

NETWORK ROUTING IN INTERNET OF VEHICLES ENVIRONMENT

Project report submitted in partial fulfillment of the requirement for
the degree of Bachelor of Technology

in

Computer Science and Engineering

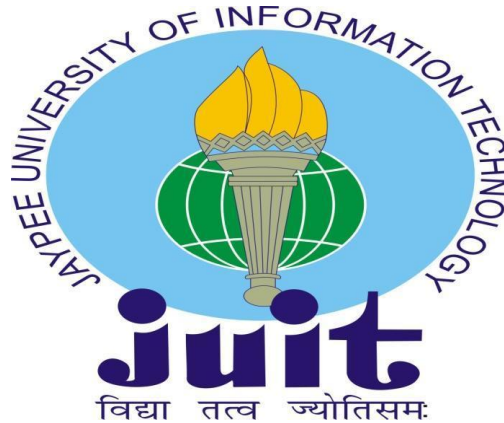
By

Ashutosh Kumar (191376)

Under the supervision of

Mr. Nishant Sharma

to



Department of Computer Science & Engineering and Information
Technology

Jaypee University of Information Technology Waknaghat,

Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled “**Network Routing in Internet of Vehicles Environment.**” in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering/** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Wagnaghat is an authentic record of my own work carried out over a period from January 2023 to May 2023 under the supervision of **Mr. Nishant Sharma** (Assistant Professor Grade-II - CSE Dept.).

I also authenticate that I have carried out the above-mentioned project work under the proficiency stream **Cloud Computing.**

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Ashutosh Kumar 191376

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Supervisor Name: Mr. Nishant Sharma

Designation: Assistant Professor Grade II

Department name: Computer Science

Dated:

Plagiarism Certificate

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT
PLAGIARISM VERIFICATION REPORT

Date:

Type of Document (Tick): PhD Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: _____ Department: _____ Enrolment No _____

Contact No. _____ E-mail. _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none"> • All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String 		Word Counts	
Report Generated on			Character Counts	
			Submission ID	Total Pages Scanned
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

Acknowledgement

Firstly, I express my heartiest thanks and gratefulness to almighty God for his divine blessing that made it possible for us to complete the project work successfully.

I am really grateful and wish my profound indebtedness to supervisor **Mr. Nishant Sharma, Assistant Professor**, Department of CSE, Jaypee University of Information Technology, Waknaghat. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to **Mr. Nishant Sharma**, Department of CSE, for his kind help to finish my project.

I would also generously welcome each one of those individuals who have helped me straight forward or in a roundabout way in making this project a win. In this unique situation, I also want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patience of my parents.

Ashutosh Kumar 191376

Table of contents

Candidate's Declaration.....	i
Plagiarism Certificate.....	ii
Acknowledgement.....	iii
List of Abbreviations.....	v
List of Figures.....	vi-vii
List of Tables.....	viii
Abstract.....	ix-x
1. INTRODUCTION.....	1-17
1.1 Problem Statement.....	4-5
1.2 Objectives.....	5-6
1.3 Internet of Vehicles Architecture.....	6-17
2. LITERATURE SURVEY.....	18-23
3. NETWORK ROUTING PROTOCOLS/ALGORITHM.....	24-43
4. EXPERIMENT & RESULT ANALYSIS.....	44-53
5. CONCLUSION.....	54-57
References.....	58-60

List of Abbreviations

IoT - Internet of Things
IoV - Internet of Vehicles
QoS - Quality of Service
M2M - Machine-To-Machine
VANETs - Vehicular Ad-Hoc Networks
AODV - Ad Hoc On-Demand Distance Vector
GPSR - Greedy Perimeter Stateless Routing
GPCR - Greedy Perimeter Coordinator Routing
MANET - Mobile Ad Hoc Network
C-V2X - Cellular Vehicle-To-Everything
V2V - Vehicle-To-Vehicle
V2I - Vehicle-To-Infrastructure
V2B - Vehicle-To-Business
UVI - User Vehicle Interface
DAL - Data Acquisition Layer
DML - The Data Management Layer
SML - Service Management Layer
SLAs - Service-Level Agreements
V2C - Vehicle-To-Cloud
IDPS - Intrusion Detection and Prevention Systems
OLSR - Optimized Link State Routing
DSDV - Destination-Sequenced Distance Vector
DSR - Dynamic Source Routing
ITS - Intelligent Transportation Systems
RREQ - Route Request
RREP - Route Reply
RERR - Route Error
TTL - Time To Live

