmisión. Cada uno de los nodos perteneciente a la red es independiente, además que pueden trabajar como router encaminando los paquetes entre los diferentes dispositivos o funcionar como terminales dependiendo de la aplicación sin que exista una conexión directa entre origen y destino.(Vidal, García, Soto, & Moreno, 2003)

Las redes ad-hoc se pueden implementar en diferentes escenarios y con distintos terminales estos pueden ser en el área marítima, aérea, vehicular, personal etc. Se utilizan múltiples tecnologías, así como protocolos de encaminamiento existentes o en estudio, los que nos permiten crear las comunicaciones inalámbricas en el estándar IEEE 802.11x, además se tratará específicamente sobre los estándares IEEE 802.11b y IEEE 802.11p que son la base para la comunicación en redes ad-hoc vehiculares.

Las características de Ad-Hoc permiten crear redes robustas y de alta eficiencia en nodos dinámicos que utilizan topologías multi-hop/multisaltos, generando así gran dificultad al establecer una comunicación. El ancho de banda/BW en este tipo de redes es reducido ya que carece de infraestructura preestablecida.

Otra característica de los nodos que operan en redes ad-hoc es la limitada energía, lo cual requiere el uso de baterías que garanticen el correcto funcionamiento de la red, este recurso es un tema de investigación por los problemas que genera la energía al momento de implementar este tipo de redes.

Las redes ad-hoc utilizan diferentes tipos de algoritmos de encaminamiento para las distintas aplicaciones que se requiera, como redes móviles MANET (Mobile Ad-Hoc Network) o redes vehiculares VANET (Vehicular Ad-Hoc Network). Estas redes son altamente dinámicas por la movilidad de sus nodos, además cada nodo perteneciente a la red puede actuar como encaminador o terminal.

Este tipo de redes se pueden implementar en diferentes escenarios de acuerdo con el número de nodos o usuarios que participarán tomando en cuenta las condiciones del lugar en el que se pretenda desarrollar la red, se puede hacer una clasificación de acuerdo con el tipo de escenario(Sarasti & Llano Ramírez, 2014), como se muestra a continuación:

- Pequeño: En este tipo de escenario permite realizar el diseño de redes MANET por el número de nodos que conforman esta clase de redes por su reducida movilidad, las que se pueden implementar en museos, centros de estudios, cafeterías, etc.

- Mediano: Para este escenario el área de implementación de la red es mayor con respeto al escenario pequeño, el número de nodos o usuarios va a aumentar ya que se tiene mayor movilidad, estas redes se podrían desarrollar en conciertos, aeropuertos, etc.

- Grande: En este modelo se tiene características diferentes a los dos escenarios revisados anteriormente ya que la cantidad de nodos aumenta considerablemente a razón de 150, 300, 500, 600 nodos con movilidad altamente dinámica como sucede en redes VANET.

4.4. VANET

VANET (Vehicular Ad-hoc Network) permiten la comunicación de varios nodos utilizando enlaces inalámbricos por medio de la utilización de redes Ad-Hoc. En esta red de acuerdo con la topología utilizada cada nodo puede actuar como un encaminador y su función es ir comunicando los datos generados entre los distintos dispositivos finales, sin que haya conexión directa entre el origen y destino. En este tipo de redes se ha generado interés por parte de las diferentes investigaciones orientadas a sectores de gobierno, automotriz, transporte, organismos que velan por la seguridad y eficiencia del transporte terrestre. En la investigación de estas redes el monitoreo de tráfico se ejecuta por medio de diferentes modelos de movilidad, los cuales indican la manera como se desplazan los vehículos en la topología utilizada.(Eze, Zhang, Liu, & Eze, 2016)

Se definen dos tipos de comunicaciones como son: V2V (vehículo a vehículo) y V2I (vehículo a infraestructura).

En el primer tipo de comunicación V2V (vehículo a vehículo) los nodos intercambian mensajes entre ellos de forma directa mediante la utilización de unidades a bordo conocidas como OBU (OnBoard Unit).(Campos, 2016)

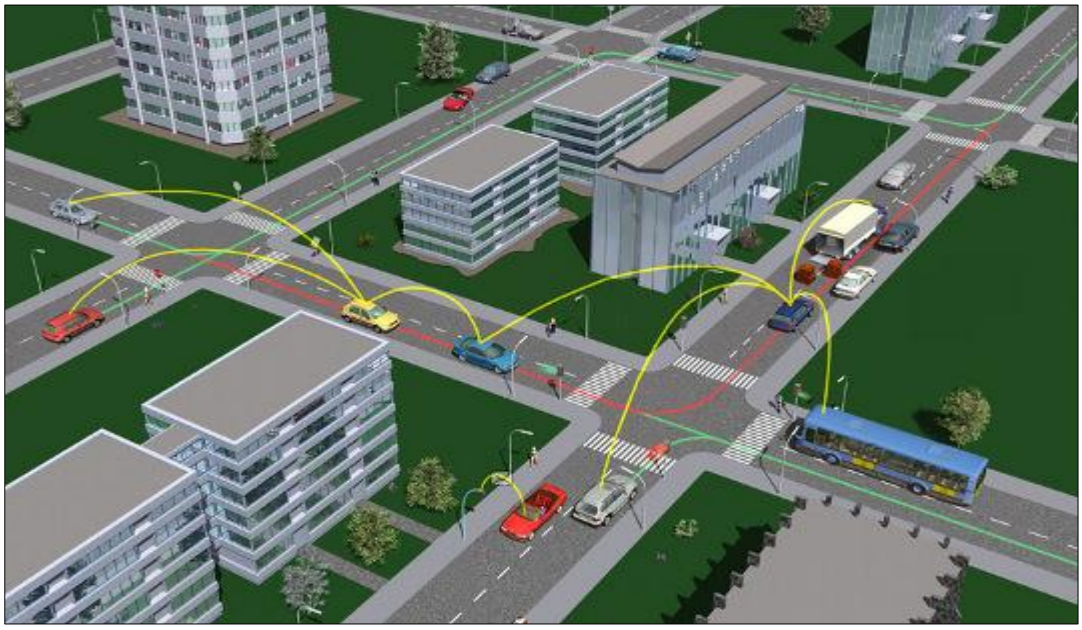


Figura 1 Redes VANET V2V
Fuente: (Marquez, 2014)

En la segunda comunicación vehículo a infraestructura, la interacción se realiza mediante dispositivos fijos los que pueden ser puntos de acceso a internet o servidores ubicados a lo largo de las vías, peajes, etc. Para esto se utiliza unidades en puntos específicos conocida como RSU (Road-Side Unit).

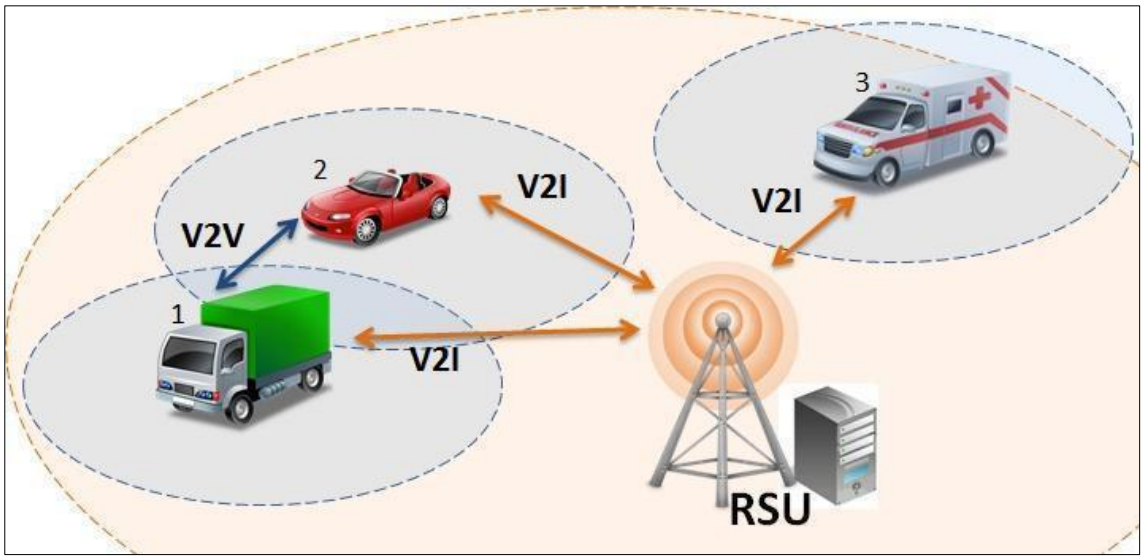


Figura 2 Redes VANET V2I

Fuente: (Massobrio & Nesmachnow, 2016)

Las características principales en una red VANET(Lares, Díaz-ramírez, Hernandez, & Quintero, 2015) se presentan a continuación:

- **Topología dinámica:** Los nodos utilizados en esta red están en movimiento constante, lo que genera gran velocidad en la vía o cuando siguen patrones predefinidos.

- **Autonomía:** Cada nodo es autónomo ya que tiene la característica de manejar la información generada en toda la red. La red no contiene infraestructura, lo que realiza en una distribución hacia todos sus nodos, lo que permite que pueda soportar diferentes fallas en la comunicación.

- **Encaminamiento distribuido:** Como se ha dicho los nodos son autónomos, estos deben tener capacidades de distribución e información a sus vecinos.

- **Terminales limitados:** Para estas redes los dispositivos terminales deben de ser livianos ya que se los instalara en los vehículos, que no poseen gran procesamiento por lo que es de suma importancia que se utilicen protocolos adecuados para optimizar recursos.

- **Energía Ilimitada:** Ya que estos nodos se ubican en los vehículos, a los sensores utilizados se les provee de energía permanentemente.

- **Redes mayores:** Esta característica trata sobre las redes que se extienden sobre el eje vial lo que genera un elevado número de nodos para lo que se debe ampliar los rangos de cobertura.

- **Capacidad Computacional**: De acuerdo con la aplicación que se vaya a implementar en la red para transferencia de datos se debe ubicar OBUs, como por ejemplo streaming de video, seguridad ante intrusiones, etc.

Las redes VANETs presentan limitaciones en su implementación a continuación, se indican algunas de las más importantes y que se deberían tener en cuenta para su desarrollo e investigación(Caballero-gil & Molina-gil, 2014).

- **Ancho de banda limitado**: Se conoce como ancho de Banda /BW a la cantidad de datos o paquetes de información que se puede enviar por medio de una comunicación de red existente, las redes VANET al carecer de infraestructura el ancho de banda que se dispone es reducido en relación con redes guiadas, esto debido a la atenuación de señales, ocasionadas por interferencias electromagnéticas.

- **Movilidad:** Uno de los objetivos de las redes inalámbricas es el poder desplazar los nodos dentro del rango de cobertura de la red, lo cual ocasiona una limitación en la movilidad, entre los nodos(vehículos) la conexión puede durar una pequeña cantidad de segundos lo que provoca una serie de problemas en la comunicación.

- **Calidad de servicio:** En redes con topologías inalámbricas se vuelve un reto garantizar la Calidad de servicio (QoS) ya que se dificulta la administración de recursos como sí se lo realiza en redes guiadas, esta característica es sumamente necesaria en redes VANETs debido al uso de aplicaciones como videoconferencias, video-streaming, información, etc.

- **Seguridad**: Se utiliza un canal de comunicación que tendría como medio el aire y al no tener infraestructura existente es más dificultoso implementar medidas de seguridad por lo que se presenta mayor vulnerabilidad a ataques externos.

Estas redes VANETs representan un sinnúmero de utilidades en cuanto a investigación y aplicación siendo unas de las principales a utilizar en los Sistemas Inteligentes de Transportes (ITS) para mejorar el control de tráfico vehicular, el conocimiento de las vías por parte de los conductores y así poder reducir el alto índice de accidentes de tránsito, además de mejorar considerablemente la fluidez vehicular.

Un parámetro importante de las redes VANETs es la seguridad activa que tiene como objetivo principal la ayuda a los conductores mediante información de distinto tipo, con el fin de estar preparados y así poder evitar accidentes o advertir de algún suceso en la vía.

A continuación, se presentan algunas características de seguridad en las redes VANETs:

Evasión de colisiones (*Collision Avoidance*): La unidad RSU advierte la probabilidad de colisión entre dos o más nodos (vehículos) e informa los conductores por medio de la unidad OBU. Los ejemplos de estas advertencias según (Hechavarria, 2016) se las considera a continuación:

- Alarma de ubicación peligrosa.

- Advertencia anticolisión.

- Alerta cambio de carril.

Advertencia de señales de tránsito: La principal función de esta característica es informar a los conductores sobre la señalización a lo largo de la vía y prestar asistencia durante el viaje.

- Advertencia de velocidad en curva.
- Alerta de evasión de señal de tránsito.

Gestión de incidentes: Empleadas en caso de accidentes de tránsito:

- Vehículo de emergencia
- Alertas de choques

Monitoreo del tráfico: Se utilizan aplicaciones que monitorean los vehículos y las condiciones en que se encuentran las vías, cuando existen problemas se informan a los conductores y a las autoridades que controlan el tránsito. Estas aplicaciones son:

- Condiciones de las vías.
- Localización y seguimiento de vehículos.
- Licencias y placas electrónicas.

Las redes VANETs permite proporcionar información en tiempo real a distintos lugares tales como policía, bomberos, hospitales, entretenimiento como parques de diversiones, centros comerciales, etc.

4.5. Protocolos de encaminamiento para redes inalámbricas.

El encaminamiento en redes se conoce como el proceso utilizado para reenviar paquetes de datos hacia el destino. Los protocolos de encaminamiento permiten la comunicación entre distintos routers, los cuales comparten información a las redes conocidas y a sus nodos más cercanos.

Existen una clasificación de los protocolos de encaminamiento en los que se pueden describir los siguientes con algunas características importantes como son:

- **Vector distancia**: Señalan la dirección y la distancia del camino a seguir por los paquetes de la red, su actualización es periódica y en sus métricas usan conteo de saltos, se calcula la mejor ruta utilizando el algoritmo Bellman-Ford. Una desventaja de los protocolos vector distancia es que son de convergencia lenta y crean fácilmente bucles.

- **Estado Enlace**: Los paquetes generados se envían a todos los nodos, lo que provoca la inundación de la red motivo por el cual se realiza una vista completa de la topología que conforma la red, y así tener un mapa de cómo están ubicados los routers y escoger el mejor camino, tienen como métrica principal el ancho de banda y utiliza el algoritmo Djkistra.

- **Fuente de enrutamiento**: La toma de decisiones sobre encaminamiento de los datos son asignadas por el origen del paquete, es decir, la fuente determina el número de saltos y los nodos intermedios que atravesará, esto genera una disminución en el consumo en recursos tanto de CPU como de ancho de banda, ya que los nodos por lo que pasa el paquete hasta llegar a su destino solo conocerán el camino del siguiente salto.

Para poder elegir qué tipo de protocolo de encaminamiento a utilizar se debe tener en cuenta la red que se va a implementar, el alcance, la forma de descubrimiento de caminos, el algoritmo que implementaran, etc.(Chirinos, 2013)

Para redes ad-hoc los protocolos se dividen en tres grupos que se van a revisar a continuación:

- **Proactivos:** En este tipo de protocolos la información de direccionamiento de los nodos se mantiene actualizada constantemente o de forma periódica, y está disponible en cada solicitud de encaminamiento por lo que genera una desventaja, que salta a la vista como es la sobrecarga por el gran tráfico que produce.

- **Reactivos:** Esta clase de protocolos contienen un algoritmo que solicita el descubrimiento del camino bajo demanda y lo mantiene mientras se

necesite o hasta que caduque, una característica importante es que no existe sobrecarga.

- **Híbridos:** El objetivo de este tipo de protocolos es unir las ventajas de los protocolos anteriores, lo que se logra haciendo que los nodos conozcan a todos sus vecinos sin generar demasiada sobrecarga.

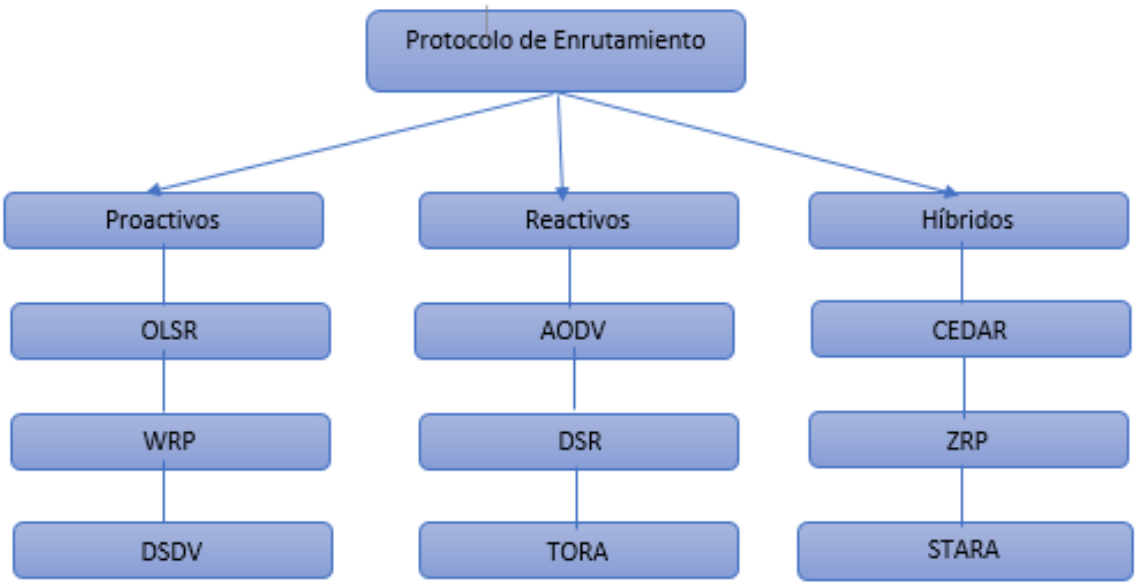


Figura 3 Clasificación Protocolos Enrutamiento

Fuente: (Rodríguez, 2015)

4.6. QoS (Métricas Calidad de Servicio)

QoS (Quality of Service), su objetivo principal se centra en ofrecer un buen servicio a los destinatarios, clasificando el tráfico generado en la red dependiendo de la prioridad que utilizan las diferentes aplicaciones en tiempo real, en la actualidad las distintas aplicaciones necesitan un ancho de banda mayor lo que genera congestión en la red y con esto la perdida de información , para evitar que ocurra el colapso de la red por la demanda de ancho de banda lo ideal es la administración y así asignar los recursos a las aplicaciones de mayor prioridad en la trasmisión de datos.

- **Fracción de entrega de paquetes (Packet Delivery Fraction**): Se refiere al número de paquetes enviados en forma completa al nodo destino, relacionado con el número de paquetes que se generan en el nodo origen.

- **Perdida de Paquetes**: Esto sucede cuando los paquetes generados debido a la congestión del canal de comunicación utilizado no llegan a su nodo de destino.

- **Retardo Promedio de paquetes (Average End-to-End delay**): Se trata de la cantidad que demora un paquete en trasladarse desde la fuente hacia destino. En el retardo de paquetes se toma en cuenta todos los retrasos provocados por retraso en cola, reenvío de paquetes de subcapa de enlace de datos MAC y demora en los tiempos de propagación y envío de los paquetes.

- **Sobrecarga de enrutamiento (Routing Overhead):** Esta métrica nos permite considerar la sobrecarga y consumo de recursos que producen los mensajes de control en la red.

- **Throughput o canal eficaz**: Se define como la cantidad que se recibe de datos en un tiempo determinado, se mide en bits por segundo, utiliza la eficiencia de la velocidad de transmisión para medir la tasa de envío de datos, el nodo destino utiliza medios físicos o lógicos.

- **Consumo de energía**: Esta métrica suele ser un problema en redes móviles, ya que los sensores son limitados, en las redes VANET no existe ese problema ya que los vehículos proveen de este recurso ilimitadamente.

4.7. Protocolo de encaminamiento de origen dinámico DSR

El protocolo de encaminamiento de origen dinámico (DSR) se creó en el año de 1994 por los investigadores Johnson y Maltz, se halla en etapa experimental se lo puede encontrar en la solicitud de comentarios (RFC) número 4728 y pertenece a la Internet Engineering Task Force (IETF) está basado en asignación de ruta lo que significa que el nodo de origen conoce toda la ruta de saltos que utiliza para llegar hacia el destino, es un protocolo simple y eficiente diseñado específicamente para que sea utilizado en redes Ad-Hoc inalámbricas con varios nodos móviles. Utilizando DSR la red es completamente auto-organizada y auto-configurable, no requiere ninguna infraestructura o administración centralizada.(Johnson, Hu, & Maltz, 2007)

Estos nodos de red colaboran en el reenvío de los paquetes de un nodo hacia el otro, y tener comunicación multi-hop/multisaltos entre nodos. Los nodos pueden moverse, unirse o salir de la red en el momento que sea necesario, todo el encaminamiento es automáticamente determinado y mantenido por el protocolo DSR.

El protocolo DSR trabaja en la capa de red del modelo TCP/IP, por lo que se utilizan distintas métricas para realizar el encaminamiento de los paquetes, como por ejemplo: Improved Expected Transmission Time (iETT), la función de esta métrica es no permitir que los flujos utilicen el canal de recursos simultáneamente cuando interfieren entre sí, en cuyo caso solo se permite el tiempo compartido, otra métrica que se utiliza en este protocolo se denomina Expected Transmission Count (ETX), Resource Aware Routing for mEsh (RARE), esta métrica emplea un monitoreo pasivo para la medición de lo los mejores caminos y así evitan la sobrecarga de red causada por el monitoreo activo, Contention-Aware Transmission Time (CATT), es una métrica de enrutamiento isotónica que representa tanto la interferencia entre flujos, así como la carga de tráfico de una manera uniforme, al hacer una suma de los retrasos de interferencia en nodos vecinos que están a 1 y 2 saltos de distancia. (Vasques, 2015)

Para poder realizar el encaminamiento utiliza los siguientes tipos de mensajes como son solicitud de camino RREQ (Route Request), respuestas de camino RREP (Route Replies) y errores de camino RERR (Route Errors) estos mensajes se envían y reciben a través del protocolo de capa de transporte UDP. La información se almacena en la memoria caché de cada nodo que conforma la red.(Johnson et al., 2007)

Las principales características de este protocolo radican en que los paquetes de encaminamiento enviados a los destinos contienen toda la información del camino a seguir. Con esto se puede decir que cada uno de los nodos almacena diferentes rutas para un mismo destino, lo que vendría siendo una ventaja en el caso de que hay errores en la transmisión, o se rompa el enlace de comunicación evitando la sobrecarga en el proceso de descubrir un camino nuevo hacia el destino. (Piles, 2016)

4.8. Protocolo encaminamiento bajo Demanda AODV

El protocolo encaminamiento bajo Demanda Ad-Hoc del Vector Distancia (AODV), fue creado en el año de 1999 por Charles E. Perkins y Elizabeth M. Royer investigadores de Sun Microsystems y de la Universidad de california respectivamente, su documentación se la puede encontrar en la Solicitud de comentarios (RFC) numero 3561 publicada en julio de 2003, se encuentra en estado experimental y pertenece a la Internet Engineering Task Force (IETF). Este protocolo es de tipo reactivo en el que cada nodo tiene una tabla de encaminamiento para los caminos conocidos. (Perkins, Belding-Royer, & Das, 2003)

Para crear esta tabla se utilizan en inicio los nodos vecinos o más cercanos, al existir la necesidad de ser ampliada lo que sucede cuando un nodo desea conectarse con otro del que no es vecino, se lo desarrolla realizando saltos a través de los nodos al largo del camino. Una de las características importantes de AODV es el consumo reducido de ancho de banda y CPU, que al ser bajo demanda no envía paquetes sin que haya la necesidad.

El protocolo de encaminamiento AODV trabaja de buena forma cuando la red a implementar tiene una gran cantidad de nodos, todas las rutas que se generen en este protocolo estarán activas cuando el nodo origen este enviando paquetes, ya que si dicho nodo ya no envía información el camino creado se mantendrá por un cierto periodo llamado tiempo de expiración, luego de este tiempo el camino se cancelará.(Maygua, 2017)

Además de decir que AODV se desenvuelve mejor en condiciones donde existe mayor movilidad, como por ejemplo en redes vehiculares como VANETs evita la formación de bucles con la ayuda de diferentes mecanismos, también detecta cambios en la topología de la red adaptando los caminos a estos cambios. Una desventaja que se le puede atribuir a AODV es que en la topología implementada para la red ningún nodo conoce la misma, sino que sólo tienen información de los nodos vecinos.

En AODV los nodos de destino de un camino crean números de secuencia antes de enviar la información de encaminamiento, una función que es muy importante ya que sirve para realizar una evaluación de como el camino que se ha creado, actualizado y así evitar la formación de bucles y estos a su vez colapsen la red. Con estos números de secuencia el protocolo elegirá el mejor camino hacia el destino, basándose en el mayor número de secuencias que pertenece la información de encaminamiento más reciente.(Gómez, Posada, & Vallejo, 2014)

Como se vi anteriormente con DSR, el protocolo AODV trabaja en la capa de red del modelo TCP/IP y contiene diferentes métricas de encaminamiento, las mismas que se las menciona a continuación: Interference-Load Aware (ILA), formada por dos tipos de métricas, Interferencia de tráfico (MTI) y costo de cambio de canal (CSC). Los dos componentes de ILA, permite capturar los efectos de la interferencia entre flujos y la diferencia en la transmisión, Interferer Neighbors Count (INX), captura la interferencia, ya que tiene en cuenta sólo la cantidad de nodos interferentes, otra métrica que se utiliza en AODV es Interference Aware routing (iAWARE), captura el efecto de la variabilidad de la relación de pérdida

del enlace, las diferencias en la velocidad de transmisión, así como las interferencias intra-flujo e inter-flujo. (Vasques, 2015)

El protocolo AODV emplea un mecanismo de descubrimiento de caminos en modo broadcast, además de soportar unicast y multicast se usan mensajes como son solicitud de camino RREQ (Route Request), respuestas de camino RREP (Route Replies) y errores de camino RERR (Route Errors), al igual que el protocolo anterior AODV también utiliza estos mensajes para poder interactuar entre el origen y el destino. (Perkins et al., 2003)

4.9. Modelos de movilidad

Se conocen por modelos de movilidad a los patrones que representan el movimiento de cada uno de los nodos que conforman una red VANET, dentro de los diferentes escenarios de movilidad en un periodo de tiempo. El modelo de movilidad es una parte fundamental del sistema que tiene como objetivo comunicar dos o más vehículos. Se lo podría definir como la parte que se encarga de especificar la forma en que se desplazan los vehículos y las decisiones que estos deben de interpretar al moverse.(Fabián Sánchez Marín & Adriana Pineda Samacá, 2016)

Estos modelos de movilidad se dividen en distintos grupos o conocidos también como modelos de tráfico que son : Modelos macroscópicos los cuales se concentran en lo que tiene que ver con la velocidad y la densidad de tráfico, el segundo grupo son conocidos como microscópicos, estos modelos nos permiten simular los movimientos de los vehículos en las vías, esto depende en gran porcentaje en las condiciones físicas del automotor para moverse y del conductor para maniobrarlo, el siguiente grupo conocido como submicroscópicos como su nombre lo dice se desprende del grupo microscópico consideran los vehículos de forma individual, y como último grupo están los modelos de simulaciones conocidas como mesoscópicas las mismas que se encuentran entre dos grupos ya vistos anteriormente, como son las macro y microscópicas es decir toman en cuenta los valores medios de velocidades en trayectos determinados de viajes. (Maldonado Narváez, 2012)

A continuación, se expondrán tres modelos pertenecientes al grupo de los microscópicos, entre ellos el modelo elegido para el presente trabajo como es Manhattan Grid.

4.9.1. Modelo Movilidad constante

Ahora revisamos el modelo de movimiento a velocidad constante (CSM) conocido por ser un modelo estático, los movimientos para cada nodo se generan de forma aleatoria. Cada conexión de los vehículos se estructura entre dos puntos es decir un emisor y receptor, se elige el destino y el modelo calcula el camino más corto posible.

Una desventaja marcada de este modelo es que al momento de calcular los movimientos de los nodos lo hace de forma individual, no tiene en cuenta los nodos que este alrededor del que transmite lo que causa solapamientos existiendo perdida de paquetes, por lo que se puede concluir que este modelo de movilidad constante funciona para nodos que se encuentran aislados no tanto así para nodos que se encuentran agrupados.

4.9.2. Modelo Kerner

Este modelo es conocido como la teoría de las 3 fases del tráfico, ya que se especializa en la congestión de vehículos, estos estados se los definen (Campos, 2016) a continuación:

- Flujo libre: Esto se presenta cuando la tasa de nodos en un tiempo es menor que el número de nodos existentes en la red, si luego de este análisis la condición anterior no se cumple significa que existe congestión en el tráfico.

- Flujo sincronizado: Esto se da cuando hay una congestión de vehículos, pero estos se pueden movilizar con fluidez debido al número de vehículos existentes, teniendo en cuenta que se debe reducir la velocidad en un extremo del camino, el flujo de tráfico viaja de manera libre mientras que en el otro extremo se tiende a crear un cuello de botella.

- Amplio congestionamiento: Este flujo se da cuando existen un alto congestionamiento de tráfico y por tal caso la velocidad de los nodos llega a ser nula, lo que se debe que en la parte inicial del camino los vehículos aceleran, por lo que tiene una vía de flujo libre, en cambio los que están atrás sufren una desaceleración por lo que experimentan un amplio congestionamiento. (Campos, 2016)

4.9.3. Modelo Manhattan Grid

Este modelo de movilidad es el que se ha elegido para realizar el presente trabajo, ya que se acopla a la propuesta de análisis y comparación de los protocolos de encaminamiento, el mismo que es basado en la movilidad para una ciudad, en nuestro caso el centro de la ciudad de Loja.

La superficie de simulación, se trata de una red de calles que emulan la parte céntrica de la ciudad, donde se despliega una red Ad-Hoc. Para la creación de las calles, numero de cuadras, límites de velocidad de los vehículos, depende del tipo de ciudad que está siendo simulada, estos parámetros forman una cuadrícula donde cada nodo perteneciente a la red inicia la simulación en un punto definido de la rejilla, cuando estos nodos llegan a un cruce eligen de forma aleatoria el siguiente movimiento.

Una de las ventajas de este modelo, es que, los nodos no se solapan, además de ser un modelo que tiene simplicidad al momento de la simulación.

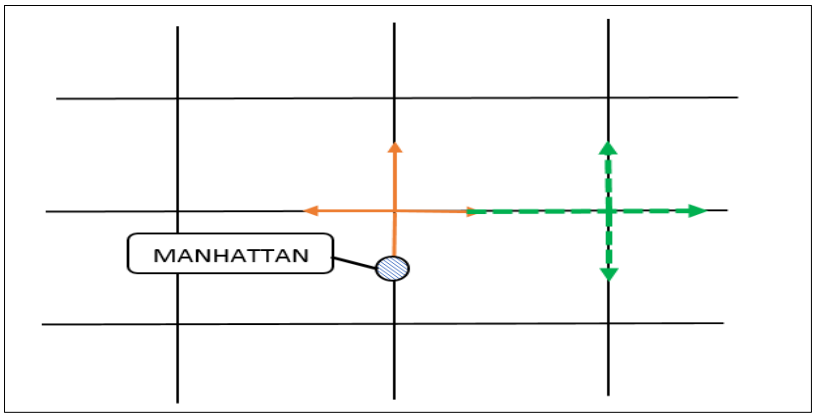


Figura 4 Modelo de movilidad Manhattan Grid

Fuente: (Olariu & Weigle, 2009)

5. MATERIALES Y MÉTODOS

5.1. Materiales.

RECURSO	Cantidad
Computador	1
Impresora	1
Software de simulación NS-3	1
Modelo de movilidad Manhattan Grid	1
Editor de texto Geany	1

Tabla 2 Materiales.

Fuente: Autor

5.2. Metodología

5.2.1 Método cualitativo

La investigación cualitativa permite interpretar con profundidad y a detalle lo que está sucediendo con un objeto de estudio, que parte de la realidad pero que dicha interpretación se hace de manera integral, el método cualitativo hace alusión a caracteres, atributos o facultades no cuantificables que pueden describir, comprender o explicar los fenómenos sociales y de estudio.(Portilla Chaves, Roja Zapata, & Hernandez Arteaga, 2014)

En este trabajo se realiza el estudio comparativo de las métricas de calidad de servicio en redes VANET, en el cual se ocupará la metodología cualitativa ya que se tomará cada uno de los atributos de los protocolos de encaminamiento AODV y DSR, en cuanto a sus métricas y se realizará la comparativa de acuerdo a los nodos que se utilice en las distintas topologías propuestas.

Tomando en cuenta las referencias teóricas y de investigación realizadas por otros autores, se obtendrá nuevos valores mediante simulaciones dando un enfoque empírico en la realización del estudio comparativo de los protocolos seleccionados.

La investigación se realizará mediante la comparación en varios escenarios con distintas densidades y velocidades de nodos, dentro de redes ad-hoc encaminadas según los protocolos DSR y AODV, tomando en cuenta para cada escenario las métricas de calidad de servicio como lo son el Throughput, Delay y Jitter.

5.3. Escenario de movilidad.

El modelo de movilidad elegido para el presente trabajo es Manhattan Grid en el que se crea un área sobre la cual se van a desplazar los nodos de forma rectangular o cuadrada, de acuerdo al número de cuadras que se van a utilizar.

Bonnmotion es la herramienta que nos permite generar los escenarios para el modelo elegido, variando la densidad de nodos para las diferentes adquisiciones de datos de las simulaciones en el software NS-3.

A continuación, se muestran los parámetros que se pueden configurar en Bonnmotion para crear el modelo de movilidad Manhattan Grid, los mismos que se han utilizado para generar el archivo de movilidad con los diferentes nodos.

Parámetro	Descripción
-f	Nombre archivo [.txt]
-d	Duración del escenario [s]
-n	Numero de nodos
-x	Ancho área simulación [m]
-y	Largo área de simulación [m]
-e	Velocidad mínima [km/h]
-m	Velocidad media [km/h]
-o	Velocidad máxima [km/h]
-u	Numero de cuadras horizontales
-v	Numero de cuadras verticales

Tabla 3 Parámetros Manhattan Grid
Fuente: Autor

5.3.1. Parámetros escenario

Para la creación del escenario, en el que se va a simular el encaminamiento de los protocolos AODV y DSR, se ha iniciado con la elección del lugar donde se va a desplegar la red Ad-Hoc en la ciudad de Loja, para lo cual se han tomado algunas consideraciones de diseño, al ser Manhattan Grid un modelo diseñado para generar movimientos que forman una cuadrícula, se ha elegido el centro de la ciudad, área que fue intervenida por la regeneración urbana y en la que se pretende optimizar el flujo vehicular por medio de la creación de redes VANETs.

Se ha tomado como referencia el área desde la calle José Antonio Eguiguren hasta la calle Lourdes, para crear el largo del escenario, y desde la Av. Universitaria hasta la calle José Joaquín de Olmedo, para crear el ancho del escenario formando así una superficie de simulación de 750 m de largo por 700 de ancho, donde se aplicarán los parámetros seleccionados.

En la figura 5 se muestra el área de creación del escenario en la ciudad de Loja, la misma que cuenta con 42 cuadras, para que los nodos pertenecientes a la red se movilicen aleatoriamente.

Figura 5 Área de simulación Ciudad de Loja

Fuente: Autor

Los parámetros para los escenarios difieren en el número o densidad de nodos que se va a utilizar.

- 25 nodos
- 50 nodos
- 100 nodos

5.4. Herramientas software

5.4.1. NS-3

Es un simulador de redes/Network Simulator, el cual se encuentra escrito en su mayoría en el lenguaje C++, Python han aportado también a su desarrollo.

NS-3 es una herramienta que nos permite la simulación de distintos tipos de protocolos de red, como los que se presentan en este trabajo es decir AODV y DSR, estos protocolos de encaminamiento se utilizan para crear redes VANETs mediante la generación de código. Además, permite la generación de tráfico, lo cual se realiza utilizando la herramienta Bonnmotion, el cual crea un archivo que se adjunta al código que se ha construido.(NS-3, 2011)

Para poder recuperar los datos que la herramienta NS-3 genera existen diferentes formas, las cuales puede ser formulas, librerías que crean ficheros, interfaz gráfica, software de análisis de tráfico, etc. A continuación, se describen algunas de ellas.

5.4.1.1. Cálculos

Una forma de extraer los datos que contienen las simulaciones de los protocolos de encaminamiento en NS-3, es mediante la utilización de fórmulas, lo cual se realiza agregando en determinada parte del código construido, las diferentes fórmulas para obtener los valores de las métricas requeridas, las cuales luego se exportan en archivos de office para su fácil interpretación.

5.4.1.2. Wireshark

Wireshark es un software de análisis de datos el cual permite capturar los paquetes generados por los envíos de los protocolos de encaminamiento, para lo cual se crea archivos con extensión “pcap”, estos archivos se crean uno para cada nodo de manera que se pueda analizar el tráfico mediante el software y así recuperar los datos requeridos. A continuación, se muestra en la figura 6 la línea de código implementada en NS-3 para poder crear los archivos pcap.

```
internet.EnablePcapIpv4All("PCAP");
```

Figura 6 Código para generar archivo pcap

Fuente: Autor

Cuando se ejecuta este comando en el código se generan archivos para cada uno de los nodos, que se encuentran en la topología como se muestran en las figuras 7 y 8 siguientes.

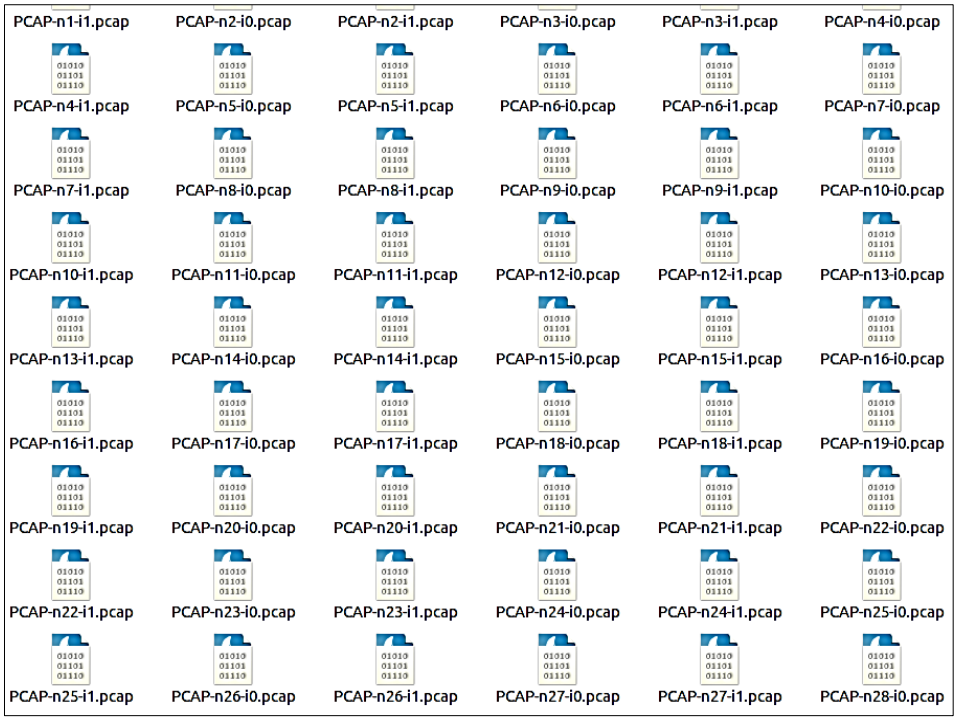


Figura 7 Archivos pcap generados

Fuente: Autor

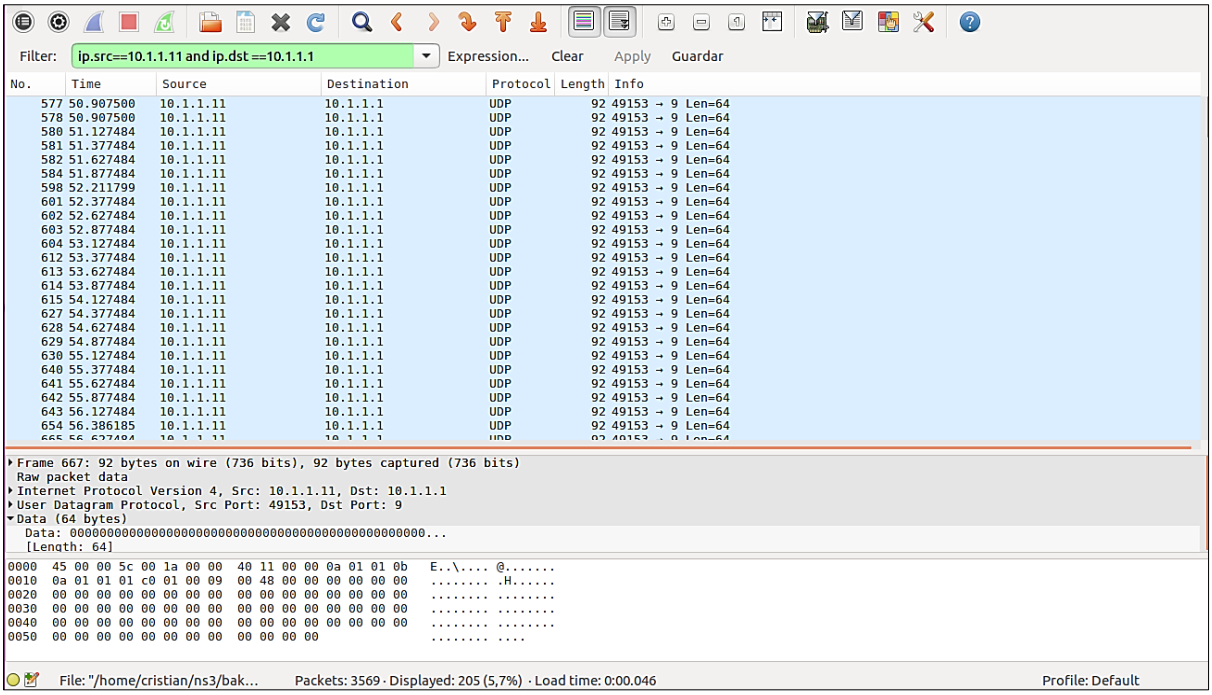


Figura 8 Análisis de tráfico en Wireshark

Fuente: Autor

5.4.1.3. NetAnim

NetAnim muestra la simulación que nos genera el software NS-3, ya que forma parte de su estructura, para poder observar dicha simulación se debe agregar las librerías correspondientes al código en el que estamos trabajando, lo que crea un archivo de rastreo "xml", el que nos permite visualizar la forma en que los nodos realizan los envíos y como se mueven de acuerdo al modelo de movilidad elegido.(J. I. Torres, 2018)

Para realizar la animación de los nodos y mostrarlos en pantalla se necesita ejecutar las siguientes líneas de código como se muestra en las figuras 9 y 10.

```
AnimationInterface anim ("comparacion_aodv.xml");
anim.EnablePacketMetadata(true);
```

Figura 9 Código para animación NetAnim

Fuente: Autor

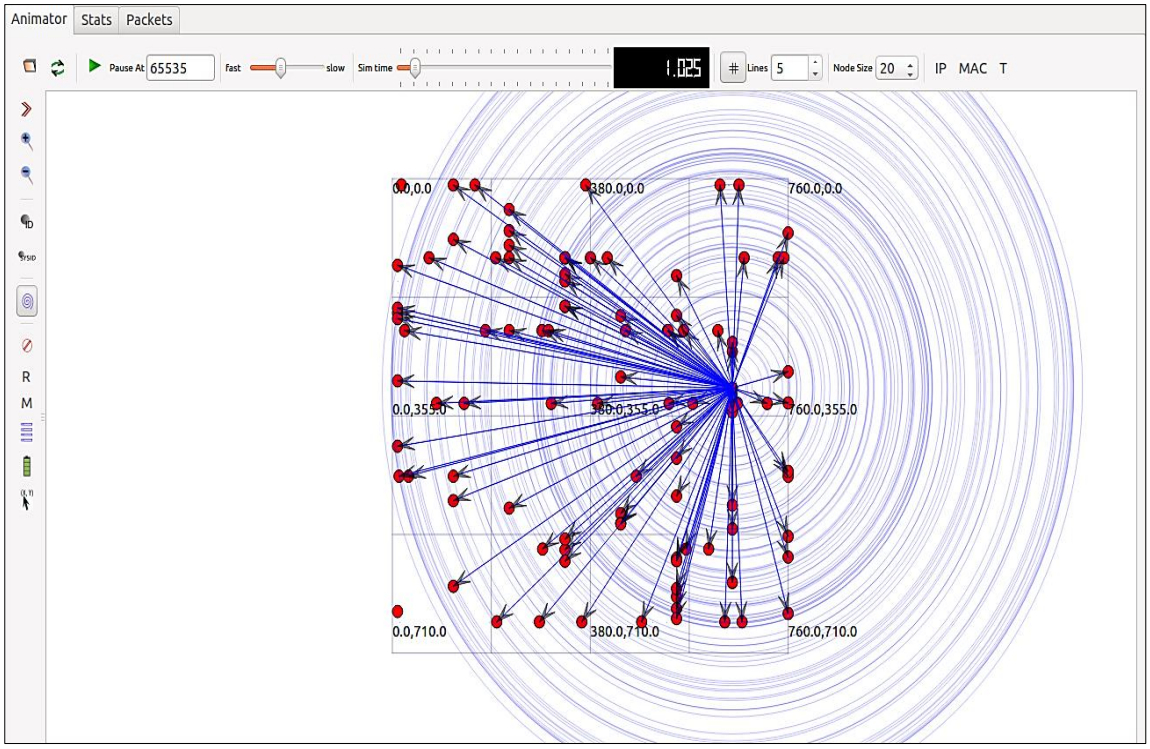


Figura 10 Animación en NetAnim de la red

Fuente: Autor

5.4.1.4. FlowMonitor

Otra forma de recopilar los datos de las métricas que generan los paquetes enviados por los protocolos de encaminamiento es conocido como Flow Monitor, que forma parte de la herramienta de simulación de redes NS-3, la que se crea agregando la librería del mismo nombre al encabezado de nuestro código y nos permite mostrar un archivo en formato "xml", las métricas de calidad de servicio (QoS) requeridas como por ejemplo Throughput, Jitter, Delay, perdida de paquetes, etc.(Fujiwara, 2017)

A continuación, se muestra la figura 11 que contiene la línea de código necesaria para recolectar estos valores.

```
Ptr<FlowMonitor> flowmon;
FlowMonitorHelper flowmonHelper;
flowmon = flowmonHelper.InstallAll ();
flowmon->SetAttribute("DelayBinWidth", DoubleValue(0.01));
flowmon->SetAttribute("JitterBinWidth", DoubleValue(0.01));
flowmon->SetAttribute("PacketSizeBinWidth", DoubleValue(1));
flowmon->SerializeToXmlFile ("AODV.xml", true, true);
```

Figura 11 Código para FlowMonitor
Fuente: Autor

Para poder ejecutar desde la consola de Ubuntu se utiliza la siguiente línea de código, para poder mostrar los valores recuperados de las diferentes métricas como se muestra en las figuras 12 y 13.

```
cristian@cristian-Dell-System-Inspiron-N4110:~$ cd ns3/bake/source/ns-3.28/
cristian@cristian-Dell-System-Inspiron-N4110:~/ns3/bake/source/ns-3.28$ ./waf --pyrun "scratch/my-flowmon-parse-results.py AODV.xml"
Waf: Entering directory `/home/cristian/ns3/bake/source/ns-3.28/build'
```

Figura 12 Ejecución flowMonitor desde consola
Fuente: Autor

```
Build commands will be stored in build/compile_commands.json
'build' finished successfully (2.230s)
Reading XML file . done.
FlowID: 1 (UDP 10.1.1.11/49153 --> 10.1.1.1/9)
TX bitrate: 2.99 kbit/s
RX bitrate: 2.99 kbit/s
Mean Delay: 141.35 ms
Packet Loss Ratio: 12.86 %
TX Bytes: 6440.00 bytes
RX Bytes: 5612.00 bytes
Hop Count: 1.69 hops
TX Packets: 70.00 packets
RX Packets: 61.00 packets
Throughput: 0.00 bps
Jitter: 102454.8200 us
FlowID: 2 (UDP 10.1.1.59/654 --> 10.1.1.74/654)
TX bitrate: 0 kbit/s
RX bitrate: 0 kbit/s
Mean Delay: 16.40 ms
Packet Loss Ratio: 0.00 %
TX Bytes: 48.00 bytes
RX Bytes: 48.00 bytes
Hop Count: 1.00 hops
TX Packets: 1.00 packets
RX Packets: 1.00 packets
Throughput: 0.00 bps
Jitter: 0.0000 us
FlowID: 3 (UDP 10.1.1.59/654 --> 10.1.1.1/654)
TX bitrate: 0.28 kbit/s
RX bitrate: 0.28 kbit/s
Mean Delay: 11.57 ms
Packet Loss Ratio: 0.00 %
TX Bytes: 192.00 bytes
RX Bytes: 192.00 bytes
Hop Count: 1.00 hops
TX Packets: 4.00 packets
RX Packets: 4.00 packets
Throughput: 0.00 bps
Jitter: 8259.4297 us
```

Figura 13 Valores de métricas en consola
Fuente: Autor

5.4.1.5. Python Viz

Esta es una herramienta que viene instalada en el software de simulación de redes NS-3, la que es conocida como un visualizador de la simulación en tiempo real para lo cual no ocupa archivos de rastreo, es de mucha importancia ya que al realizar una simulación de las redes en vivo, permite observar cómo funcionan los modelos de movilidad implementados en nuestro código, además de realizar un seguimiento a cada segundo de la simulación de los paquetes enviados, recibidos y perdidos; así como también nos muestra la dirección IP y MAC de cada nodo que se encuentra en la red. En las figuras 14 y 15 a continuación se muestra la simulación en tiempo real con la ayuda de esta herramienta.(Rodríguez, 2015)

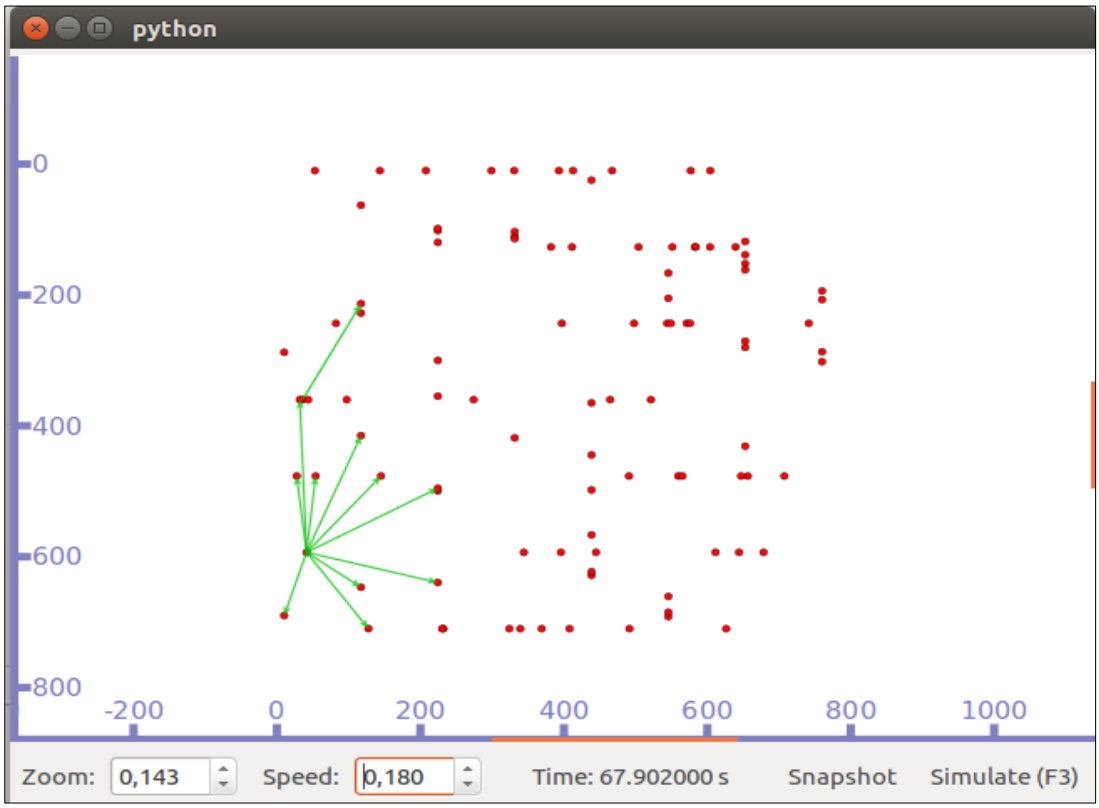


Figura 14 Simulación Python viz primeros envíos
Fuente: Autor

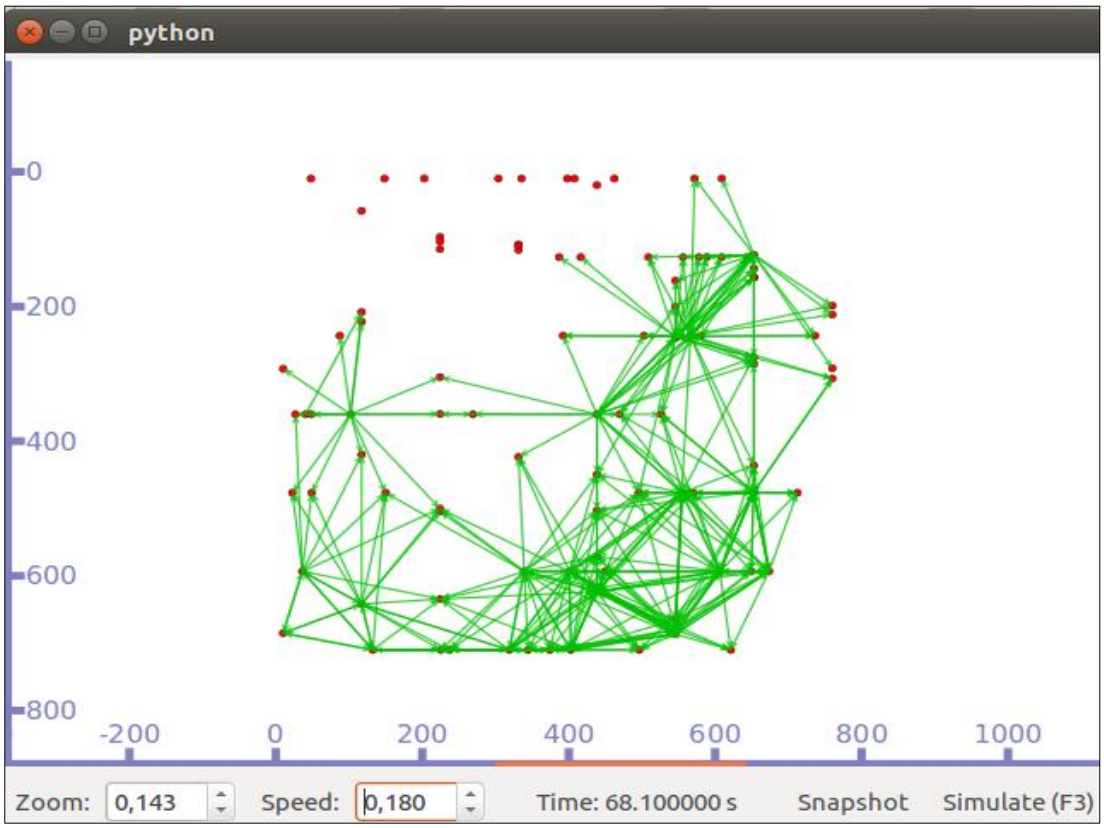


Figura 15 Movimiento nodos Python viz
Fuente: Autor

5.4.1.6. V4Ping

V4 Ping es una librería que pertenece a NS-3, el modo de funcionamiento es igual al de un ping convencional en las redes de datos, conocido en el protocolo ICMP, que sirve para realizar un diagnóstico de cómo funciona la red, y que en nuestro trabajo nos sirve para conocer tiempos de envío y respuestas; así como paquetes entregados, perdidos, con estos valores se pueden calcular métricas de calidad de servicio, las mismas que nos sirven para realizar los diferentes análisis, de cómo funciona la red, para la implementación de esta librería, se la adjunta al código y se especifica las direcciones tanto de origen como destino, en las que se desea realizar las mediciones. (NS-3, 2011)

En el código que se construye se debe agregar las siguientes líneas de código para poder observar en consola los valores generados por esta librería como se los indica en las figuras 16 y 17.

```
V4PingHelper ping (interfaces.GetAddress (size - 1));
ping.SetAttribute ("Verbose", BooleanValue (true));

ApplicationContainer p = ping.Install (nodes.Get (0));
p.Start (Seconds (0));
p.Stop (Seconds (totalTime) - Seconds (0.001));
```

Figura 16 Generación de v4ping en el código
Fuente: Autor

```
Starting simulation for 100 s ...
PING  10.0.0.10 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=7 ttl=56 time=2101 ms
64 bytes from 10.0.0.10: icmp_seq=8 ttl=56 time=1110 ms
64 bytes from 10.0.0.10: icmp_seq=9 ttl=56 time=110 ms
64 bytes from 10.0.0.10: icmp_seq=10 ttl=56 time=12 ms
64 bytes from 10.0.0.10: icmp_seq=11 ttl=56 time=9 ms
64 bytes from 10.0.0.10: icmp_seq=12 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=13 ttl=56 time=12 ms
64 bytes from 10.0.0.10: icmp_seq=14 ttl=56 time=9 ms
64 bytes from 10.0.0.10: icmp_seq=15 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=16 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=17 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=18 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=19 ttl=56 time=14 ms
64 bytes from 10.0.0.10: icmp_seq=20 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=21 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=22 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=23 ttl=56 time=14 ms
64 bytes from 10.0.0.10: icmp_seq=24 ttl=56 time=12 ms
64 bytes from 10.0.0.10: icmp_seq=25 ttl=56 time=9 ms
64 bytes from 10.0.0.10: icmp_seq=26 ttl=56 time=12 ms
64 bytes from 10.0.0.10: icmp_seq=27 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=28 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=29 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=30 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=31 ttl=56 time=10 ms
64 bytes from 10.0.0.10: icmp_seq=32 ttl=56 time=11 ms
64 bytes from 10.0.0.10: icmp_seq=33 ttl=56 time=12 ms
--- 10.0.0.10 ping statistics ---
100 packets transmitted, 27 received, 73% packet loss, time 99999ms
rtt min/avg/max/mdev = 9/132.7/2101/446.6 ms
cristian@cristian-Dell-System-Inspiron-N4110:~/ns3/bake/source/ns-3.28$
```

Figura 17 V4ping mostrado en consola
Fuente: Autor

5.5. Protocolos de Encaminamiento

En el presente trabajo se ha definido como objetivo principal la comparativa de los protocolos de encaminamiento AODV y DSR sobre redes VANETs, analizando métricas de calidad de servicio (QoS) y de cómo cada uno de dichos protocolos se desenvuelven en los escenarios propuestos en el centro de la ciudad de Loja, que es donde se ha creado el modelo de movilidad.

Para realizar la comparativa de estos protocolos se deben obtener los distintos valores de las diferentes métricas existentes, y así realizar el análisis en la topología de la red Ad-Hoc desplegada en la ciudad.

5.5.1. Métricas

Como ya se ha mencionado anteriormente existen diferentes métricas de calidad de servicio para analizar los protocolos seleccionados, de las que se han escogido tres con las que se ha creído conveniente realizar el presente trabajo, debido a que son de suma importancia en el análisis de este tipo de redes, es decir, el Throughput que se conoce como la eficiencia con que se entregan los paquetes, es la métrica que más utilizada en este tipo de comparaciones ya que permite conocer que cantidad del mensaje ha sido recibido, el Delay es necesario ya que se debe conocer el tiempo que se demoran en llegar los paquetes a su destino, así mismo el Jitter permite saber las variaciones de retrasos en la red y así conocer en que parte se están perdiendo los envíos y predecir el comportamiento del protocolo en dicho modelo, con estas métricas se cumplir con los objetivos propuestos y definir cual protocolo tiene mejor eficiencia de acuerdo con el escenario y la densidad de nodos elegidos.

En la tabla 4 se muestra la comparación de las métricas de calidad de servicio (QoS), de acuerdo a las características de simulación y en base a estos parámetros se ha elegido las métricas para el presente proyecto.

Métrica	Facilidad de Implementación de código	Obtención mediante librerías	Códigos previos	Obtención en los protocolos	Total
Throughput	X	X	X	X	4
Delay	X	X	X	X	4
Jitter	X	X	X	X	4
Consumo energía		X			1
Pérdida paquetes		X	X		2

Tabla 4 Comparación métricas de calidad de servicio
Fuente: Autor

5.5.1.1. Throughput

Una de las métricas elegidas dada su importancia en el análisis de redes VANETs es el throughput ya que al comparar redes que se caracterizan por la elevada movilidad de sus nodos o vehículos, se debe conocer la eficiencia con que la que llegan los paquetes hacia su destino y así mismo conocer la cantidad de datos perdidos en los trayectos de los escenarios propuestos.(Rodríguez, 2015)

5.5.1.2. Delay

La segunda métrica elegida para realizar la comparativa y posterior análisis de los protocolos a utilizar es el Delay ya que es una de las métricas más importantes en el rendimiento de nuestra red, la cual nos permite conocer el tiempo que demora un paquete en viajar desde el nodo de origen hacia el nodo de destino, conociendo estos valores se puede saber cuan distantes se encuentran los nodos y si el tamaño del escenario creado es válido para el número de nodos que se encuentran moviéndose en la red. (Rodríguez, 2015)

5.5.1.3. Jitter

La tercera y última métrica que se calcula en esta comparativa es conocida como Jitter y se define como la variación entre los retardos de transmisión de los paquetes, esta métrica es importante debido que al conocer el tiempo que genera estos retrasos se puede deducir su causa, dicha variación depende de las rutas que eligen los nodos para la transmisión o por el aumento de flujo en cada nodo intermedio de la red, lo cual varía con cada protocolo y número de usuarios. (Rodríguez, 2015)

5.6. Estructura del código NS-3

En la construcción del programa que es necesario para cumplir con los objetivos señalados para el presente trabajo, se ha tomado como base el ejemplo descrito en el software de simulación NS-3 conocido como "Manet-routing-compare" (NS-3, 2011), el cual ha sido creado en software libre y con licencia pública de GNU version 2, fue desarrollado por Justin Rohrer perteneciente al Centro de Tecnología de la Información y las Telecomunicaciones (ITTC) y el Departamento de Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Kansas Lawrence, KS USA.(NS-3, 2011)

Este algoritmo se ha modificado en sus diferentes fases para poder ejecutar dicho proyecto, que tiene como finalidad la simulación de protocolos de encaminamiento en redes VANETs, donde el principal cambio se da en la adquisición del modelo de movilidad, que es el que genera los desplazamientos con los diferentes tipos de redes.

5.6.1. Librerías Utilizadas

Para iniciar con la construcción del programa, se necesitan adjuntar las librerías provistas por el software NS-3, para que cada bloque construido funcione de manera correcta. En la figura 18 se muestran las librerías que se usaron.

```
#include <fstream>
#include <iostream>
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/wifi-module.h"
#include "ns3/dsr-module.h"
#include "ns3/applications-module.h"
```

Figura 18 Librerías utilizadas
Fuente: Autor

5.6.2. Definición de nombres

En las siguientes figuras 19 y 20 se muestran la definición de los nombres que se van a utilizar a lo largo del programa, para hacer más fácil el manejo de las variables en la programación tanto para el protocolo AODV como para DSR.

```
using namespace ns3;
using namespace std;
```

Figura 19 Definición de nombres para AODV
Fuente: Autor

```
using namespace ns3;
using namespace dsr;
using namespace std;
```

Figura 20 Definición de nombres para DSR
Fuente: Autor

5.6.3. Función Principal

En esta etapa se define la función principal en donde se realiza la simulación, se crean la variables y archivos donde se van a guardar los resultados que sean requeridos, se definen los "sinks" o nodos sumideros que se van a utilizar, potencia ocupada, etc. A continuación, se muestra en la figura 21 la forma en que está constituida esta parte.

```
int
main (int argc, char *argv[])
{
 RoutingExperiment experiment;
 string CSVfileName = experiment.CommandSetup (argc,argv);
 ofstream out (CSVfileName.c_str ());
 out << "Tiempo," <<
 "Throughput," <<
 "Paquetes Recibidos," <<
 "Ip Origen ," <<
 "Ip Destino," <<
 "Protocolo," <<
 "Delay,"<<
 "Jitter"<<
 endl;
 out.close ();
 int nSinks = 5;
 double txp = 7.5;

 experiment.Run (nSinks, txp, CSVfileName);
}
```

Figura 21 Definición de Función principal del programa
Fuente: Autor

5.6.4. Topología

Para definir la topología a utilizar, se realiza la creación de los nodos wifi, ya que el tipo de redes a simular en nuestro caso es del tipo Ad-Hoc entonces esto se lo muestra a continuación en la figura 22.

```
NodeContainer adhocNodes;
adhocNodes.Create (nWifis);
```

Figura 22 Creación de los nodos Ad-Hoc

Fuente: Autor

Para realizar la propagación y comunicación de las capas inferiores del modelo de red, se utiliza la configuración que nos permita realizar la conexión de la capa física y enlace de datos con la capa de red, donde se desenvuelven los protocolos estudiados.

A continuación, se muestran los "helpers" que realizan las tareas repetitivas de manera más sencilla, además de facilitar la conexión entre los nodos de la red, evitan que los programas se vuelvan más largos y difíciles de entender.(NS-3, 2011)

```
WifiHelper wifi;
wifi.SetStandard (WIFI_PHY_STANDARD_80211b);

YansWifiPhyHelper wifiPhy =  YansWifiPhyHelper::Default ();
YansWifiChannelHelper wifiChannel;
wifiChannel.SetPropagationDelay ("ns3::ConstantSpeedPropagationDelayModel");
wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel");
wifiPhy.SetChannel (wifiChannel.Create ());

WifiMacHelper wifiMac;
wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager",
                              "DataMode",StringValue (phyMode),
                              "ControlMode",StringValue (phyMode));

wifiPhy.Set ("TxPowerStart",DoubleValue (txp));
wifiPhy.Set ("TxPowerEnd", DoubleValue (txp));

wifiMac.SetType ("ns3::AdhocWifiMac");
NetDeviceContainer adhocDevices = wifi.Install (wifiPhy, wifiMac, adhocNodes);
```

Figura 23 Etapa de propagación y conexión inalámbrica

Fuente: Autor

5.6.5. Movilidad

En cuanto a la movilidad en el presente trabajo se ha elegido el modelo denominado Manhattan Grid, que de acuerdo con la simulación en redes VANETs nos genera un escenario similar a una matriz, lo que se relaciona al centro de la ciudad de Loja, en el software de NS-3 nuestro modelo no se encuentra en las librerías por lo que se debe generar en Bonnmotion, que es compatible con NS-2, lo que indica que se debe importar el modelo generado en la herramienta antes mencionada, para lo cual se debe agregar las siguientes líneas de código a la etapa de movilidad del programa construido como se muestra en la figura 24.

```
MobilityHelper mobilityAdhoc;
 std::string traceFile = "mod_100.ns_movements";
//Helper ns2 leyendo el archivo de traza generado
Ns2MobilityHelper ns2 = Ns2MobilityHelper (traceFile);
ns2.Install ();
```

Figura 24 Código para importar modelo de movilidad

Fuente: Autor

5.6.6. Encaminamiento de red

Esta etapa corresponde a la parte central del presente trabajo, ya que nos referimos al encaminamiento de los nodos en redes VANETs, es decir la capa de red en la que se desenvuelven los protocolos elegidos para el análisis como son AODV y DSR, para poder realizar el encaminamiento se debe agregar la librerías para cada uno de los protocolos a utilizar, como se indicó anteriormente, una vez definido esto se activa los "helpers" o ayudantes, ya creados en el simulador y así habilitar para cada protocolo el encaminamiento como se puede observar en la las figuras 25 y 26 a continuación.

```
AodvHelper aodv;
Ipv4ListRoutingHelper list;
InternetStackHelper internet;
list.Add (aodv, 100);
internet.SetRoutingHelper (list);
internet.Install (adhocNodes);
```

Figura 25 Ayudante de encaminamiento para protocolo AODV

Fuente: Autor

```
DsrHelper dsr;
DsrMainHelper dsrMain;
Ipv4ListRoutingHelper list;
InternetStackHelper internet;
m_protocolName = "DSR";
internet.Install (adhocNodes);
dsrMain.Install (dsr, adhocNodes);
```

Figura 26 Ayudante de encaminamiento para protocolo DSR

Fuente: Autor

Para poder completar el encaminamiento de la red se debe proveer de una dirección a cada nodo que conforma la red, para esto se debe realizar la asignación del direccionamiento IPV4, cabe recalcar que el tamaño de direcciones asignadas a la red depende del número de nodos a utilizar en la topología, para poder implementar el direccionamiento se lo debe agregar las siguientes líneas de código como se muestra en la figura 27.

```
NS_LOG_INFO ("assigning ip address");
Ipv4AddressHelper addressAdhoc;
addressAdhoc.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer adhocInterfaces;
adhocInterfaces = addressAdhoc.Assign (adhocDevices);
```

Figura 27 Direccionamiento IPV4

Fuente: Autor

5.6.7. Obtención de Datos

Para la recuperación de datos de cada una de las métricas en el presente trabajo se lo realizará mediante cálculos, que se los ubicará en el código construido en el software NS-3, dichas operaciones se las hará para las tres métricas de calidad de servicio (QoS) ya definidas: Throughput, Delay y Jitter, para lo que se ha dispuesto las siguientes formulas:

En la métrica del Throughput, se realiza el producto entre el número de paquetes en bytes por 8 para realizar la conversión a bits, esto se divide para el tiempo de simulación, obteniendo un resultado en kbps.

$$Th = \frac{bytes*8}{T_s*1000}$$ ***Ecuación 1***

Donde:

$\boldsymbol{Th}$ = Throughput

$\boldsymbol{bytes}$ = Total de Bytes de cada Paquete

$\boldsymbol{Ts}$ = Tiempo de Simulación

En el Delay o retardo, se debe considerar el tiempo en que llega cada paquete, con este valor se obtiene la diferencia entre el tiempo actual y tiempo anterior en el que llega el paquete previo, para esto se utiliza la siguiente formula.

$$Dly = T_A - T_P$$ ***Ecuación 2***

Donde:

$\boldsymbol{Dly}$ = Delay

$\boldsymbol{T_A}$ = Tiempo Actual de Simulación o de Recepción

$\boldsymbol{T_P}$ = Tiempo Previo de Simulación o Recepción

Para realizar el cálculo del Jitter se obtiene a partir de la variación de los datos de la métrica calculada anteriormente; es decir el retardo o Delay para lo que se realiza la diferencia entre retardo actual y retardo anterior, luego de realizar este cálculo al valor resultante se le realiza la diferencia del valor de Jitter previo y se divide todo para 16 que de acuerdo con el RFC 3393 se utiliza para reducir el ruido.

$$Jitt = \frac{[(Dly_A - Dly_P) - Jitt_P]}{16}$$ ***Ecuación 3***

Donde:

$\boldsymbol{Jitt}$ = Jitter

$\boldsymbol{Dly_A}$ = Delay del paquete Actual

$\boldsymbol{Dly_P}$ = Delay del paquete Previo

$\boldsymbol{Jitt_P}$ = Jitter del paquete Previo

Las fórmulas para el cálculo de las métricas se agregan al programa construido como se muestra a continuación en la figura 28.

```
void
RoutingExperiment::ReceivePacket (Ptr<Socket> socket)
{
  Ptr<Packet> packet;
  Address senderAddress;
  while ((packet = socket->RecvFrom (senderAddress)))
    {
      a=socket->GetNode ()->GetId ();
      bytesTotal += packet->GetSize ();
      packetsReceived += 1;
       if (a==0){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time0)*1000;
        m_time0 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter0);
        m_jitter0 =abs (abs(jitterSum) - m_jitter0) / 16;
        m_jitter=m_jitter0;
        jitter0 = delaySum;
        time=jitter0;
      NS_LOG_UNCOND (PrintReceivedPacket (socket, packet, senderAddress));
void
RoutingExperiment::CheckThroughput ()
{
  double kbs = (bytesTotal * 8.0) / (1000*(time/1000));
  bytesTotal = 0;
  out.close ();
    jitterSum = 0;
  delaySum = 0;
  jitter=0;
  m_jitter=0;
  packetsReceived = 0;
  Simulator::Schedule (Seconds (0.001), &RoutingExperiment::CheckThroughput, this);
```

Figura 28 Cálculo de métricas de calidad de servicio

Fuente: Autor

5.6.8. Aplicaciones y simulación

Se define como aplicación el ayudante *OnOffHelper*, que tiene dos niveles, encendido que es el estado en el que se genera tráfico en la red y el apagado que sería el estado donde el tráfico se mantiene inactivo, este tráfico se caracteriza por la velocidad de los datos y el tamaño de los paquetes a transmitir, esta opción se la agrega utilizando el código que se muestra a continuación en la figura 29.

```
OnOffHelper onoff1 ("ns3::UdpSocketFactory",Address ());
onoff1.SetAttribute ("OnTime", StringValue ("ns3::ConstantRandomVariable[Constant=1.0]"));
onoff1.SetAttribute ("OffTime", StringValue ("ns3::ConstantRandomVariable[Constant=0.0]"));
```

Figura 29 Aplicación para activación de generación de tráfico
Fuente: Autor

Para finalizar la construcción del programa se debe declarar la parte del código que nos permita iniciar y terminar la simulación, para esto se agrega lo siguiente.

```
Simulator::Stop (Seconds (TotalTime));
Simulator::Run ();
Simulator::Destroy ();
```

Figura 30 Simulación del programa

Fuente: Autor

6. RESULTADOS

6.1. Generación de Traza

Cuando se tiene ya definidos los parámetros del escenario de movilidad que se va a simular, se procede a generar las trazas para el movimiento de los nodos correspondientes, lo que se lo realiza mediante la herramienta Bonnmotion, que es compatible con el Sistema operativo Ubuntu.

.

A continuación, se muestra la forma como se ingresan los datos definidos para los diferentes escenarios en este caso el modelo Manhattan Grid.

Como primer paso abrimos una terminal y nos ubicamos en la carpeta Bonnmotion 3.0.1 para poder generar las trazas como se muestra en la figura 31 a continuación:

```
cd  bonnmotion -3.0.1/bin
```

Figura 31 Ubicación en la carpeta de Bonnmotion

Fuente: Autor

Con la siguiente línea se muestra la ayuda para crear el modelo correspondiente en nuestro caso Manhattan Grid esto se indica en las figuras 32 y 33.

```
./bm -hm ManhattanGrid
```

Figura 32 Creación del modelo Manhattan Grid

Fuente: Autor

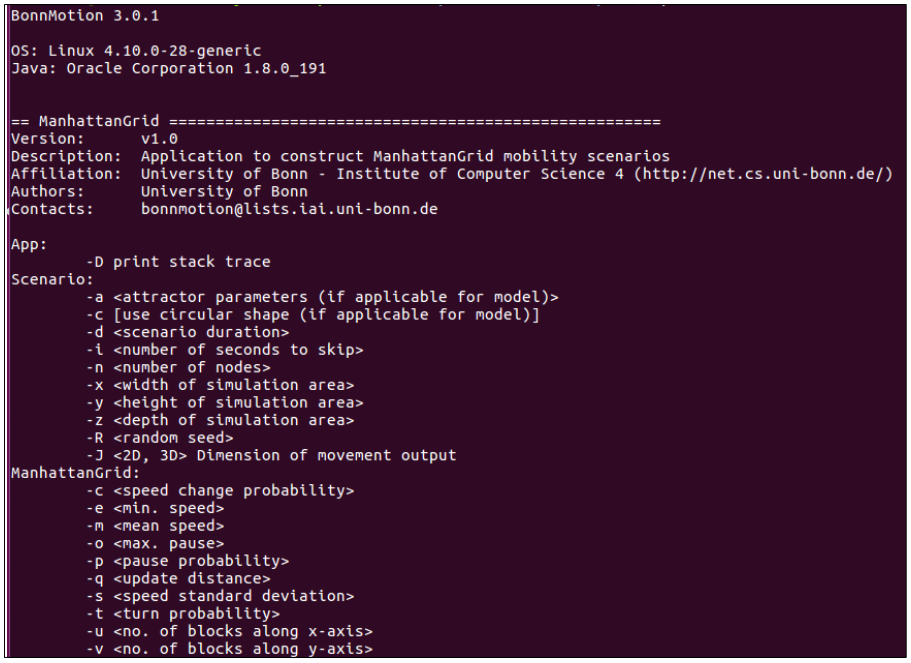


Figura 33 Manhattan Grid en Bonnmotion

Fuente: Autor

Para el ingreso de los parámetros, se utiliza la siguiente línea de comando con cada uno de los valores definidos en las tablas de cada escenario como esta en la figura 34.

```
./bm -f MOD_50 ManhattanGrid -d <200> -n <50> -x <750>
-y <700> -e <10> -m <25 >-o <40 >-u <7> -v <6>
```

Figura 34 Ingreso de parámetros para creación de escenario

Fuente: Autor

Una vez realizado esto, se genera un archivo con extensión "movements" en el cual se encuentran todas las trazas de movimientos para cada uno de los nodos que se ingresó.

El archivo que nos entrega Bonnmotion tiene una extensión, pero aún no se lo puede exportar hacia el software NS-3, para lo cual hay que ingresar otra línea de comando para que el archivo sea compatible y poder trazar en la simulación.

```
./bm NSFile -f MOD_50
```

Figura 35 Creación de archivo para exportar a Ns-3

Fuente: Autor

Con la ayuda de este comando creamos un archivo con extensión ".ns_movements" el cual ya es posible utilizar y exportar a nuestro código generado en NS-3.

6.2. Escenarios

En la tabla 5 que se muestra a continuación, se puede observar los parámetros elegidos para la simulación de nuestro modelo, en los que se define el tamaño del paquete a enviar, que en el Anexo I de este documento se lo muestra en el código fuente con el valor en bytes de 64, lo que transformado a su valor en bits tenemos 512, así mismo el número de paquetes generados, la duración de la simulación, además se muestra valores de los nodos Sink que se han utilizado para los tres escenarios propuestos.

Parámetros simulación	**Valores**
Tamaño Paquete	512 bits
Paquetes Generados	200
Duración simulación	50 segundos
Protocolos encaminamiento	AODV, DSR
Nodos Sink (sumidero)	1, 3,5

Tabla 5 Parámetros simulación

Fuente: Autor

A continuación, en la tabla 6 se presenta los parámetros que se utilizaron para la creación del modelo de movilidad Manhattan Grid, con la ayuda de la herramienta Bonnmotion se generó los archivos que contienen los patrones de movilidad para cada uno de los nodos, se ha tenido en cuenta que estos valores aquí presentados sean semejantes a la realidad, lo cuales comprenden la superficie de simulación, duración, velocidades de los vehículos, etc.

Parámetros Modelo	Cantidad
Área de escenario [m]	x=750 y=700
Modelo de movilidad	Manhattan Grid
Duración de simulación [s]	200
Velocidad mínima [km/h]	10
Velocidad media [km/h]	25
Velocidad máxima [km/h]	50
Número de cuadras horizontales	7
Número de cuadras verticales	6

Tabla *6 Parámetros movilidad 25 nodos*
Fuente: Autor

Para cada escenario propuesto se obtuvieron 9 valores de los datos recolectados, es decir, se han realizado simulaciones con diferentes valores de nodos sumideros/ Sink, como se muestran en las tablas 7, 8 ,9 de este documento.

Cuando se utiliza 1 nodo sumidero, cada envió se lo realiza entre el nodo 0 y el nodo 10, para los cuales se asignan las direcciones IP 10.1.1.1 para el nodo de origen y 10.1.1.11 para el nodo de destino. Lo mismo sucede cuando se aumenta el número de nodos sumidero, es decir que si tenemos 3 Sink se realiza envíos entre los nodos 0, 1,2 como origen y serían los nodos 10, 11, y 12 los que reciban la información, y así sucesivamente cada vez que se aumente el número de nodos sumideros.

6.2.1. Escenario 1

- 25 nodos

De acuerdo a los parámetros elegidos para el escenario de 25 nodos se procede a simular en NS-3, y así obtener los valores de las métricas a comparar, para apreciar el movimiento de los nodos en el escenario propuesto se ha utilizado la herramienta Python Viz, la que nos permite observar el comportamiento de los usuarios de acuerdo con el modelo de movilidad propuesto.

A continuación, se muestran dos gráficas la figura 36 nos indica los envíos de paquetes que se realizan durante la simulación, la figura 37 nos muestra el descubrimiento que hacen los protocolos por cada nodo perteneciente al escenario.

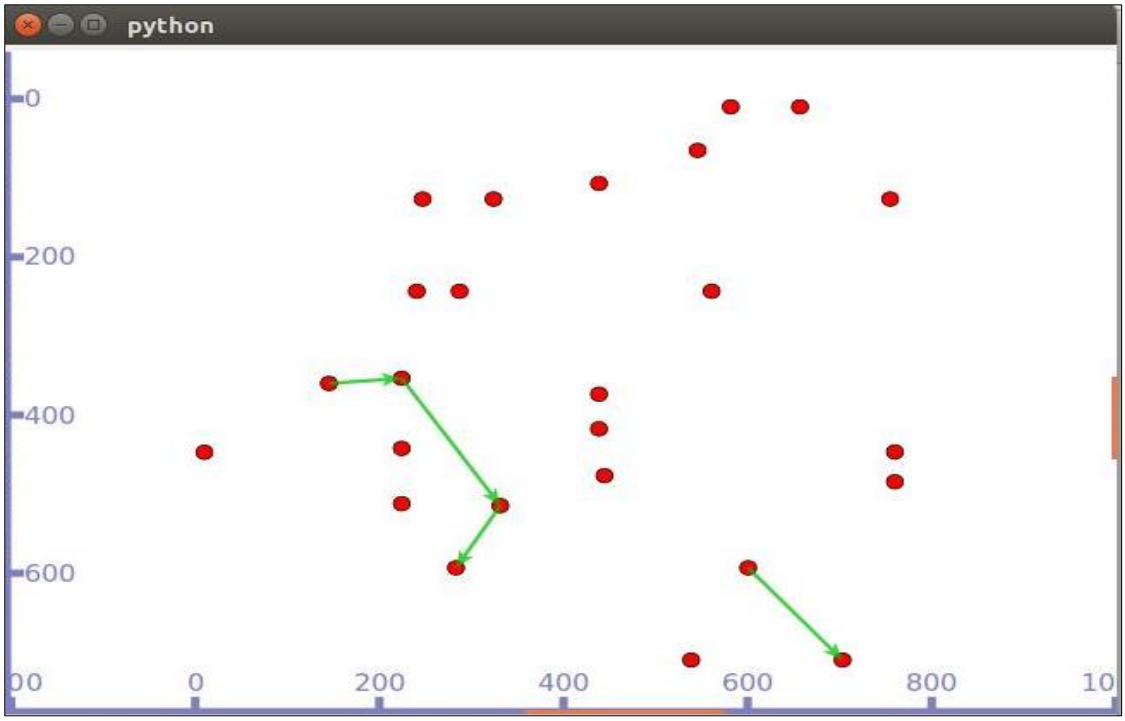


Figura 36 Envío de paquetes escenario de 25 nodos

Fuente: Autor

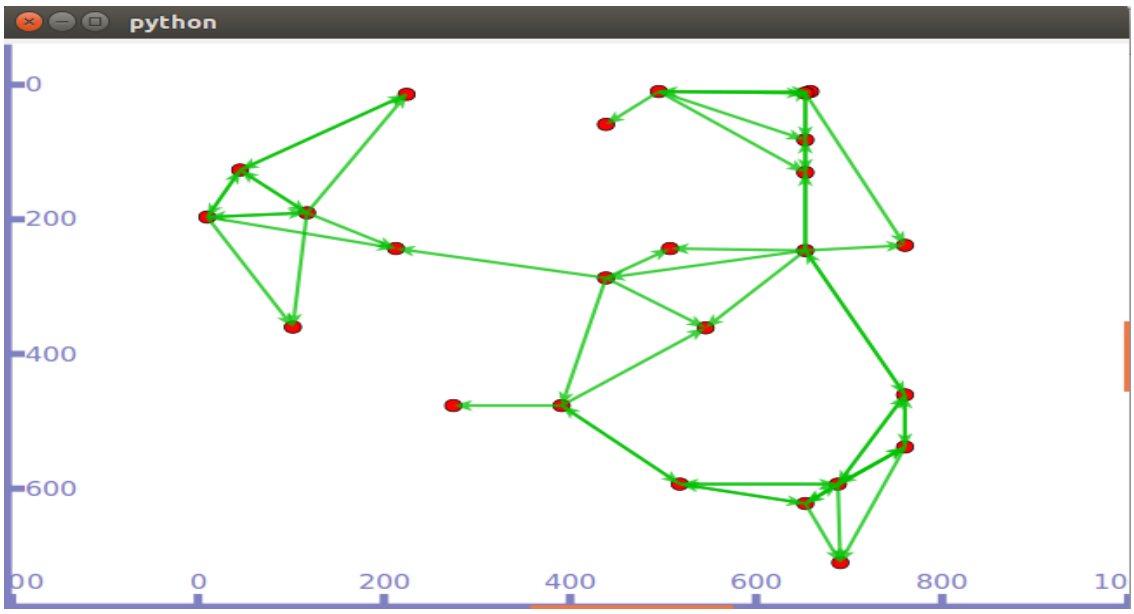


Figura 37 Descubrimiento de red escenario 25 nodos
Fuente: Autor

Una vez realizada la simulación en el primer escenario se obtuvo los valores de las métricas de calidad de servicio (QoS), tanto para los protocolos de encaminamiento AODV y DSR para lo que se ha recopilado los resultados en la tabla que se muestra a continuación.

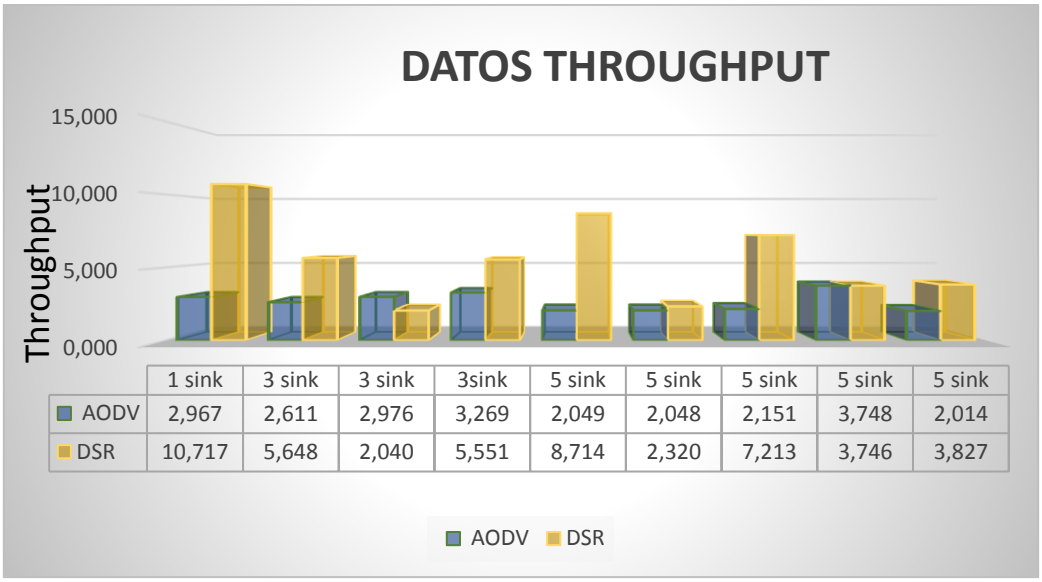
AODV			DSR			
Throughput [Kbps]	**Delay [ms]**	**Jitter [ms]**	**Throughput [Kbps]**	**Delay [ms]**	**Jitter [ms]**	**# Sink [U]**
2,967	249,521	6,502	10,717	337,093	26,646	1
2,611	306,777	5,571	5,648	284,724	11,008	3
2,976	339,048	6,943	2,040	254,800	1,495	3
3,269	339,048	4,772	5,551	253,564	9,513	3
2,049	281,845	3,576	8,714	187,662	16,164	5
2,048	250,018	12,586	2,320	314,705	12,729	5
2,151	277,709	11,936	7,213	264,887	11,427	5
3,748	305,090	15,050	3,746	262,356	16,699	5
2,014	267,901	13,424	3,827	252,356	15,699	5

Tabla 7 Valores métricas Escenario 1

Fuente: Autor

6.2.1.1. Throughput

En la figura 38 se realiza la comparación de los valores de throughput que es la primera métrica que analizamos y se puede observar que en el escenario para 25 nodos del modelo creado en Manhattan Grid, el rendimiento en la entrega de paquetes en AODV tiene una tasa estable, pero para DSR la tasa de entrega de paquetes es mayor lo que significa que el protocolo de encaminamiento DSR tiene mejor rendimiento en un escenario con menor cantidad de nodos ya que las rutas se almacenan en las cabeceras descartando la sobrecarga.



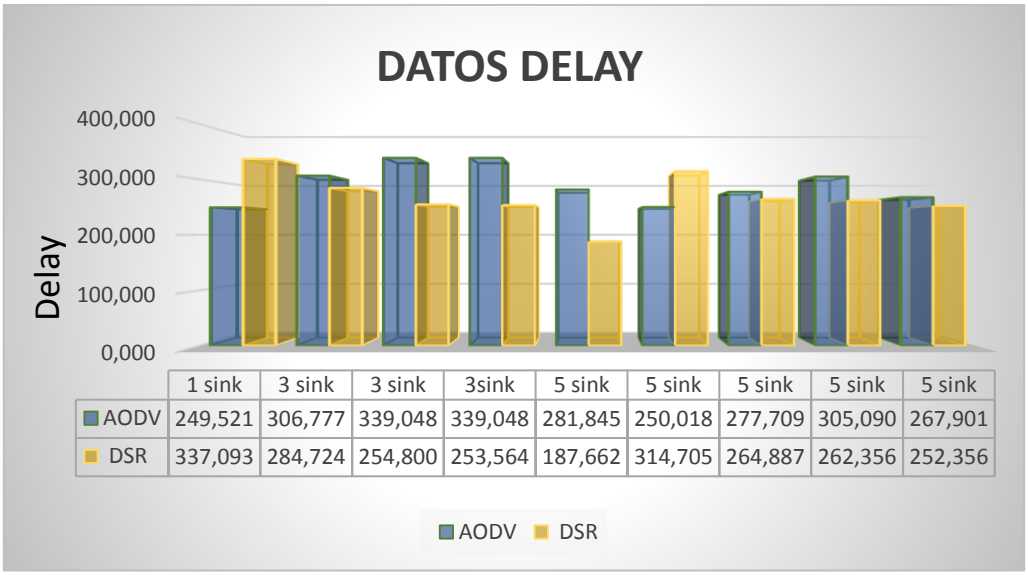
	1 sink	3 sink	3 sink	3sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	2,967	2,611	2,976	3,269	2,049	2,048	2,151	3,748	2,014
DSR	10,717	5,648	2,040	5,551	8,714	2,320	7,213	3,746	3,827

Figura 38 Gráfica Throughput escenario 25 nodos

Fuente: Autor

6.2.1.2. Delay

Para el primer escenario en la figura 39 nos muestra la comparación de la métrica Delay, lo que se puede observar que el protocolo AODV tiene mayor tiempo de retardo en la entrega ya que al haber menor número de nodos en un área mayor existe más retraso que en DSR por la distancia que se encuentran los nodos, dado que AODV debe hacer un descubrimiento constante de rutas.



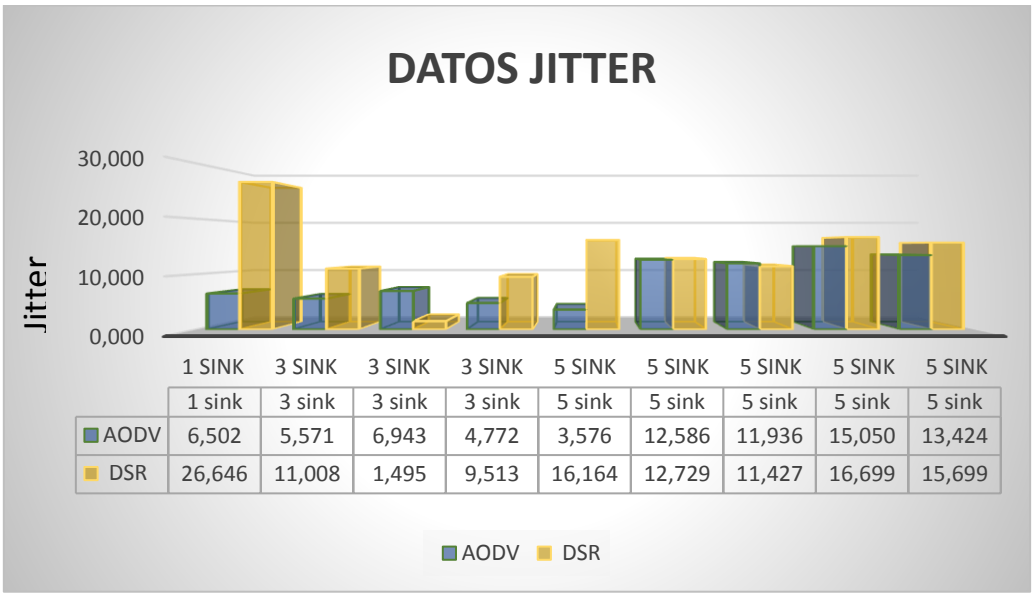
	1 sink	3 sink	3 sink	3sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	249,521	306,777	339,048	339,048	281,845	250,018	277,709	305,090	267,901
DSR	337,093	284,724	254,800	253,564	187,662	314,705	264,887	262,356	252,356

Figura 39 Gráfica Delay escenario 25 nodos

Fuente: Autor

6.2.1.3. Jitter

Ahora se muestra en la figura 40 la comparativa de la métrica que corresponde al Jitter, se puede observar que el protocolo AODV tiene menor variación en el retardo ya que como se mencionó anteriormente este protocolo está en constante descubrimiento de rutas; es decir tiene mayor estabilidad y convergencia que DSR, ya existe nodos intermedios que generan nuevos caminos por lo que hay menor variación de retardo.



	1 SINK	3 SINK	3 SINK	3 SINK	5 SINK	5 SINK	5 SINK	5 SINK	5 SINK
	1 sink	3 sink	3 sink	3 sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	6,502	5,571	6,943	4,772	3,576	12,586	11,936	15,050	13,424
DSR	26,646	11,008	1,495	9,513	16,164	12,729	11,427	16,699	15,699

Figura 40 Gráfica Jitter escenario 25 nodos

Fuente: Autor

6.2.2. Escenario 2

- 50 nodos

También se muestra de la misma manera en las figuras 41 y 42, la distribución de los nodos en el escenario en envío de paquetes y descubrimiento de rutas.

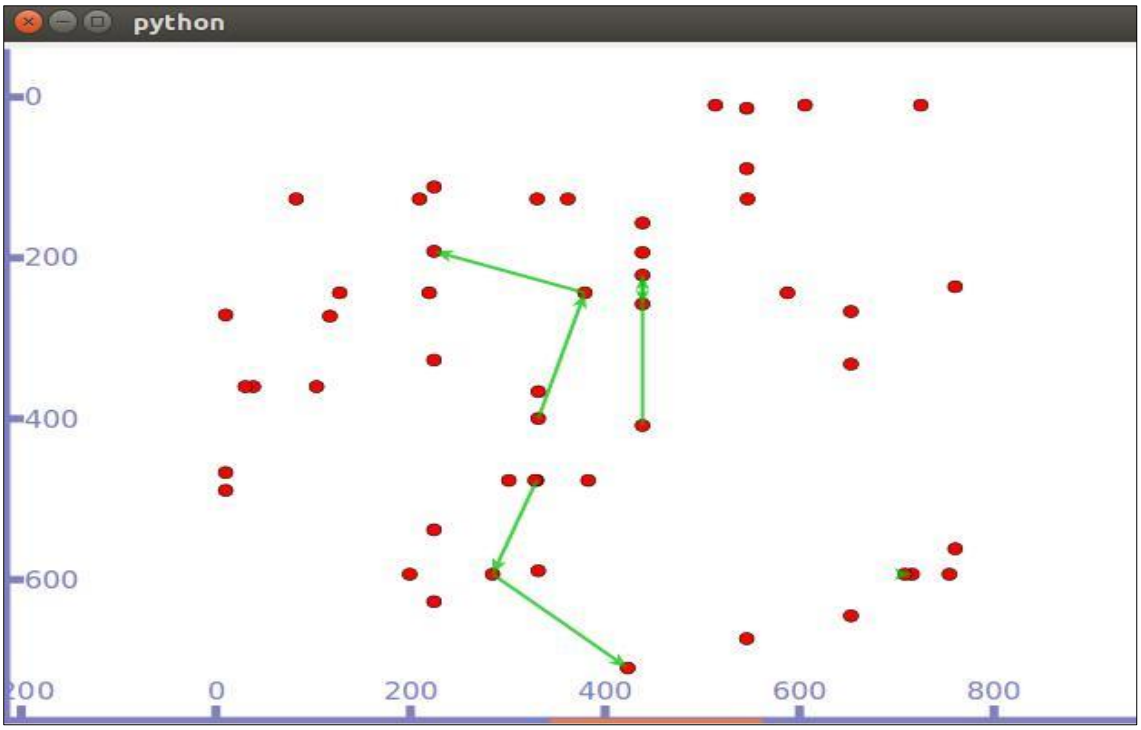


Figura 41 Envío de paquetes escenario de 50 nodos
Fuente: Autor

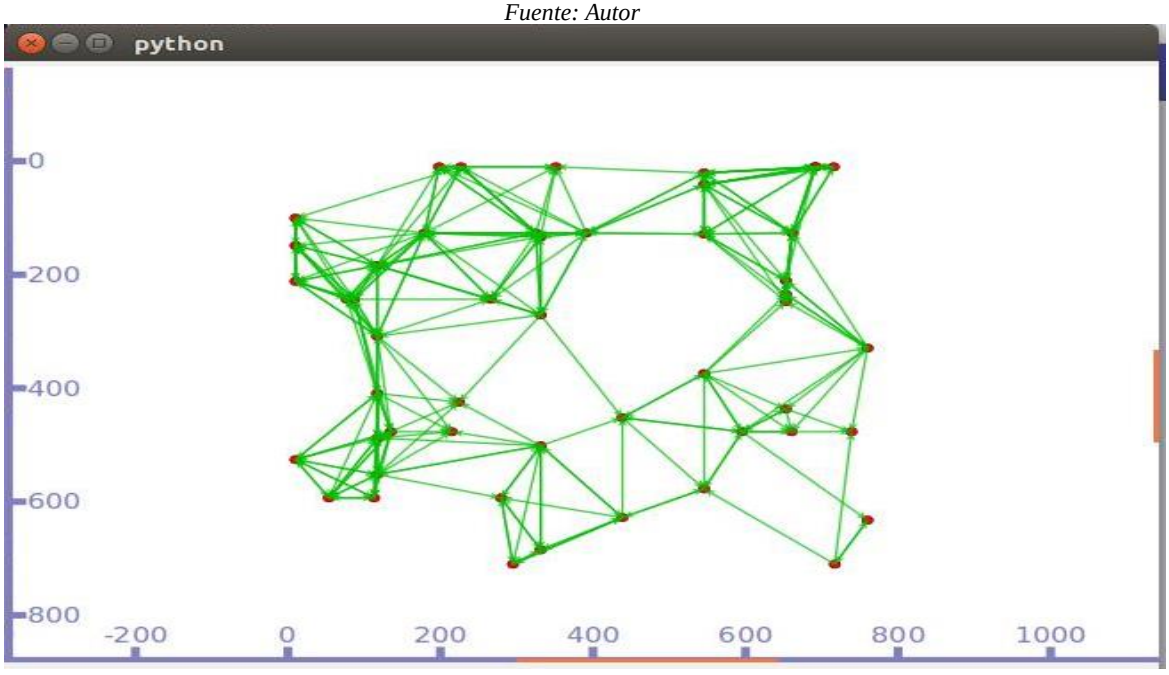


Figura 42 Descubrimiento red escenario 50 nodos
Fuente: Autor

AODV			DSR			
Throughput [Kbps]	**Delay [ms]**	**Jitter [ms]**	**Throughput [Kbps]**	**Delay [ms]**	**Jitter [ms]**	**# Sink [U]**
9,829	285,929	9,353	3,953	319,916	17,888	1
7,910	332,526	12,506	2,048	250,032	4,085	3
12,581	318,867	16,422	2,463	399,580	18,197	3
7,722	329,357	12,534	9,193	347,007	17,631	3
16,422	277,020	5,023	2,106	305,290	9,605	5
6,805	257,476	10,288	4,508	298,047	10,652	5
10,914	251,211	12,060	2,125	262,203	3,182	5
2,025	322,675	12,704	4,974	347,069	13,418	5
10,694	282,350	9,039	14,048	425,442	11,169	5

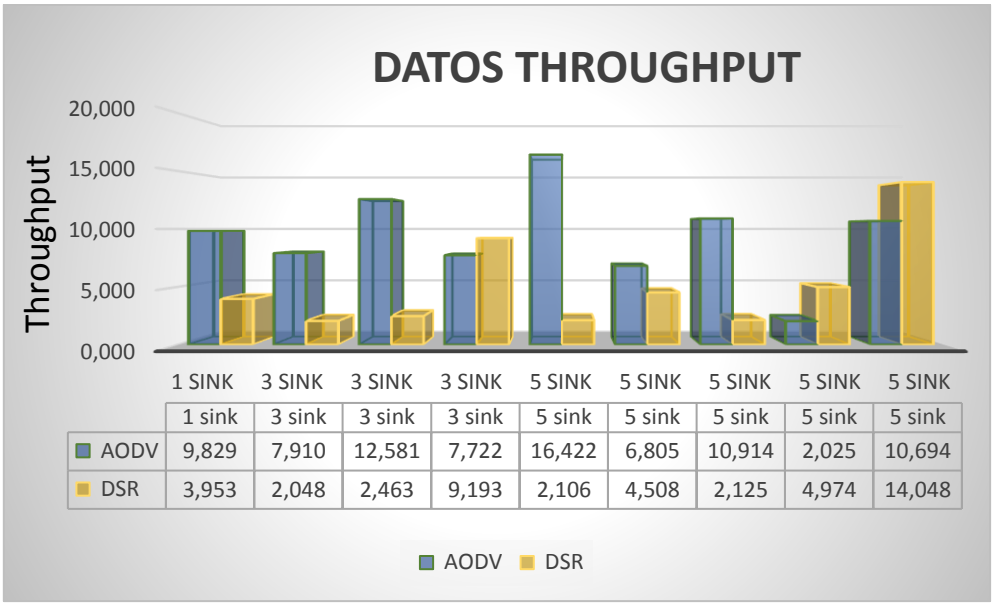
Tabla 8 Valores métricas Escenario

Fuente: Autor

Para el segundo escenario que contiene el mismo ambiente de trabajo entre áreas y velocidades, pero se aumentado el número de nodos a 50, para lo cual se ha elaborado un resumen de los valores obtenidos de las métricas en la simulación como se muestra en la tabla anterior.

6.2.2.1. Throughput

En la figura 43 podemos observar que el throughput del segundo escenario con 50 nodos mejoró el rendimiento para el protocolo AODV, ya que al tener mayor número de nodos o vehículos movilizándose en el área prevista existe menor número de paquetes perdidos a diferencia de DSR que mantiene una ruta desde su origen y al alejarse más los nodos tenemos mayor pérdida de paquetes y por ende decae el rendimiento en la entrega de los paquetes.

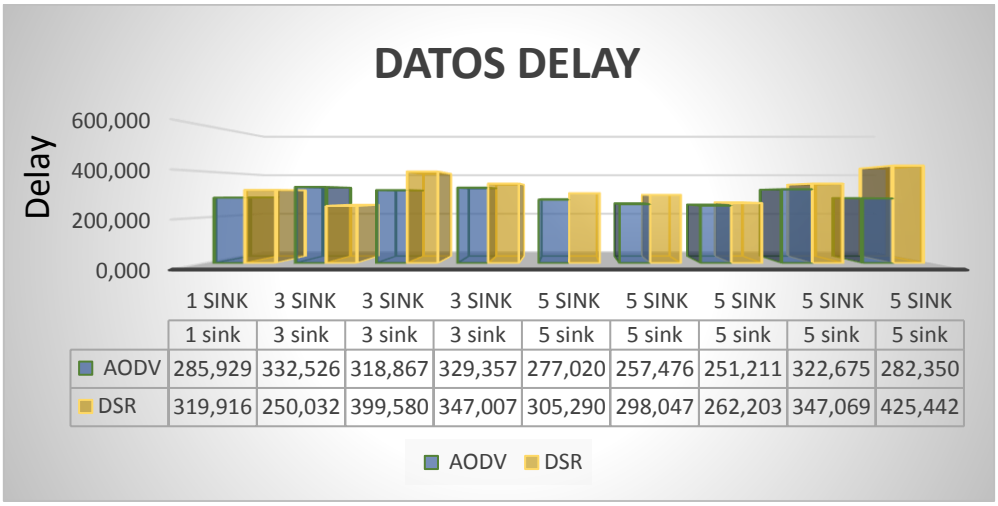


	1 SINK	3 SINK	3 SINK	3 SINK	5 SINK	5 SINK	5 SINK	5 SINK	5 SINK
	1 sink	3 sink	3 sink	3 sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	9,829	7,910	12,581	7,722	16,422	6,805	10,914	2,025	10,694
DSR	3,953	2,048	2,463	9,193	2,106	4,508	2,125	4,974	14,048

Figura 43 Gráfica Throughput escenario 50 nodos
Fuente: Autor

6.2.2.2. Delay

Para el segundo escenario la comparativa del Delay o retardo en la entrega de paquetes para el protocolo DSR existe una cierta variación en el retraso con respecto al protocolo AODV, aunque se puede observar que esta diferencia no es tan significativa se la puede atribuir a que DSR mantiene la ruta desde el nodo emisor y al existir más nodos movilizándose por todo el escenario se pierde la comunicación lo que provoca que se vuelva a realizar el descubrimiento desde el nodo de origen.

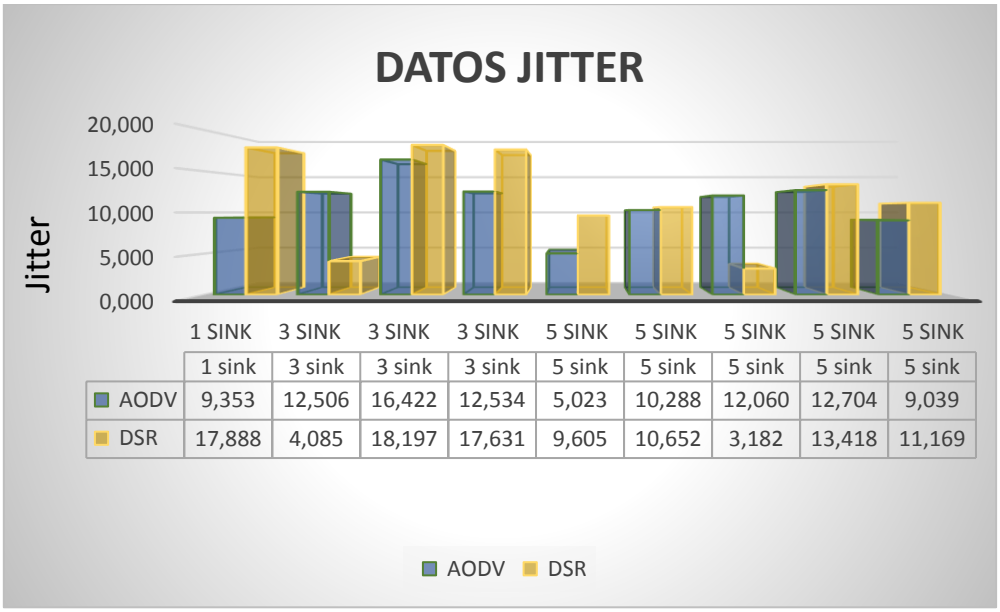


	1 SINK	3 SINK	3 SINK	3 SINK	5 SINK	5 SINK	5 SINK	5 SINK	5 SINK
	1 sink	3 sink	3 sink	3 sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	285,929	332,526	318,867	329,357	277,020	257,476	251,211	322,675	282,350
DSR	319,916	250,032	399,580	347,007	305,290	298,047	262,203	347,069	425,442

Figura 44 Gráfica Delay escenario 50 nodos
Fuente: Autor

6.2.2.3. Jitter

Para la figura 45 de la tercera métrica en el escenario que consta de 50 nodos se puede observar que el protocolo AODV es más estable y tiene mayor grado de convergencia, por tener un valor de Jitter menor con respecto a DSR, al existir mayor número de densidad de nodos en el escenario propuesto y al mantener una sola ruta desde el nodo origen genera una sobrecarga, además de tener un tráfico elevado por lo que obtenemos un porcentaje mayor en la variación de retardo en DSR.



	1 sink	3 sink	3 sink	3 sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	9,353	12,506	16,422	12,534	5,023	10,288	12,060	12,704	9,039
DSR	17,888	4,085	18,197	17,631	9,605	10,652	3,182	13,418	11,169

Figura 45 Gráfica Jitter escenario 50 nodos

Fuente: Autor

6.2.3. Escenario 3

- 100 nodos

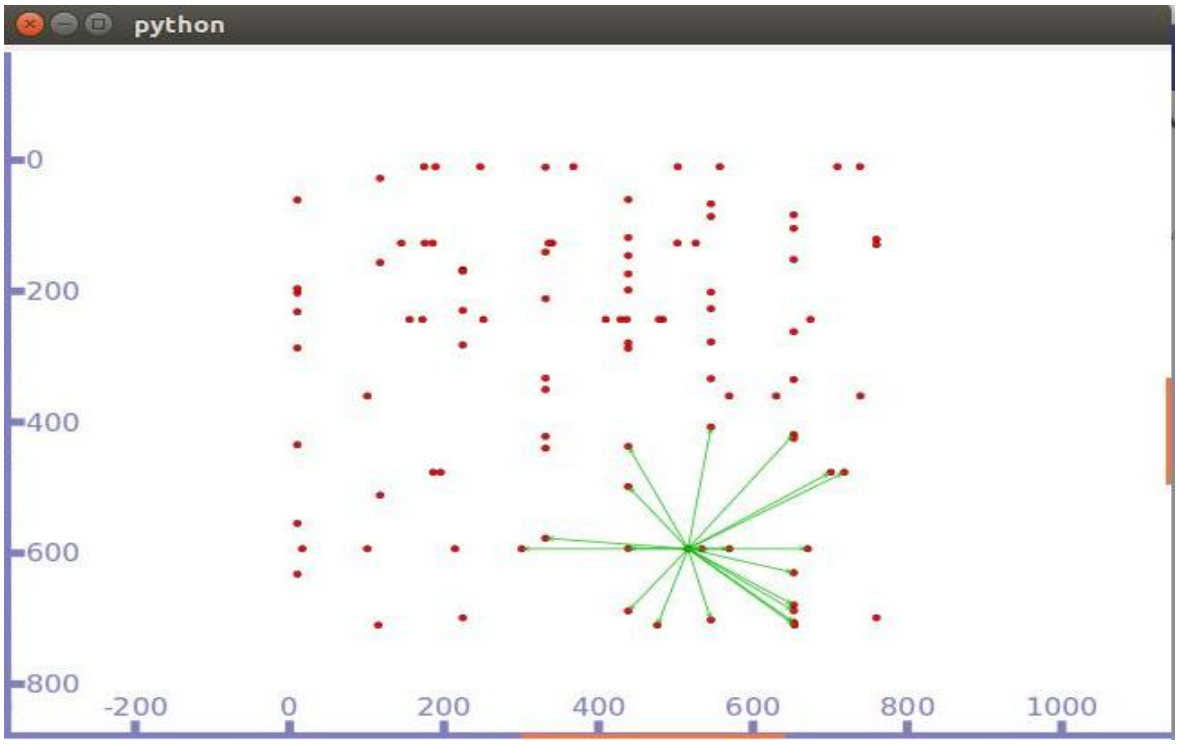


Figura 46 Envío de paquetes escenario de 100 nodos
Fuente: Autor

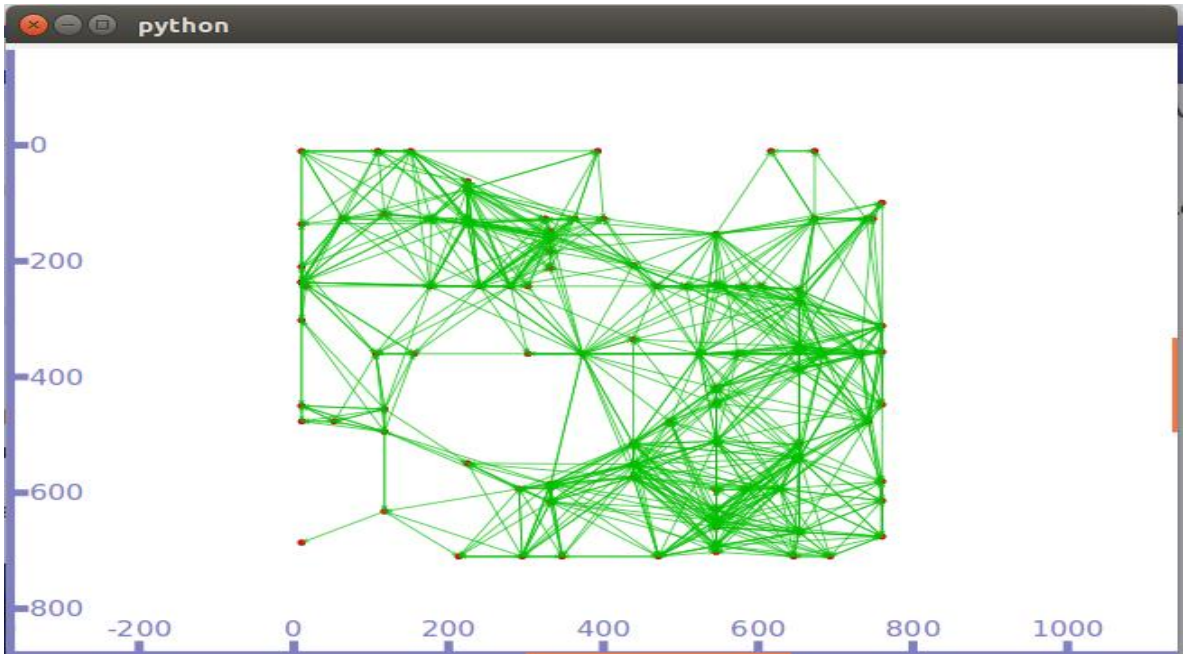


Figura 47 Descubrimiento de la red escenario 100 nodos
Fuente: Autor

Para el tercer escenario que se ha propuesto en el presente trabajo se ha elevado el número de nodos a la red, con lo que se ha obtenido un resumen de los valores de las métricas calculadas en la simulación se presentan a continuación.

AODV			DSR			
Throughput [Kbps]	**Delay [ms]**	**Jitter [ms]**	**Throughput [Kbps]**	**Delay [ms]**	**Jitter [ms]**	**# Sink [U]**
3,729	277,113	3,720	2,006	341,426	11,303	1
7,476	261,766	11,329	5,986	303,297	7,665	3
8,975	402,091	25,733	10,438	230,563	8,819	3
11,877	377,385	21,952	3,827	288,663	15,534	3
8,086	297,557	17,834	3,854	321,979	9,179	5
8,348	340,520	16,680	4,298	338,683	17,163	5
9,144	343,540	16,155	4,986	373,414	18,472	5
8,370	230,082	20,969	8,350	356,474	22,989	5
4,602	239,780	20,513	10,294	256,262	10,199	5

Tabla 9 Valores métricas Escenario 3

Fuente: Autor

6.2.3.1. Throughput

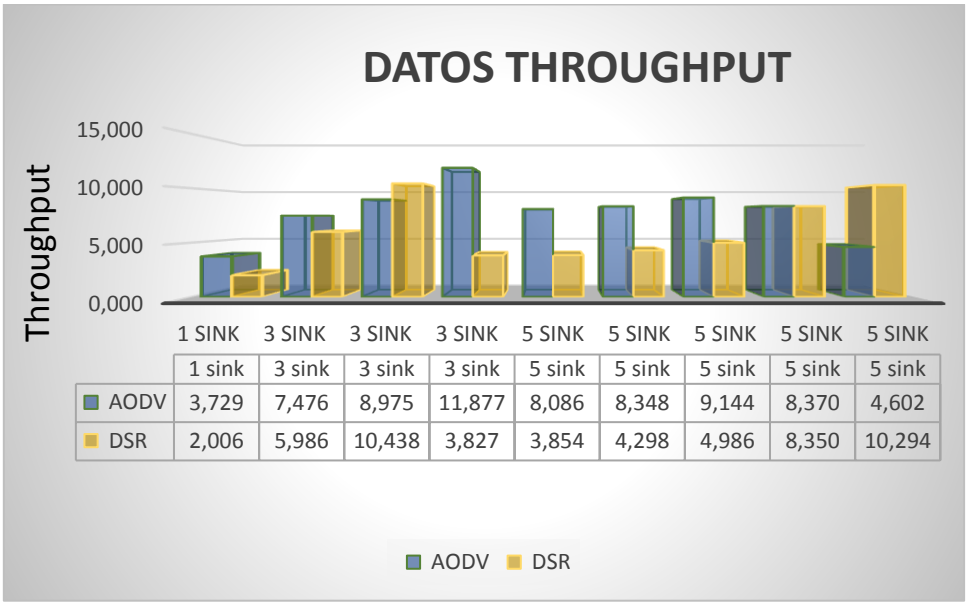


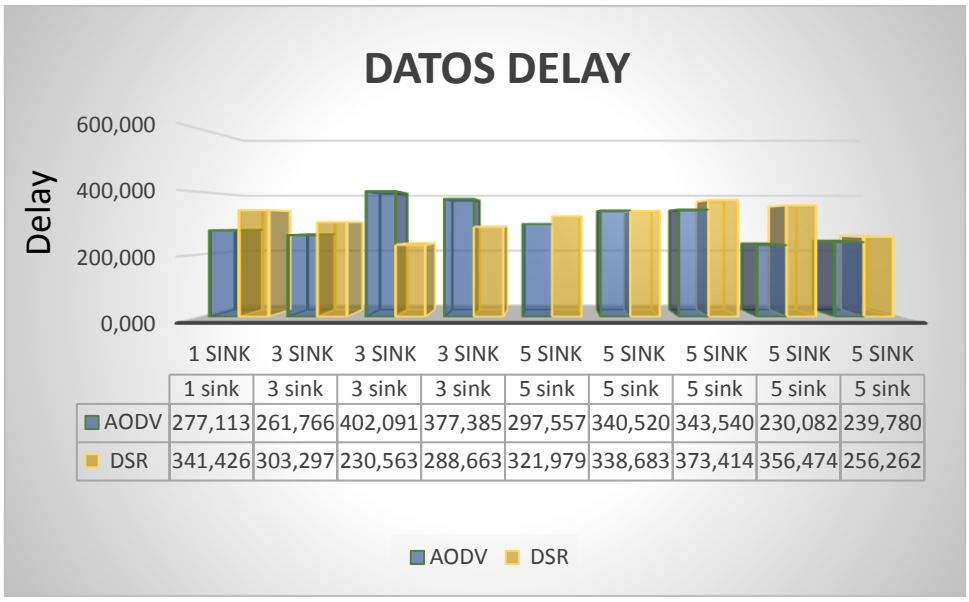
Figura 48 Gráfica Throughput escenario 100 nodos

Fuente: Autor

En la figura 48 se puede observar los resultados de la simulación del throughput para el tercer escenario con una densidad de 100 nodos distribuidos en el área seleccionada, en los valores calculados en esta métrica se muestra un mejor rendimiento en el protocolo AODV, lo que nos indica que al aumentar el número de nodos AODV reacciona de manera más eficiente ya que al tener mayor número de rutas descubierta se vuelve más estable, cabe indicar que el protocolo DSR empieza a decaer al aumentar el número de nodos o conexiones debido a que necesita ubicar en la cabecera los saltos necesarios entre el origen y destino para lo que necesita de un determinado tiempo, lo que ocasiona que se desborde el buffer de cada nodo ocasionando perdida de paquetes.

6.2.3.2. Delay

En la figura 49 se puede observar los resultados de los valores recopilados para la segunda métrica en este caso el Delay o retardo en la entrega de paquetes, en este escenario con 100 nodos se muestra que los protocolos AODV y DSR se mantienen con valores similares, siendo el aumento en DSR más pronunciado en ciertas mediciones debido a que almacena toda la ruta del paquete en la cabecera, lo que podría generar sobrecarga y retrasos en la red.



	1 sink	3 sink	3 sink	3 sink	5 sink	5 sink	5 sink	5 sink	5 sink
AODV	277,113	261,766	402,091	377,385	297,557	340,520	343,540	230,082	239,780
DSR	341,426	303,297	230,563	288,663	321,979	338,683	373,414	356,474	256,262

Figura 49 Gráfica Delay escenario 50 nodos

Fuente: Autor

6.2.3.3. Jitter

En la figura 50 que muestra los resultados del Jitter, se puede observar que AODV en este escenario tiene en promedio aumento en la variación del retardo y es algo comprensible ya que al generar más caminos se genera mayor tráfico y por lo tanto incrementan el número de colisiones, lo que vuelve a este protocolo inestable y con poca convergencia, este valor depende mucho de número de nodos ubicados en el escenario y de la distancia que estos podrían tomar en el área a simular.

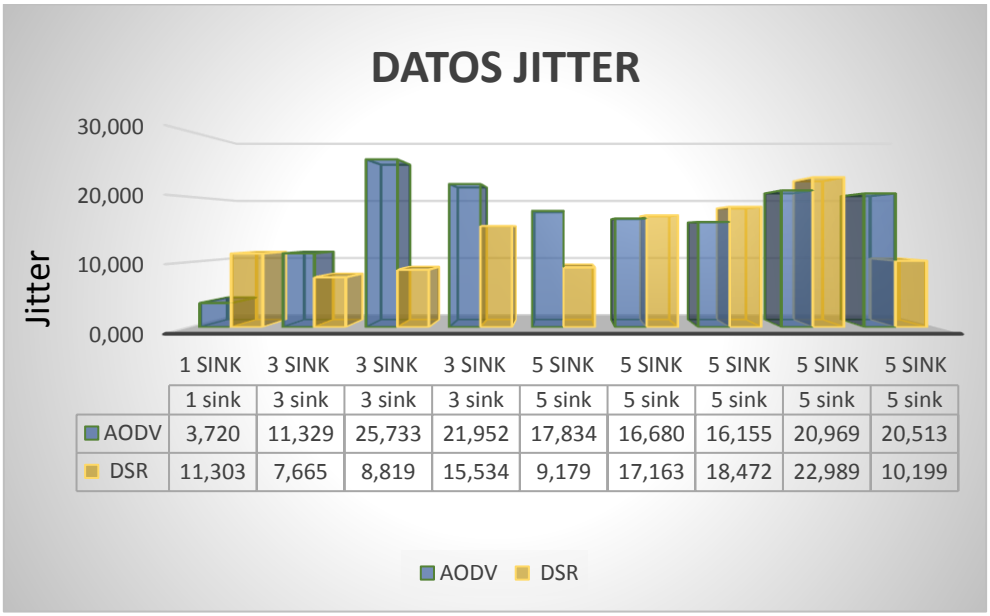


Figura 50 Gráfica Jitter escenario 100 nodos

Fuente: Autor

6.3. Comparación de resultados

6.3.1. Throughput

A continuación, se presenta un resumen y comparación general de los resultados en los diferentes escenarios, en la tabla 10 podemos observar los valores de Throughput tanto para DSR como para AODV con distintas densidades de nodos.

# Nodos	AODV	DSR
25	2,64812064	5,53061577
50	9,83340845	5,04646567
100	9,8453409	6,00439856

Tabla 10 Comparación resultados Throughput

Fuente: Autor

De acuerdo con los resultados obtenidos en la tabla anterior se puede indicar que el protocolo DSR tiene un mejor rendimiento en los escenarios que contienen menor cantidad de nodos, a medida que se agregan más nodos al escenario el protocolo AODV aumenta la métrica de Throughput como se puede observar en la figura 51.

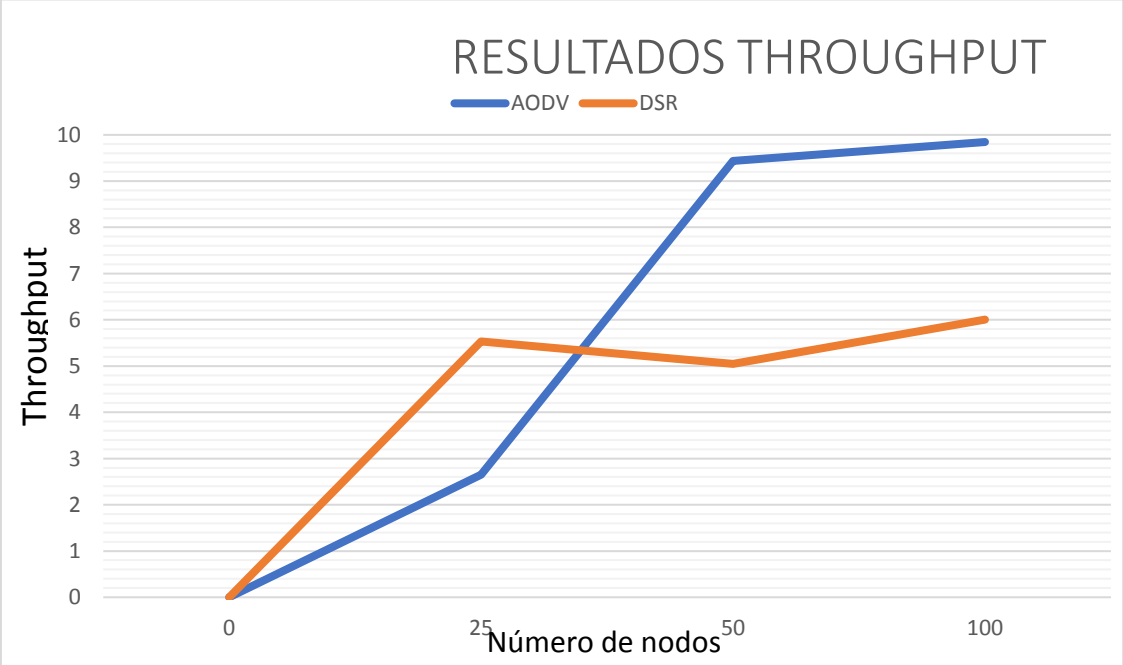


Figura 51 Gráfica resultados Throughput

Fuente: Autor

6.3.2. Delay

En la tabla 11 podemos observar los valores de la métrica Delay tanto para DSR como para AODV con distintas densidades de nodos.

# Nodos	AODV	DSR
25	290,772823	268,016313
50	295,267837	328,28725
100	307,759292	312,306818

Tabla 11 Comparación resultados Delay
Fuente: Autor

Los resultados de la tabla anterior nos indican que el Delay se mantiene en general para los dos protocolos de encaminamiento elegidos, lo que significa que al ser ambos protocolos reactivos obtiene su ruta bajo demanda por lo que no experimentan retrasos pronunciados entre ellos, se puede observar en la figura 52 que a medida que se aumenta nodos a los escenarios aumenta también el Delay.

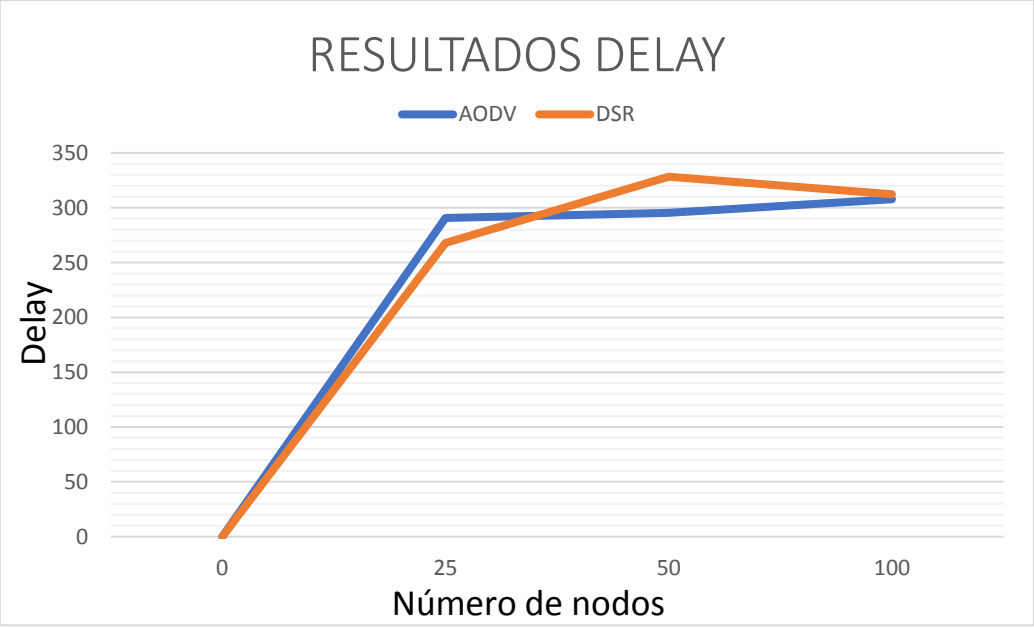


Figura 52 Gráfica resultados Delay
Fuente: Autor

6.3.3. Jitter

En la tabla 12 podemos observar los valores de los resultados generales para métrica Jitter, para DSR como para AODV con distintas densidades de nodos.

# Nodos	AODV	DSR
25	8,92879862	13,487
50	11,103	11,758746
100	17,209	13,4804167

Tabla 12 Comparación resultados Jitter
Fuente: Autor

Se obtuvieron los resultados para el Jitter en las simulaciones realizadas como se muestra en la tabla anterior. Ahora se puede observar que esta métrica tiene valores parecidos en cada protocolo, la variación de retraso va en aumento a medida que se agregan nodos a la red, en la figura 53 muestra los resultados en los diferentes escenarios.

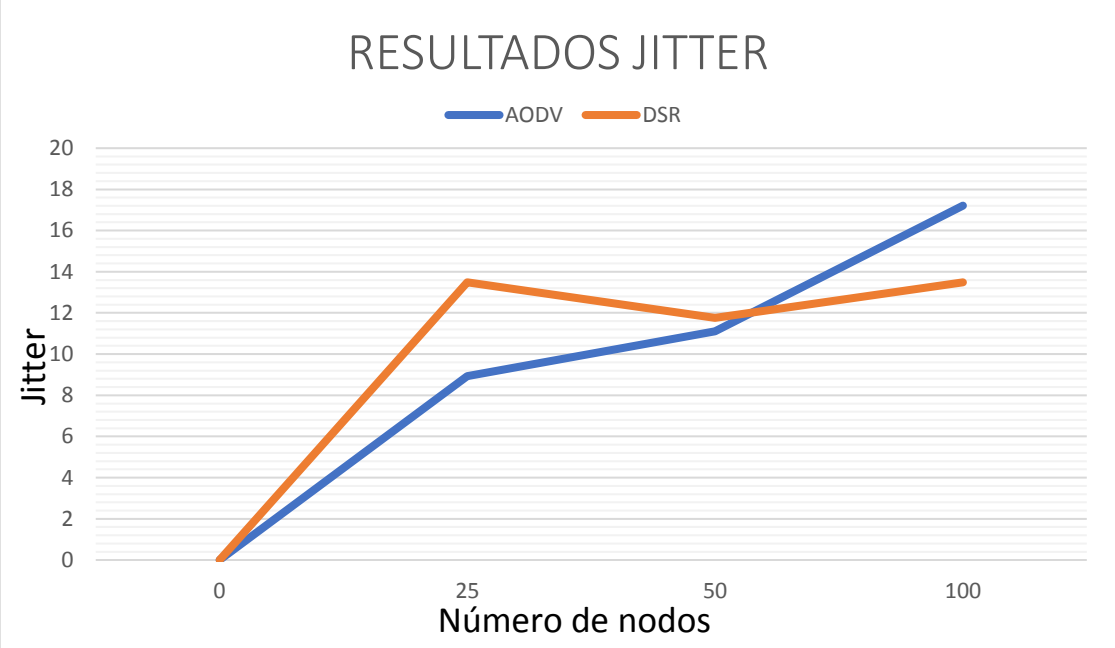


Figura 53 Gráfica resultados Jitter
Fuente: Autor

7. DISCUSIÓN

En el presente trabajo se ha tratado sobre el análisis y comparativa de los protocolos de encaminamiento AODV y DSR sobre redes VANETs, donde su parte medular se ha centrado en recopilar los valores de las métricas de calidad de servicio (QoS) como son: Throughput (Rendimiento), Delay (Retardo) y Jitter (Variación de Retardo) de la misma manera se ha dispuesto de tres escenarios con diferente densidad de nodos (vehículos).

Con los resultados obtenidos se ha podido observar que en escenarios con menor cantidad de nodos el protocolo DSR tiene mejor desenvolvimiento en cada una de las métricas calculadas ya que al contar con menor número de nodos disminuye la sobrecarga en la cabecera del protocolo, a medida que se aumenta los nodos en los escenarios elegidos los valores del Throughput experimentan una pequeña atenuación en DSR a diferencia del protocolo AODV que es más eficiente en este tipo de escenarios debido a que se mantiene en constante descubrimiento de rutas por lo que existe menor perdida de paquetes ya que escoge el mejor camino de envío.

De acuerdo con los estudios y la teoría prevista en la realización del presente trabajo se ha podido corroborar algunos análisis sobre el encaminamiento de protocolos AODV y DSR en redes VANETs y además diferenciar algunos resultados ya que de acuerdo con nuestro estudio los protocolos se han desempeñado de distinta forma de acuerdo al escenario escogido, (Saadi, Kafhali, Haqiq, & Nassereddine, 2016) sostiene en sus resultados que independientemente del tamaño de la red o la tasa de movilidad, AODV y DSR se desempeñaron de igual manera y entregaron más del 90% de los paquetes de datos, los mismo que tiene un tamaño de 512 bits y el número de paquetes enviados corresponde a 4 paquetes por envío, lo que comparado con nuestros resultados, la eficiencia de cada uno de los protocolos depende del área, la densidad de nodos (vehículos) y el modelo de movilidad en donde se despliegue la red, con lo cual se puede decir que AODV tiene mejor Throughput que DSR en dos de los tres escenarios propuestos.

Se presentaron algunas limitaciones al momento de utilizar los métodos para la extracción de la métricas de calidad de servicio (QoS) en los protocolos de encaminamiento, específicamente estos problemas se dieron en la recolección de datos para DSR, ya que agrega el envío de paquetes en el encabezado lo que dificulta hacer el seguimiento de datos por lo que se decidió utilizar las fórmulas propuestas para el cálculo del Throughput, Delay, Jitter y así recuperar los valores necesarios para realizar la comparativa y análisis de los respectivos protocolos en estudio.

De acuerdo con el estudio realizado se propone como trabajo a futuro el análisis e investigación del protocolo SAODV (Secure Ad-Hoc On-Demand Distance Vector). Ya que a pesar de la estabilidad que demuestra el protocolo AODV en los diferentes escenarios de esta investigación, presenta problemas de seguridad en cuanto a intrusiones, por lo que sería relevante el estudio de la implementación de seguridad en este protocolo de encaminamiento reactivo.

8. CONCLUSIONES

- Para poder cumplir con los objetivos planteados es de suma importancia la utilización del software NS-3, ya que es un simulador de red completo y está compuesto por diferentes módulos, además contiene información guía de las redes implementadas y puntualmente sobre los protocolos aquí analizados.

- Para la simulación de los diferentes escenarios de movilidad en redes VANETs, NS-3 permite agregar tecnologías de transmisión, distintos protocolos de comunicación, modelos de movilidad, además de poder importar trazas de movilidad desde las diferentes herramientas compatibles con el software, como por ejemplo Bonnmotion herramienta necesaria para redes VANETs.

- Redes VANETs representan un sinnúmero de utilidades en cuanto a investigación y aplicación en los Sistemas Inteligentes de Transportes (ITS), mejoran el control de tráfico vehicular, conocimiento de las vías por parte de los conductores, reducir el alto índice de accidentes de tránsito y mejorar considerablemente la fluidez vehicular.

- En el estudio del modelo movilidad Manhattan Grid, los protocolos AODV y DSR muestran valores similares en los tres escenarios propuestos, siendo DSR el protocolo con mejor calidad de servicio en escenarios que tienen menor densidad de nodos a diferencia de AODV que es más eficiente cuando se agregan más nodos a la red, lo que se debe a las características de descubrimiento constante de rutas, además cabe indicar que AODV tiene mayor retardo de extremo a extremo ya que al existir más colisiones se debe establecer nuevas rutas periódicamente, a diferencia de lo que sucede en DSR que mantiene un ruta desde la cabecera del nodo de origen hacia el destino.

- Según lo dicta uno de los objetivos planteados en el presente trabajo, se ha generado con éxito los escenarios para el modelo de movilidad Manhattan Grid mediante el uso de la herramienta Bonnmotion, para lo que se ha creado archivos que son normalmente compatibles con el software NS-2 pero que con ayuda de diferentes líneas de código se pudo importar hacia NS-3 que es el software base con lo que hemos ejecutado el presente trabajo.

- De acuerdo a los datos obtenidos en las simulaciones para los tres escenarios, cuando el protocolo de encaminamiento DSR realiza un envío debe almacenar la ruta de destino y colocarla en la cabecera del paquete, al contar con escenarios que experimentan aumento de nodos se genera sobrecarga, ya que al tener rutas más largas el tamaño del encabezado y de las tablas de ruteo tienen un crecimiento considerable lo que causa perdida de paquetes y se experimenta reducción en la calidad de servicio (QoS).

- Basándonos en el análisis y comparación de las diferentes métricas de calidad de servicio (QoS) para cada uno de los escenarios y modelos propuestos se puede concluir que el protocolo de encaminamiento AODV tiene un mejor rendimiento en los ambientes con alta densidad de nodos, a diferencia que el protocolo DSR es estable en escenarios más pequeños, por tal motivo el protocolo a tener en cuenta para futuras pruebas en el campo debe de ser AODV por temas de escalabilidad, crecimiento de la ciudad, mayor tráfico y aumento de aplicaciones.

9. RECOMENDACIONES

- Se recomienda que para realizar el estudio y análisis de los diferentes protocolos de encaminamiento se debe tener una base de conocimientos en lo que tiene que ver con temas de RFC de cada protocolo a analizar, redes de computadoras, lenguaje de programación C++, software libre, etc.

- Para evitar problemas al momento de simular los protocolos AODV y DSR en redes VANETs se recomienda utilizar versiones estables de los softwares como Ubuntu 16.04 y NS-3, versión 3.28 ya que cuentan con la mayoría de herramientas documentadas y en funcionamiento.

- Se debe poner especial atención al momento de obtener los valores de las métricas de calidad de servicio en los protocolos ya que los diferentes métodos que existen para extraerlas no funcionan de la misma manera en AODV como en DSR debido a la forma que tiene cada uno de enviar paquetes en la red, lo que genera problemas en la obtención de los datos.

- Al elegir modelos de movilidad para la simulación en redes VANETs se debe tener en cuenta que estos se encuentren o tengan librerías asociadas al software NS-3, o en su defecto que puedan ser creadas con la ayuda de herramientas externas y así realizar las comparativas sin contratiempos.

- Se debe tomar en cuenta el número de nodos utilizados en los diferentes escenarios de simulación, de acuerdo con las características y aplicaciones utilizadas en NS-3, se requiere equipos con mayor procesamiento tanto en hardware como en software cada ves que aumenta la densidad de nodos.

10. REFERENCIAS

Andrade Tenén, M. J. (2017). *Estudio y simulación de redes heterogéneas vanet, con la variación de tecnología inalámbrica según la capacidad de usuarios en zonas urbanas de varios sectores de la ciudad* (No. 1–9). Universidad Politecnica Salesiana, Quito.

Boussoufa-lahlah, S., Semchedine, F., & Bouallouche-medjkoune, L. (2018). Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A survey. *Vehicular Communications*, *11*, 20–31. https://doi.org/10.1016/j.vehcom.2018.01.006

Caballero-gil, P., & Molina-gil, J. (2014). Gestión de Grupos en VANETs : Descripción de Fases, (September).

Campos, M. G. (2016). *Evaluación de Protocolos de Encaminamiento Para Redes Vehiculares (VANET)*. Recuperado de http://bibing.us.es/proyectos/abreproy/12216/fichero/PFC-Evaluacion+de+protocolos+de+encaminamientos+para+redes+vehiculares+%28VANET%29+-+Jose+Manuel+Garcia+Campos.pdf

Carrión, M. (2016). *Analísis de la simulación de la diseminación de mensajes de emergencia en redes Vehiculares Ad-Hoc Mediante software libre*. Universidad Politecnica Salesiana de Cuenca.

Chirinos, S. (2013). *Evaluación de los protocolos de enrutamiento AODV y OLSR en redes VANET*. Recuperado de https://repository.javeriana.edu.co/bitstream/handle/10554/13616/ChirinosCadavidSantiago2013.pdf?sequence=1&isAllowed=y

Cunha, F., Villas, L., Boukerche, A., Maia, G., Viana, A., Mini, R. A. F., & Loureiro, A. A. F. (2016). Data communication in VANETs : Survey , applications and challenges. *Ad Hoc Networks*. https://doi.org/10.1016/j.adhoc.2016.02.017

Curay Cuastumal, J. F. (2016). *SIMULACION DE ESCENARIOS DE COMUNICACIÓN INALÁMBRICA ENTRE VEHÍCULOS EN EL SUR DE QUITO POR MEDIO DE PROTOCOLOS DE ENRUTAMIENTO BASADA EN TECNOLOGÍA VANET*. Quito.

Eze, E. C., Zhang, S., Liu, E., & Eze, J. C. (2016). Advances in Vehicular Ad-Hoc Networks (VANETs): Challenges and Road-map for Future Development 2 Overview of VANETs.

Fabián Sánchez Marín, A., & Adriana Pineda Samacá, R. (2016). Evaluación De Redes Vanet Orientada Al Tráfico Vehicular En La Ciudad De Bogotá. Evaluation of Networks Vanet Vehicular Oriented Traffic in the City of Bogota.

Frodigh, M., & Larsson, P. (2000). *Formación de redes inalámbricas ad hoc-El arte de la formación de redes sin red*. *Ericsson Review*. Recuperado de https://s3.amazonaws.com/academia.edu.documents/44827736/ericcson.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1532726663&Signature=g74XUO9ViTvwL6fUcC6jewyfGtQ%3D&response-content-

disposition=inline%3B filename%3DEriccsonh.pdf

Fujiwara, A. (2017). State space reduction techniques for model cheking of MANET protocolos, *7*(1), 29–49.

Gómez, P. A., Posada, G. A., & Vallejo, M. A. (2014). *Evaluación del desempeño del protocolo de enrutamiento AODV para diferentes escenarios de redes de sensores inalámbricos*. https://doi.org/10.14482/inde.32.1.4560

Hechavarria, L. (2016). *Impacto del tamaño de la ventana de contención (CW) del protocolo IEEE 802.11p en el desempeño de las Redes Ad-Hoc Vehiculares*. Universidad Central " Marta Abreu " de Las Villas.

Hirai, T., & Murase, T. (2018). Communication Method to improve QoS in V2X Communications in Crash Warning Application, *1*(5), 1–6. https://doi.org/10.1587/comex.2018XBL0095

Johnson, D., Hu, Y., & Maltz, D. (2007). *The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4*. Recuperado de https://www.rfc-editor.org/pdfrfc/rfc4728.txt.pdf

Lares, L. I., Díaz-ramírez, A., Hernandez, H. S., & Quintero, V. (2015). Evaluacion de protocolos de encaminamiento para redes vehiculares Ad hoc. *Congreso Internacional de Robótica y Computación (CIRC 2015)*, 26–31.

Maldonado Narvaez, V. (2012). *Comparación de protocolos de enrutamiento y modelos de movilidad para Redes Ad-Hoc Vehiculares usando mapas reales*. Universidad Tecnica Particular de Loja.

Marquez, N. P. L. (2014). Performance Evaluation of Bandwidth- Aware Routing Protocol in Urban Scenarios for Vehicular Ad hoc NETworks, 141.

Massobrio, R., & Nesmachnow, S. (2016). *Posicionamiento de infraestructura para redes vehiculares*. Recuperado de https://docplayer.es/53547351-Posicionamiento-de-infraestructura-para-redes-vehiculares.html

Matsumoto, K. (2018). *Data Uploading Control Method Based on Exchange of Metadata in VANET for On-demand Onboard Camera Picture Sharing System* (Vol. 26). https://doi.org/10.2197/ipsjjip.26.2

Maygua, L. (2017). *Simulación de un mecanismo de control de potencia en AODV en función del número de vecinos usando cross layering en una MANET*. Escuela Politecnica Nacional.

NS-3. (2011). ns-3 | a discrete-event network simulator for internet systems. Recuperado el 26 de enero de 2019, de https://www.nsnam.org/

Olariu, S., & Weigle, M. A. C. (2009). *Vehicular networks : from theory to practice*. CRC Press.

Palacios Morocho, M. E. (2017). *Implementación de un simulador de redes mediante software libre para el laboratorio de Telecomunicaciones del AEIRNNR*. Universidad Nacional de Loja. Recuperado de http://dspace.unl.edu.ec/jspui/bitstream/123456789/19778/1/Palacios

Morocho%2C Maritza Elizabeth.pdf

Perez, C. (2017). *Implementación del protocolo de encaminamiento*. Universidad Politecnica de Cataluña.

Perkins, C., Belding-Royer, E., & Das, S. (2003). *Ad hoc On-Demand Distance Vector (AODV) Routing Status*. Santa Barbara. Recuperado de https://www.rfc-editor.org/pdfrfc/rfc3561.txt.pdf

Piles, J. J. (2016). Uso de rutas cacheadas en el encaminamiento seguro basado en DSR.

Portilla Chaves, M., Roja Zapata, A., & Hernandez Arteaga, I. (2014). *Investigación Cualitativa: Una reflexión desde la educacion como hecho social* (Vol. 3).

Rodríguez, G. (2015). *" Análisis y Simulación de protocolos de enrutamiento adecuados en diferentes escenarios para redes AdHoc , mediante la herramienta Ns-3 "*. Universidad Nacional de Loja.

Rondinone, M. (2017). Enrutamiento Basado en Conectividad Multi-hop en Redes Ad-hoc Vehiculares.

Saadi, Y., Kafhali, S. El, Haqiq, A., & Nassereddine, B. (2016). *Simulation Analysis of Routing Protocols using Manhattan Grid Mobility Model in MANET* (Vol. 45).

Sánchez, A. J. (2017). *Redes Vehiculares Aplicadas a la Movilidad Inteligente y Sostenibilidad Ambiental en Entornos de Ciudades Inteligentes*. Recuperado de http://digibuo.uniovi.es/dspace/bitstream/10651/45013/1/TD_JoseAntonioSanchez.pdf

Sarasti, O. O., & Llano Ramírez, G. (2014). APLICACIONES PARA REDES VANET ENFOCADAS EN LA SOSTENIBILIDAD AMBIENTAL, UNA REVISIÓN SISTEMÁTICA, *24–2*. Recuperado de http://www.scielo.org.co/pdf/cein/v24n2/v24n2a07.pdf

Torres, A. (2010). *ANÁLISIS DE LA CALIDAD DE SERVICIO EN EL ENRUTAMIENTO DE LAS REDES MÓVILES AD-HOC*. Universidad Tecnica Particular de Loja.

Torres, J. I. (2018). *Implementación del protocolo de enrutamiento GPRS para redes móviles en la herramienta NS·*. Universidad Técnica Particular de Loja. Recuperado de http://dspace.utpl.edu.ec/bitstream/20.500.11962/22633/1/Torres Torres%2C Junior Israel.pdf

Vasques, F. (2015). *DHT-based Cluster Routing Protocol (DCRP): A Scalable Path Selection and Forwarding Protocol for IEEE 802.11s Mesh Networks*. Universidade do Porto. Recuperado de https://repositorio-aberto.up.pt/bitstream/10216/78935/2/35010.pdf

Vidal, I., García, C., Soto, I., & Moreno, I. (2003). *Servicios de Valor Añadido en Redes Móviles Ad-hoc*. Recuperado de http://www.it.uc3m.es/ividal/articulos/telecom03_adhoc.pdf

Vintimilla, E. P. (2016). *Simulación de protocolos de encaminamiento para la difusión de mensajes de alarma en redes vehiculares ad hoc mediante el uso de software*

libre. Universidad Politecnica Salesiana de Cuenca.

Viscaino, J. J. (2018). *EVALUACION DE RENDIMIENTO DE LAS TECNOLOGIAS 802.11 Y LTE PARA PROVEER SERVICIOS WEB EN REDES VEHICULARES AD-HOC*. Escuela Superior Politecnica de Chimborazo.

Yan, G., & Rawat, D. (2014). Data Mining Intrusion Detection in Vehicular Ad Hoc Network, (7), 1719–1726.

11. ANEXOS

ANEXO 1: CÓDIGO FUENTE

```
 // Definición de Librerías

#include <fstream>
#include <iostream>
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/mobility-module.h"
#include "ns3/wifi-module.h"
#include "ns3/aodv-module.h"
#include "ns3/applications-module.h"

 // Definición de nombres
 using namespace ns3;
 using namespace std;
 // Declaración de la clase Routing Experiment
class RoutingExperiment
{
 // Definición de variable de la clase Routing Experiment
public:
  RoutingExperiment ();
  void Run (int nSinks, double txp, string CSVfileName);
                                    int slotDistance);

// Declaración e Inicialización de las funciones privadas de la clase
Routing Experiment
private:
  Ptr<Socket> SetupPacketReceive (Ipv4Address addr, Ptr<Node> node);
  void ReceivePacket (Ptr<Socket> socket);
  void CheckThroughput ();

  uint32_t port;
  uint32_t bytesTotal;
  uint32_t packetsReceived;

  string m_CSVfileName;
  int m_nSinks;
  std::string m_protocolName;
  double m_txp;
  double m_time0=50;
  double m_time1=50;
  double m_time2=50;
  double m_time3=50;
```

```
  double m_time4=50;
  double time = 0;
  double m_jitter=0;
  double m_jitter0=0;
  double m_jitter1=0;
  double m_jitter2=0;
  double m_jitter3=0;
  double m_jitter4=0;
  double jitter = 0;
  double jitter0 = 0;
  double jitter1 = 0;
  double jitter2 = 0;
  double jitter3 = 0;
  double jitter4 = 0;
  double jitterSum = 0;
  double delaySum = 0;
  bool m_traceMobility;
  uint32_t m_protocol;
  double a;
};
  // Definición de la función Routing Experiment
RoutingExperiment::RoutingExperiment ()
  : port (9),
    bytesTotal (0),
    packetsReceived (1),
    m_CSVfileName ("comparacion_aodv.csv"),
    m_traceMobility (true)

{
}
  // Función para visualización de datos en la terminal, de datos de
recepción de paquetes.
static inline string
PrintReceivedPacket (Ptr<Socket> socket, Ptr<Packet> packet, Address
senderAddress)
{
  ostringstream oss;

  oss << Simulator::Now ().GetSeconds () << "   El nodo "  << socket-
>GetNode ()->GetId ();

  if (InetSocketAddress::IsMatchingType (senderAddress))
    {
      InetSocketAddress addr = InetSocketAddress::ConvertFrom
(senderAddress);
      oss << " Recibió un paquete de la direccion "<< addr.GetIpv4 () ;
```

```
    }
  else
    {
      oss << " received one packet!";
    }
  return oss.str ();
}
 // Declaración de la Función Receive Packet para el cálculo de métricas
delay, jitter y recolección de datos
void
RoutingExperiment::ReceivePacket (Ptr<Socket> socket)
{
  Ptr<Packet> packet;
  Address senderAddress;
  while ((packet = socket->RecvFrom (senderAddress)))
    {
    a=socket->GetNode ()->GetId ();
      bytesTotal += packet->GetSize ();
      packetsReceived += 1;
       if (a==0){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time0)*1000;
        m_time0 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter0);
        m_jitter0 =abs (abs(jitterSum) - m_jitter0) / 16;
        m_jitter=m_jitter0;
      jitter0 = delaySum;
        time=jitter0;
      } else if(a==1){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time1)*1000;
        m_time1 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter1);
        m_jitter1 =abs (abs(jitterSum) - m_jitter1) / 16;
        m_jitter=m_jitter1;
      jitter1 = delaySum;
        time=jitter1;
      } else if (a==2){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time2)*1000;
        m_time2 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter2);
        m_jitter2 =abs (abs(jitterSum) - m_jitter2) / 16;
        m_jitter=m_jitter2;
      jitter2 = delaySum;
        time=jitter2;
      }  else if (a==3){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time3)*1000;
        m_time3 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter3);
```

```
        m_jitter3 =abs (abs(jitterSum) - m_jitter3) / 16;
        m_jitter=m_jitter3;
      jitter3 = delaySum;
        time=jitter3;
      }  else if (a==4){
        delaySum = ((Simulator::Now ().GetSeconds ())-m_time4)*1000;
        m_time4 = (Simulator::Now ().GetSeconds ());
        jitterSum = (delaySum - jitter4);
        m_jitter4 =abs (abs(jitterSum) - m_jitter4) / 16;
        m_jitter=m_jitter4;
      jitter4 = delaySum;
        time=jitter4;
      }
      NS_LOG_UNCOND (PrintReceivedPacket (socket, packet,
senderAddress));
    }
}
 //Declaración de Función  CheckThroughput para el cálculo y la obtención
de métrica del Throughput
void
RoutingExperiment::CheckThroughput ()
{
  double kbs = (bytesTotal * 8.0) / (1000*(time/1000));
  bytesTotal = 0;

    // Impresión de datos obtenidos en archivo .CSV
  ofstream out (m_CSVfileName.c_str (), ios::app);

  out << (Simulator::Now ()).GetSeconds () << ","
      << kbs << ","
      << packetsReceived << ","
      << "10.1.1."<< a+11 << ","
      << "10.1.1."<< a+1 << ","
      << "AODV" << ","
      << delaySum << ","
      << m_jitter << ","
      << endl;
  out.close ();
  jitterSum = 0;
  delaySum = 0;
  jitter=0;
  m_jitter=0;
  packetsReceived = 0;
  Simulator::Schedule (Seconds (0.001),
&RoutingExperiment::CheckThroughput, this);
}
```

```
 // Declaración de función SetupPacketReceive para almacenar datos de
paquete enviado
Ptr<Socket>
RoutingExperiment::SetupPacketReceive (Ipv4Address addr, Ptr<Node> node)
{
  TypeId tid = TypeId::LookupByName ("ns3::UdpSocketFactory");
  Ptr<Socket> sink = Socket::CreateSocket (node, tid);
  InetSocketAddress local = InetSocketAddress (addr, port);
  sink->Bind (local);
  sink->SetRecvCallback (MakeCallback (&RoutingExperiment::ReceivePacket,
this));

  return sink;
}

 // Declaración función principal
int main (int argc, char *argv[])
{
  RoutingExperiment experiment;
  string CSVfileName = "comparacion_aodv";

 // Declaración para almacenamiento de variables en archivo .CSV
  ofstream out (CSVfileName.c_str ());
  out << "Tiempo," <<
  "Throughput," <<
  "Paquetes Recibidos," <<
  "Ip Origen ," <<
  "Ip Destino," <<
  "Protocolo," <<
  "Delay,"<<
  "Jitter"<<
  endl;
  out.close ();

  int nSinks = 1;
  double txp = 7.5;

  experiment.Run (nSinks, txp, CSVfileName);
}
 // Definición de ejecución de simulación
void
RoutingExperiment::Run (int nSinks, double txp, string CSVfileName)
{
  Packet::EnablePrinting ();
  m_nSinks = nSinks;
  m_txp = txp;
  m_CSVfileName = CSVfileName;
```

```
 // Definición de número de nodos en la red
  int nWifis = 100;

  double TotalTime = 100.0;
  string rate ("2048bps");
  string phyMode ("DsssRate11Mbps");
  string tr_name ("comparacion_aodv");
  int nodeSpeed = 20; //in m/s
  int nodePause = 0; //in s

//valores iniciales para empezar simulación
  Config::SetDefault  ("ns3::OnOffApplication::PacketSize",StringValue
("64"));
  Config::SetDefault ("ns3::OnOffApplication::DataRate",  StringValue
(rate));

  Config::SetDefault
("ns3::WifiRemoteStationManager::NonUnicastMode",StringValue (phyMode));

// Creación de nodos de ad-hoc
  NodeContainer adhocNodes;
  adhocNodes.Create (nWifis);

   // Establecer phywifi y canal utilizando helpers
  WifiHelper wifi;
  wifi.SetStandard (WIFI_PHY_STANDARD_80211b);

  // Definición de modelo de propagación en la transmisión
  YansWifiPhyHelper wifiPhy =  YansWifiPhyHelper::Default ();
  YansWifiChannelHelper wifiChannel;
  wifiChannel.SetPropagationDelay
("ns3::ConstantSpeedPropagationDelayModel");
  wifiChannel.AddPropagationLoss ("ns3::FriisPropagationLossModel");
  wifiPhy.SetChannel (wifiChannel.Create ());

  WifiMacHelper wifiMac;
  wifi.SetRemoteStationManager ("ns3::ConstantRateWifiManager",
                                "DataMode",StringValue (phyMode),
                                "ControlMode",StringValue (phyMode));
// Definición de la potencia de tx de la red
  wifiPhy.Set ("TxPowerStart",DoubleValue (txp));
  wifiPhy.Set ("TxPowerEnd", DoubleValue (txp));
```

```
 // Declaración del tipo de red a utilizar

  wifiMac.SetType ("ns3::AdhocWifiMac");
  NetDeviceContainer adhocDevices = wifi.Install (wifiPhy, wifiMac,
adhocNodes);
// Creación de escenario de movilidad
  MobilityHelper mobilityAdhoc;
  std::string traceFile = "/home/cristian/ns3/bake/source/ns-
3.28/scratch/mod_100.ns_movements";

//Helper ns2 leyendo el archivo de traza generado

  Ns2MobilityHelper ns2 = Ns2MobilityHelper (traceFile);
  ns2.Install ();

  // Declaración de características de capa 3, encaminamiento de
protocolos
  AodvHelper aodv;
  Ipv4ListRoutingHelper list;
  InternetStackHelper internet;
      list.Add (aodv, 100);
      internet.SetRoutingHelper (list);
      internet.Install (adhocNodes);

   // Asignación de direccionamiento en la red

  Ipv4AddressHelper addressAdhoc;
  addressAdhoc.SetBase ("10.1.1.0", "255.255.1.0");
  Ipv4InterfaceContainer adhocInterfaces;
  adhocInterfaces = addressAdhoc.Assign (adhocDevices);


// Definición de características de capa de transporte
  OnOffHelper onoff1 ("ns3::UdpSocketFactory",Address ());
  onoff1.SetAttribute ("OnTime", StringValue
("ns3::ConstantRandomVariable[Constant=1.0]"));
  onoff1.SetAttribute ("OffTime", StringValue
("ns3::ConstantRandomVariable[Constant=0.0]"));

// Ejecución de envíos en la red

  for (int i = 0; i < nSinks; i++)
    {
      Ptr<Socket> sink = SetupPacketReceive (adhocInterfaces.GetAddress
(i), adhocNodes.Get (i));
```

```
      AddressValue remoteAddress (InetSocketAddress
(adhocInterfaces.GetAddress (i), port));
      onoff1.SetAttribute ("Remote", remoteAddress);

      Ptr<UniformRandomVariable> var =
CreateObject<UniformRandomVariable> ();
      ApplicationContainer temp = onoff1.Install (adhocNodes.Get (i +
10));
      temp.Start (Seconds (var->GetValue (50.0,51.0)));
      temp.Stop (Seconds (TotalTime));
    }

  CheckThroughput ();
// Comando de inicio y simulación
  Simulator::Stop (Seconds (TotalTime));
  Simulator::Run ();
  Simulator::Destroy ();
}
```

ANEXO 2: DATOS SIMULACIÓN PROTOCOLOS

- **SIMULACIÓN PROTOCOLO AODV**

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay	Jitter
50,306	1,674	1	10.1.1.11	10.1.1.1	AODV	305,936	19,121
50,554	0,925	1	10.1.1.13	10.1.1.3	AODV	553,379	34,586
50,555	2,062	1	10.1.1.11	10.1.1.1	AODV	248,328	2,405
50,557	179,035	1	10.1.1.13	10.1.1.3	AODV	2,860	32,246
50,732	0,700	1	10.1.1.15	10.1.1.5	AODV	731,068	45,692
50,765	2,463	1	10.1.1.13	10.1.1.3	AODV	207,877	10,798
50,803	2,066	1	10.1.1.11	10.1.1.1	AODV	247,823	0,119
50,976	2,094	1	10.1.1.15	10.1.1.5	AODV	244,459	27,557
51,015	2,042	1	10.1.1.13	10.1.1.3	AODV	250,755	2,005
51,053	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,129
51,226	2,044	1	10.1.1.15	10.1.1.5	AODV	250,440	1,349
51,304	2,035	1	10.1.1.11	10.1.1.1	AODV	251,608	0,092
51,316	1,701	1	10.1.1.13	10.1.1.3	AODV	301,072	3,020
51,476	2,052	1	10.1.1.15	10.1.1.5	AODV	249,560	0,029
51,553	2,061	1	10.1.1.11	10.1.1.1	AODV	248,392	0,195
51,727	2,038	1	10.1.1.15	10.1.1.5	AODV	251,172	0,099
51,771	1,126	1	10.1.1.13	10.1.1.3	AODV	454,580	9,406
51,803	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,088
52,031	1,687	1	10.1.1.15	10.1.1.5	AODV	303,470	3,262
52,047	1,855	1	10.1.1.13	10.1.1.3	AODV	275,962	10,576
52,053	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,006
52,226	2,621	1	10.1.1.15	10.1.1.5	AODV	195,358	6,553
52,267	2,330	1	10.1.1.13	10.1.1.3	AODV	219,715	2,854
52,303	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,000
52,476	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	3,006
52,512	2,087	1	10.1.1.13	10.1.1.3	AODV	245,361	1,424
52,553	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,000
52,577	2,031	1	10.1.1.14	10.1.1.4	AODV	252,039	120,44
52,726	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,188
52,766	2,013	1	10.1.1.13	10.1.1.3	AODV	254,372	0,474
52,803	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,000

52,833	2,003	1	10.1.1.14	10.1.1.4	AODV	255,636	7,303
52,976	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,012
53,033	1,923	1	10.1.1.13	10.1.1.3	AODV	266,310	0,717
53,053	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,000
53,069	2,170	1	10.1.1.14	10.1.1.4	AODV	235,996	0,771
53,226	2,044	1	10.1.1.15	10.1.1.5	AODV	250,460	0,028
53,268	2,172	1	10.1.1.13	10.1.1.3	AODV	235,728	1,867
53,303	2,048	1	10.1.1.11	10.1.1.1	AODV	250,000	0,000
53,318	2,049	1	10.1.1.14	10.1.1.4	AODV	249,92	0,822
53,476	2,052	1	10.1.1.15	10.1.1.5	AODV	249,54	0,056
53,517	2,058	1	10.1.1.13	10.1.1.3	AODV	248,77	0,698
53,558	2,007	1	10.1.1.11	10.1.1.1	AODV	255,07	0,317
53,569	2,041	1	10.1.1.14	10.1.1.4	AODV	250,88	0,008
53,726	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,025
53,803	2,090	1	10.1.1.11	10.1.1.1	AODV	244,93	0,613
53,818	2,056	1	10.1.1.14	10.1.1.4	AODV	249,04	0,114
53,976	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,002
54,053	2,048	1	10.1.1.11	10.1.1.1	AODV	250,00	0,278
54,12	1,696	1	10.1.1.14	10.1.1.4	AODV	301,97	3,301
54,226	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,000
54,303	2,048	1	10.1.1.11	10.1.1.1	AODV	250,00	0,017
54,318	2,590	1	10.1.1.14	10.1.1.4	AODV	197,67	6,312
54,476	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,000
54,553	2,048	1	10.1.1.11	10.1.1.1	AODV	250,00	0,001
54,569	2,043	1	10.1.1.14	10.1.1.4	AODV	250,60	2,913
54,726	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,000
54,803	2,048	1	10.1.1.11	10.1.1.1	AODV	250,00	0,000
54,818	2,053	1	10.1.1.14	10.1.1.4	AODV	249,40	0,107
54,976	2,048	1	10.1.1.15	10.1.1.5	AODV	250,00	0,000

Tabla 13 Datos primeros 5 segundos al inicio de simulación

Fuente: Autor

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay	Jitter
70,069	2,051	1	10.1.1.14	10.1.1.4	AODV	249,620	0,063
70,229	2,033	1	10.1.1.15	10.1.1.5	AODV	251,832	3,628
70,319	2,046	1	10.1.1.14	10.1.1.4	AODV	250,240	0,035
70,482	2,023	1	10.1.1.15	10.1.1.5	AODV	253,126	0,146
70,569	2,048	1	10.1.1.14	10.1.1.4	AODV	250,060	0,009
70,786	1,685	1	10.1.1.15	10.1.1.5	AODV	303,883	3,163
70,819	2,050	1	10.1.1.14	10.1.1.4	AODV	249,780	0,017
70,992	2,484	1	10.1.1.15	10.1.1.5	AODV	206,140	5,911
71,002	0,130	1	10.1.1.11	10.1.1.1	AODV	3945,710	233,710
71,004	242,676	1	10.1.1.11	10.1.1.1	AODV	2,110	231,868
71,005	469,819	1	10.1.1.11	10.1.1.1	AODV	1,090	14,428
71,074	2,013	1	10.1.1.14	10.1.1.4	AODV	254,366	0,286
71,081	43,437	1	10.1.1.12	10.1.1.2	AODV	11,787	412,095
71,097	5,576	1	10.1.1.11	10.1.1.1	AODV	91,816	4,769
71,118	13,720	1	10.1.1.12	10.1.1.2	AODV	37,318	24,160
71,234	2,114	1	10.1.1.15	10.1.1.5	AODV	242,176	1,883
71,242	4,124	1	10.1.1.12	10.1.1.2	AODV	124,153	3,917
71,304	2,479	1	10.1.1.11	10.1.1.1	AODV	206,562	6,874
71,485	1,245	1	10.1.1.14	10.1.1.4	AODV	411,331	9,792
71,489	2,071	1	10.1.1.12	10.1.1.2	AODV	247,207	7,446
71,553	2,050	1	10.1.1.11	10.1.1.1	AODV	249,740	2,269
71,568	6,128	1	10.1.1.14	10.1.1.4	AODV	83,554	19,874
71,761	1,883	1	10.1.1.12	10.1.1.2	AODV	271,954	1,081
71,781	0,936	1	10.1.1.15	10.1.1.5	AODV	547,163	18,944
71,804	2,046	1	10.1.1.11	10.1.1.1	AODV	250,240	0,111
71,818	2,049	1	10.1.1.14	10.1.1.4	AODV	249,920	9,156
71,978	2,609	1	10.1.1.15	10.1.1.5	AODV	196,209	20,751
71,986	2,276	1	10.1.1.12	10.1.1.2	AODV	224,964	2,869
72,053	2,052	1	10.1.1.11	10.1.1.1	AODV	249,520	0,038
72,068	2,049	1	10.1.1.14	10.1.1.4	AODV	249,860	0,568
72,228	2,048	1	10.1.1.15	10.1.1.5	AODV	249,960	2,063
72,234	2,062	1	10.1.1.12	10.1.1.2	AODV	248,244	1,276
72,304	2,046	1	10.1.1.11	10.1.1.1	AODV	250,300	0,046
72,318	2,046	1	10.1.1.14	10.1.1.4	AODV	250,300	0,008
72,478	2,042	1	10.1.1.15	10.1.1.5	AODV	250,720	0,081

72,489	2,011	1	10.1.1.12	10.1.1.2	AODV	254,632	0,320
72,553	2,051	1	10.1.1.11	10.1.1.1	AODV	249,640	0,038
72,569	2,047	1	10.1.1.14	10.1.1.4	AODV	250,140	0,009
72,728	2,049	1	10.1.1.15	10.1.1.5	AODV	249,820	0,051
72,734	2,087	1	10.1.1.12	10.1.1.2	AODV	245,368	0,559
72,818	2,050	1	10.1.1.14	10.1.1.4	AODV	249,744	0,024
72,855	1,697	1	10.1.1.11	10.1.1.1	AODV	301,628	3,247
72,977	2,055	1	10.1.1.15	10.1.1.5	AODV	249,200	0,036
72,984	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,255
73,053	2,579	1	10.1.1.11	10.1.1.1	AODV	198,552	6,239
73,069	2,046	1	10.1.1.14	10.1.1.4	AODV	250,256	0,030
73,234	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,016
73,237	1,970	1	10.1.1.15	10.1.1.5	AODV	259,912	0,667
73,303	2,049	1	10.1.1.11	10.1.1.1	AODV	249,860	2,817
73,318	2,049	1	10.1.1.14	10.1.1.4	AODV	249,820	0,025
73,486	2,033	1	10.1.1.12	10.1.1.2	AODV	251,896	0,118
73,517	1,832	1	10.1.1.15	10.1.1.5	AODV	279,448	1,179
73,553	2,047	1	10.1.1.11	10.1.1.1	AODV	250,080	0,162
73,568	2,051	1	10.1.1.14	10.1.1.4	AODV	249,580	0,013
73,735	2,058	1	10.1.1.12	10.1.1.2	AODV	248,820	0,185
73,803	2,048	1	10.1.1.11	10.1.1.1	AODV	250,020	0,006
73,843	1,860	1	10.1.1.14	10.1.1.4	AODV	275,263	1,604
73,984	2,054	1	10.1.1.12	10.1.1.2	AODV	249,284	0,017
74,054	2,045	1	10.1.1.11	10.1.1.1	AODV	250,340	0,020
74,068	2,274	1	10.1.1.14	10.1.1.4	AODV	225,177	3,030
74,234	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,044
74,303	2,050	1	10.1.1.11	10.1.1.1	AODV	249,760	0,035
74,319	2,043	1	10.1.1.14	10.1.1.4	AODV	250,641	1,402
74,488	2,021	1	10.1.1.12	10.1.1.2	AODV	253,312	0,204
74,554	2,046	1	10.1.1.11	10.1.1.1	AODV	250,260	0,029
74,57	2,041	1	10.1.1.14	10.1.1.4	AODV	250,895	0,072
74,736	2,061	1	10.1.1.12	10.1.1.2	AODV	248,384	0,295
74,803	2,051	1	10.1.1.11	10.1.1.1	AODV	249,680	0,034
74,869	1,710	1	10.1.1.14	10.1.1.4	AODV	299,345	3,024
74,984	2,062	1	10.1.1.12	10.1.1.2	AODV	248,304	0,013

Tabla 14 Datos de 5 segundos a la mitad de la simulación

Fuente: Autor

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay	Jitter
95,226	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,000
95,234	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,000
95,303	0,684	1	10.1.1.11	10.1.1.1	AODV	748,534	31,320
95,476	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,000
95,484	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,000
95,553	2,048	1	10.1.1.11	10.1.1.1	AODV	249,940	29,205
95,726	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,000
95,734	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,000
95,803	2,045	1	10.1.1.11	10.1.1.1	AODV	250,360	1,799
95,976	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,000
95,984	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,000
96,054	2,046	1	10.1.1.11	10.1.1.1	AODV	250,200	0,102
96,226	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,000
96,234	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,000
96,304	2,049	1	10.1.1.11	10.1.1.1	AODV	249,840	0,016
96,477	2,041	1	10.1.1.15	10.1.1.5	AODV	250,816	0,051
96,486	2,038	1	10.1.1.12	10.1.1.2	AODV	251,216	0,076
96,553	2,051	1	10.1.1.11	10.1.1.1	AODV	249,660	0,010
96,726	2,055	1	10.1.1.15	10.1.1.5	AODV	249,184	0,099
96,747	1,963	1	10.1.1.12	10.1.1.2	AODV	260,828	0,596
96,803	2,048	1	10.1.1.11	10.1.1.1	AODV	249,940	0,017
96,976	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	0,045
96,984	2,152	1	10.1.1.12	10.1.1.2	AODV	237,956	1,392
97,054	2,041	1	10.1.1.11	10.1.1.1	AODV	250,884	0,058
97,241	1,931	1	10.1.1.15	10.1.1.5	AODV	265,204	0,947
97,243	1,983	1	10.1.1.12	10.1.1.2	AODV	258,173	1,177
97,304	2,050	1	10.1.1.11	10.1.1.1	AODV	249,736	0,068
97,523	1,828	1	10.1.1.12	10.1.1.2	AODV	280,106	1,297
97,553	2,051	1	10.1.1.11	10.1.1.1	AODV	249,580	0,005
97,736	1,034	1	10.1.1.15	10.1.1.5	AODV	494,937	14,299
97,737	2,390	1	10.1.1.12	10.1.1.2	AODV	214,191	4,039
97,807	2,022	1	10.1.1.11	10.1.1.1	AODV	253,172	0,224
97,976	2,135	1	10.1.1.15	10.1.1.5	AODV	239,859	15,049
97,995	1,986	1	10.1.1.12	10.1.1.2	AODV	257,794	2,473
98,055	2,058	1	10.1.1.11	10.1.1.1	AODV	248,824	0,258

98,236	2,123	1	10.1.1.12	10.1.1.2	AODV	241,156	0,885
98,485	1,006	1	10.1.1.15	10.1.1.5	AODV	509,084	15,886
98,514	1,840	1	10.1.1.12	10.1.1.2	AODV	278,269	2,264
98,576	0,984	1	10.1.1.11	10.1.1.1	AODV	520,293	16,951
98,736	2,310	1	10.1.1.12	10.1.1.2	AODV	221,631	3,398
98,816	1,546	1	10.1.1.15	10.1.1.5	AODV	331,087	10,132
98,843	1,918	1	10.1.1.11	10.1.1.1	AODV	267,012	14,771
98,976	3,203	1	10.1.1.15	10.1.1.5	AODV	159,829	10,070
98,985	2,051	1	10.1.1.12	10.1.1.2	AODV	249,688	1,541
99,054	2,426	1	10.1.1.11	10.1.1.1	AODV	211,019	2,576
99,226	2,048	1	10.1.1.15	10.1.1.5	AODV	250,000	5,006
99,235	2,049	1	10.1.1.12	10.1.1.2	AODV	249,920	0,082
99,316	1,951	1	10.1.1.11	10.1.1.1	AODV	262,429	3,052
99,48	2,014	1	10.1.1.15	10.1.1.5	AODV	254,266	0,046
99,487	2,038	1	10.1.1.12	10.1.1.2	AODV	251,256	0,078
99,587	1,890	1	10.1.1.11	10.1.1.1	AODV	270,962	0,343
99,726	2,084	1	10.1.1.15	10.1.1.5	AODV	245,734	0,530
99,736	2,057	1	10.1.1.12	10.1.1.2	AODV	248,944	0,140
99,977	2,042	1	10.1.1.15	10.1.1.5	AODV	250,696	0,277
99,986	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,057
73,803	2,048	1	10.1.1.11	10.1.1.1	AODV	250,020	0,006
73,843	1,860	1	10.1.1.14	10.1.1.4	AODV	275,263	1,604
73,984	2,054	1	10.1.1.12	10.1.1.2	AODV	249,284	0,017
74,054	2,045	1	10.1.1.11	10.1.1.1	AODV	250,340	0,020
74,068	2,274	1	10.1.1.14	10.1.1.4	AODV	225,177	3,030
74,234	2,048	1	10.1.1.12	10.1.1.2	AODV	250,000	0,044
74,303	2,050	1	10.1.1.11	10.1.1.1	AODV	249,760	0,035
74,319	2,043	1	10.1.1.14	10.1.1.4	AODV	250,641	1,402
74,488	2,021	1	10.1.1.12	10.1.1.2	AODV	253,312	0,204
74,554	2,046	1	10.1.1.11	10.1.1.1	AODV	250,260	0,029
74,57	2,041	1	10.1.1.14	10.1.1.4	AODV	250,895	0,072
74,736	2,061	1	10.1.1.12	10.1.1.2	AODV	248,384	0,295
74,803	2,051	1	10.1.1.11	10.1.1.1	AODV	249,680	0,034
74,869	1,710	1	10.1.1.14	10.1.1.4	AODV	299,345	3,024
74,984	2,062	1	10.1.1.12	10.1.1.2	AODV	248,304	0,013

Tabla 15 Datos de 5 segundos al final de simulación
Fuente: Autor

- **SIMULACIÓN PROTOCOLO DSR**

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay	Jitter
61,504	29,894	1	10.1.1.12	10.1.1.2	DSR	17,13	39,533
61,58	6,740	1	10.1.1.12	10.1.1.2	DSR	75,96	1,206
61,581	2,043	1	10.1.1.11	10.1.1.1	DSR	250,56	39,246
61,589	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	38,885
61,627	11,036	1	10.1.1.12	10.1.1.2	DSR	46,39	1,773
61,645	27,575	1	10.1.1.12	10.1.1.2	DSR	18,57	1,628
61,831	2,053	1	10.1.1.11	10.1.1.1	DSR	249,44	2,383
61,839	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	2,430
61,883	2,154	1	10.1.1.12	10.1.1.2	DSR	237,65	13,591
62,081	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,114
62,089	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,152
62,127	2,096	1	10.1.1.12	10.1.1.2	DSR	244,22	0,439
62,331	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,007
62,34	2,041	1	10.1.1.13	10.1.1.3	DSR	250,81	0,041
62,388	1,964	1	10.1.1.12	10.1.1.2	DSR	260,64	0,999
62,581	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
62,591	2,043	1	10.1.1.13	10.1.1.3	DSR	250,62	0,010
62,627	2,137	1	10.1.1.12	10.1.1.2	DSR	239,54	1,256
62,831	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
62,839	2,060	1	10.1.1.13	10.1.1.3	DSR	248,57	0,127
62,878	2,041	1	10.1.1.12	10.1.1.2	DSR	250,83	0,627
62,932	2,244	1	10.1.1.14	10.1.1.4	DSR	228,17	38,235
63,081	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
63,089	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,081
63,128	2,050	1	10.1.1.12	10.1.1.2	DSR	249,78	0,027
63,182	2,049	1	10.1.1.14	10.1.1.4	DSR	249,90	1,032
63,331	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
63,339	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,005
63,432	2,046	1	10.1.1.14	10.1.1.4	DSR	250,28	0,041
63,582	2,039	1	10.1.1.11	10.1.1.1	DSR	251,06	0,067
63,601	1,955	1	10.1.1.13	10.1.1.3	DSR	261,88	0,742
63,682	2,051	1	10.1.1.14	10.1.1.4	DSR	249,66	0,036
63,827	0,732	1	10.1.1.12	10.1.1.2	DSR	699,01	28,075
63,831	2,057	1	10.1.1.11	10.1.1.1	DSR	248,94	0,129

63,839	2,150	1	10.1.1.13	10.1.1.3	DSR	238,13	1,438
63,88	9,601	1	10.1.1.12	10.1.1.2	DSR	53,33	38,600
63,932	2,047	1	10.1.1.14	10.1.1.4	DSR	250,18	0,030
64,081	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,058
64,089	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,652
64,127	2,074	1	10.1.1.12	10.1.1.2	DSR	246,83	9,682
64,182	2,048	1	10.1.1.14	10.1.1.4	DSR	249,98	0,011
64,331	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,004
64,339	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,041
64,378	2,040	1	10.1.1.12	10.1.1.2	DSR	250,99	0,345
64,432	2,047	1	10.1.1.14	10.1.1.4	DSR	250,14	0,009
64,581	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
64,589	2,048	1	10.1.1.13	10.1.1.3	DSR	250,00	0,003
64,632	2,016	1	10.1.1.12	10.1.1.2	DSR	253,94	0,163
64,637	103,930	1	10.1.1.12	10.1.1.2	DSR	4,93	15,553
64,682	2,052	1	10.1.1.14	10.1.1.4	DSR	249,54	0,037
64,831	2,048	1	10.1.1.11	10.1.1.1	DSR	250,00	0,000
64,879	2,141	1	10.1.1.12	10.1.1.2	DSR	239,18	14,739
64,931	2,050	1	10.1.1.14	10.1.1.4	DSR	249,80	0,014

Tabla 16 Datos de 5 segundos al inicio de simulación

Fuente: Autor

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay	Jitter
95,224	2,13877	1	10.1.1.14	10.1.1.4	DSR	239,39	391,108
95,241	2,06012	1	10.1.1.12	10.1.1.2	DSR	248,53	3,98037
95,297	1,23838	1	10.1.1.11	10.1.1.1	DSR	413,442	3,95551
95,479	2,01497	1	10.1.1.14	10.1.1.4	DSR	254,098	23,525
95,49	2,05672	1	10.1.1.12	10.1.1.2	DSR	248,94	0,223129
95,579	1,81465	1	10.1.1.11	10.1.1.1	DSR	282,148	7,95865
95,723	2,09157	1	10.1.1.14	10.1.1.4	DSR	244,792	0,88867
95,784	1,7414	1	10.1.1.12	10.1.1.2	DSR	294,017	2,80335
95,978	2,01428	1	10.1.1.14	10.1.1.4	DSR	254,185	0,531535
95,992	1,2394	1	10.1.1.11	10.1.1.1	DSR	413,103	7,68728
96,002	2,34859	1	10.1.1.12	10.1.1.2	DSR	218,003	4,57564
96,11	4,33324	1	10.1.1.11	10.1.1.1	DSR	118,156	17,9537
96,112	329,967	1	10.1.1.11	10.1.1.1	DSR	1,55167	6,16569
96,224	2,07881	1	10.1.1.14	10.1.1.4	DSR	246,295	0,459905
96,263	3,37515	1	10.1.1.11	10.1.1.1	DSR	151,697	8,99873
96,278	0,249124	1	10.1.1.15	10.1.1.5	DSR	2055,2	106,809
96,281	1,83231	1	10.1.1.12	10.1.1.2	DSR	279,429	3,55312
96,473	2,05309	1	10.1.1.14	10.1.1.4	DSR	249,38	0,164068
96,503	2,13968	1	10.1.1.11	10.1.1.1	DSR	239,288	4,91199
96,505	2,28912	1	10.1.1.12	10.1.1.2	DSR	223,667	3,26304
96,575	1,72294	1	10.1.1.15	10.1.1.5	DSR	297,166	103,201
96,723	2,04964	1	10.1.1.14	10.1.1.4	DSR	249,8	0,0159958
96,753	2,06449	1	10.1.1.12	10.1.1.2	DSR	248,004	1,31711
96,754	2,03482	1	10.1.1.11	10.1.1.1	DSR	251,619	0,46373
96,825	2,05009	1	10.1.1.15	10.1.1.5	DSR	249,745	3,48632
96,975	2,03505	1	10.1.1.14	10.1.1.4	DSR	251,591	0,110924
97,003	2,05885	1	10.1.1.11	10.1.1.1	DSR	248,683	0,154562
97,077	2,03322	1	10.1.1.15	10.1.1.5	DSR	251,817	0,0883917
97,223	2,05731	1	10.1.1.14	10.1.1.4	DSR	248,869	0,163165
97,256	2,02198	1	10.1.1.11	10.1.1.1	DSR	253,217	0,273764
97,265	0,999267	1	10.1.1.12	10.1.1.2	DSR	512,376	16,4409
97,321	2,0915	1	10.1.1.15	10.1.1.5	DSR	244,801	0,433011
97,476	2,03133	1	10.1.1.14	10.1.1.4	DSR	252,052	0,188725
97,499	2,18632	1	10.1.1.12	10.1.1.2	DSR	234,184	16,3594
97,505	2,05826	1	10.1.1.11	10.1.1.1	DSR	248,754	0,261877

97,724	2,05968	1	10.1.1.14	10.1.1.4	DSR	248,582	0,205052
97,751	2,08105	1	10.1.1.11	10.1.1.1	DSR	246,03	0,153877
97,79	1,76145	1	10.1.1.12	10.1.1.2	DSR	290,669	2,50785
97,973	2,05469	1	10.1.1.14	10.1.1.4	DSR	249,186	0,0248796
98,004	2,02095	1	10.1.1.11	10.1.1.1	DSR	253,346	0,447653
98,039	2,05954	1	10.1.1.12	10.1.1.2	DSR	248,599	2,47265
98,223	2,04931	1	10.1.1.14	10.1.1.4	DSR	249,84	0,0393475
98,476	2,02695	1	10.1.1.14	10.1.1.4	DSR	252,596	0,169791
98,501	2,04784	1	10.1.1.11	10.1.1.1	DSR	250,02	0,193015
98,506	1,09545	1	10.1.1.12	10.1.1.2	DSR	467,386	13,5197
98,728	2,02977	1	10.1.1.14	10.1.1.4	DSR	252,245	0,0113314
98,747	2,12713	1	10.1.1.12	10.1.1.2	DSR	240,7	13,3229
98,752	2,03487	1	10.1.1.11	10.1.1.1	DSR	251,613	0,0874993
98,863	0,332142	1	10.1.1.15	10.1.1.5	DSR	1541,51	81,0173
98,974	2,08487	1	10.1.1.14	10.1.1.4	DSR	245,579	0,415905
99,001	2,05948	1	10.1.1.11	10.1.1.1	DSR	248,607	0,182406
99,004	1,98769	1	10.1.1.12	10.1.1.2	DSR	257,586	0,222697
99,224	2,04767	1	10.1.1.14	10.1.1.4	DSR	250,04	0,252813
99,242	2,14945	1	10.1.1.12	10.1.1.2	DSR	238,2	1,19769
99,251	2,04734	1	10.1.1.11	10.1.1.1	DSR	250,08	0,0806603
99,474	2,04505	1	10.1.1.14	10.1.1.4	DSR	250,36	0,00419915
99,489	2,07984	1	10.1.1.12	10.1.1.2	DSR	246,173	0,42341
99,501	2,04507	1	10.1.1.11	10.1.1.1	DSR	250,358	0,0123199
99,639	1,25626	1	10.1.1.15	10.1.1.5	DSR	407,558	1,79634
99,724	2,05128	1	10.1.1.14	10.1.1.4	DSR	249,6	0,0472375
99,752	2,04078	1	10.1.1.11	10.1.1.1	DSR	250,884	0,0321116
99,78	1,75719	1	10.1.1.12	10.1.1.2	DSR	291,374	2,7986
99,857	2,3449	1	10.1.1.15	10.1.1.5	DSR	218,346	11,7135
99,975	2,0389	1	10.1.1.14	10.1.1.4	DSR	251,116	0,0917977
99,989	2,45042	1	10.1.1.12	10.1.1.2	DSR	208,944	4,97696

Tabla 17 Datos de 5 segundos a la mitad de la simulación

Fuente: Autor

Tiempo	Throughput	Paquetes Recibidos	Ip Destino	Ip Origen	Protocolo	Delay
115,001	2,04734	1	10.1.1.11	10.1.1.1	DSR	250,08
115,252	2,04253	1	10.1.1.11	10.1.1.1	DSR	250,67
115,503	2,04064	1	10.1.1.11	10.1.1.1	DSR	250,902
115,751	2,06328	1	10.1.1.11	10.1.1.1	DSR	248,148
116,001	2,04751	1	10.1.1.11	10.1.1.1	DSR	250,06
116,251	2,04669	1	10.1.1.11	10.1.1.1	DSR	250,16
116,502	2,04011	1	10.1.1.11	10.1.1.1	DSR	250,967
116,751	2,0543	1	10.1.1.11	10.1.1.1	DSR	249,233
117,001	2,04816	1	10.1.1.11	10.1.1.1	DSR	249,98
117,251	2,05145	1	10.1.1.11	10.1.1.1	DSR	249,58
117,501	2,04751	1	10.1.1.11	10.1.1.1	DSR	250,06
117,751	2,04849	1	10.1.1.11	10.1.1.1	DSR	249,94
118,001	2,04604	1	10.1.1.11	10.1.1.1	DSR	250,24
118,254	2,02303	1	10.1.1.11	10.1.1.1	DSR	253,086
118,504	2,04775	1	10.1.1.11	10.1.1.1	DSR	250,031
118,751	2,07369	1	10.1.1.11	10.1.1.1	DSR	246,903
119,002	2,03752	1	10.1.1.11	10.1.1.1	DSR	251,286
119,073	0,0367123	1	10.1.1.14	10.1.1.4	DSR	13946,3
119,264	1,95613	1	10.1.1.11	10.1.1.1	DSR	261,741
119,331	1,98567	1	10.1.1.14	10.1.1.4	DSR	257,848
119,474	3,56956	1	10.1.1.14	10.1.1.4	DSR	143,435
119,501	2,16442	1	10.1.1.11	10.1.1.1	DSR	236,553
119,828	1,44654	1	10.1.1.14	10.1.1.4	DSR	353,948
119,89	1,31432	1	10.1.1.11	10.1.1.1	DSR	389,556
119,984	3,28542	1	10.1.1.14	10.1.1.4	DSR	155,84
96,975	2,03505	1	10.1.1.14	10.1.1.4	DSR	251,591
97,003	2,05885	1	10.1.1.11	10.1.1.1	DSR	248,683
97,077	2,03322	1	10.1.1.15	10.1.1.5	DSR	251,817
97,223	2,05731	1	10.1.1.14	10.1.1.4	DSR	248,869
97,256	2,02198	1	10.1.1.11	10.1.1.1	DSR	253,217
97,265	0,999267	1	10.1.1.12	10.1.1.2	DSR	512,376
97,321	2,0915	1	10.1.1.15	10.1.1.5	DSR	244,801
97,476	2,03133	1	10.1.1.14	10.1.1.4	DSR	252,052
97,499	2,18632	1	10.1.1.12	10.1.1.2	DSR	234,184
97,505	2,05826	1	10.1.1.11	10.1.1.1	DSR	248,754

97,724	2,05968	1	10.1.1.14	10.1.1.4	DSR	248,582
97,751	2,08105	1	10.1.1.11	10.1.1.1	DSR	246,03
97,79	1,76145	1	10.1.1.12	10.1.1.2	DSR	290,669
97,973	2,05469	1	10.1.1.14	10.1.1.4	DSR	249,186
98,004	2,02095	1	10.1.1.11	10.1.1.1	DSR	253,346
98,039	2,05954	1	10.1.1.12	10.1.1.2	DSR	248,599
98,223	2,04931	1	10.1.1.14	10.1.1.4	DSR	249,84
98,251	2,07679	1	10.1.1.11	10.1.1.1	DSR	246,534
98,476	2,02695	1	10.1.1.14	10.1.1.4	DSR	252,596
98,501	2,04784	1	10.1.1.11	10.1.1.1	DSR	250,02
98,506	1,09545	1	10.1.1.12	10.1.1.2	DSR	467,386
98,728	2,02977	1	10.1.1.14	10.1.1.4	DSR	252,245
98,747	2,12713	1	10.1.1.12	10.1.1.2	DSR	240,7
98,752	2,03487	1	10.1.1.11	10.1.1.1	DSR	251,613
98,863	0,332142	1	10.1.1.15	10.1.1.5	DSR	1541,51
98,974	2,08487	1	10.1.1.14	10.1.1.4	DSR	245,579
99,001	2,05948	1	10.1.1.11	10.1.1.1	DSR	248,607
99,004	1,98769	1	10.1.1.12	10.1.1.2	DSR	257,586
99,242	2,14945	1	10.1.1.12	10.1.1.2	DSR	238,2
99,251	2,04734	1	10.1.1.11	10.1.1.1	DSR	250,08
99,474	2,04505	1	10.1.1.14	10.1.1.4	DSR	250,36
99,489	2,07984	1	10.1.1.12	10.1.1.2	DSR	246,173
99,501	2,04507	1	10.1.1.11	10.1.1.1	DSR	250,358
99,639	1,25626	1	10.1.1.15	10.1.1.5	DSR	407,558
99,724	2,05128	1	10.1.1.14	10.1.1.4	DSR	249,6
99,752	2,04078	1	10.1.1.11	10.1.1.1	DSR	250,884
99,78	1,75719	1	10.1.1.12	10.1.1.2	DSR	291,374
99,857	2,3449	1	10.1.1.15	10.1.1.5	DSR	218,346
99,975	2,0389	1	10.1.1.14	10.1.1.4	DSR	251,116
99,989	2,45042	1	10.1.1.12	10.1.1.2	DSR	208,944

Tabla 18 Datos de 5 segundos al final de simulación

Fuente: Autor

ANEXO 3: ANÁLISIS DE PLAGIO EN URKUND

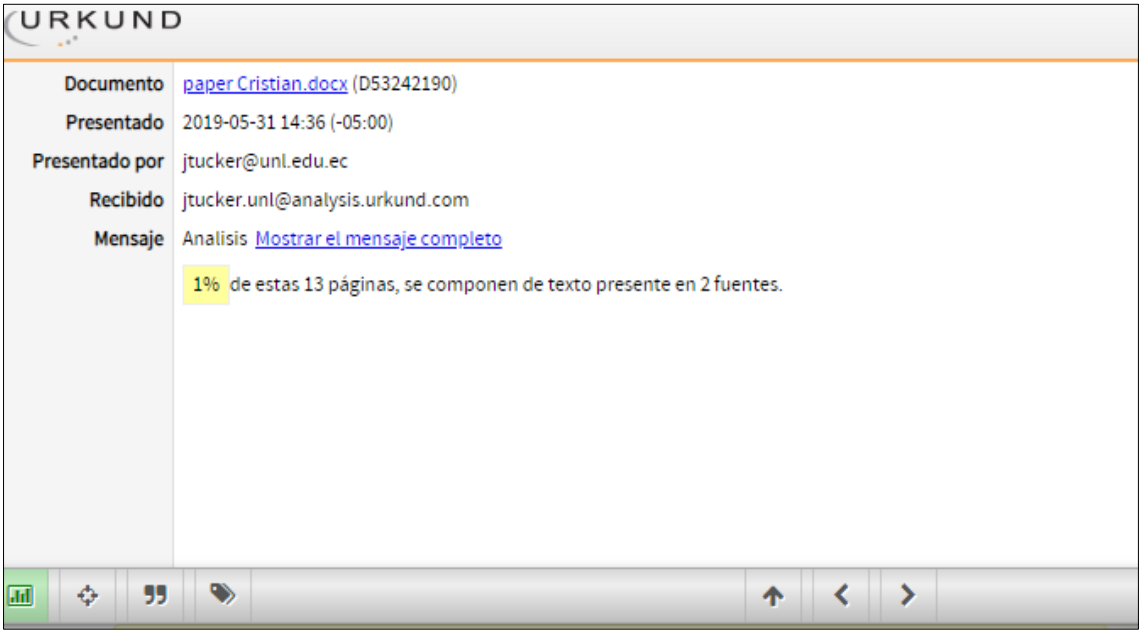


Figura 54 Análisis plagio Urkund

Fuente: Urkund