List of Figures

1.1	The network model of IoV with the three network elements.....	5
1.2	Seven-layer IoV architecture.....	7
3.1	Propagation of the RREQ.....	25
3.2	Path of the RREP to the source.....	25
3.3	Node A sending message to node E.....	27
3.4	Node A generates a RREQ message with initial time TTL.....	27
3.5	Node A sets a special flag in the RREQ message.....	28
3.6	Route Discovery in DSR.....	32
3.7	Route Discovery in DSR1.....	32
3.8	Route Discovery in DSR2.....	33
3.9	Route Reply in DSR.....	33
3.10	Data Delivery in DSR.....	34
3.11	Route Error.....	35
3.12	Route Error1.....	35
3.13	Route Error2.....	35
3.14	Caches on Selected Nodes After one RREQ-RREP Cycle.....	36
3.15	Architecture OLSR Routing Protocol.....	41
4.1	Installing the packages for NS3 Tool.....	47
4.2	Creating build of the NS3 packages.....	48
4.3	Running the hello-simulator file using Ns3 command.....	48
4.4	Running the first.py in build script.....	49
4.5	Running the script for Aodv Routing Protocol.....	49
4.6	Receiving data packets in Aodv Routing Protocol.....	50
4.7	Routing table for Aodv Routing Protocol.....	50
4.8	Routing table represents Destination and Interface of each node.....	51
4.9	Analyzing packets in Aodv Routing Protocol using Wireshark.....	51
4.10	Running the script for DSDV Routing Protocol.....	52
4.11	Analyzing packets in Dsdv Routing Protocol using Wireshark.....	52

4.12	Creating build for the Dsr code script.....	53
4.13	Message type, Source ID for each node in Dsr Routing Protocol.....	53

List of Tables

1	Literature Review.....	20-21
---	------------------------	-------

Abstract

IoT and vehicular networking are integrated in the upcoming technology known as the IoV to enable intelligent transportation systems. In the Internet of Vehicles, reliable and timely data transfer between vehicles and infrastructure is one of the major issues. IoV's dynamic and heterogeneous network environment, which includes high mobility, sporadic connectivity, and scarce resources, makes network routing challenging. The report begins with a description of the idea of IoV and some of its potential applications. This is followed by a brief exposition of the core ideas behind IoV network routing. The following section of the study provides a thorough analysis of the current network routing protocols used in the IoV, including geographic routing, topology-based routing, and clustering-based routing. Data packets are routed between source and destination nodes using the well-liked routing method known as geographic routing. Using the network topology, topology-based routing chooses the optimum route between nodes, whereas clustering-based routing divides the network into clusters and chooses a cluster head to transmit data packets. The study also examines the difficulties with network routing in IoV, including security, energy efficiency, QoS provisioning, and mobility management. As nodes move quickly and frequently, the topology regularly changes, making mobility management in IoV a considerable problem. Researchers have proposed a number of mobility-aware routing strategies that consider the node's speed and direction to forecast its future location in order to address this issue. As nodes have limited battery power, energy efficiency is another major IoV difficulty. Researchers have put forth a number of energy-conscious routing protocols that choose the optimum path for data transmission while taking the energy level of the nodes into consideration. Additionally, QoS provisioning is a crucial prerequisite for IoV applications including intelligent transportation systems, entertainment, and emergency services. Researchers have put out a number of QoS-aware routing protocols that give traffic the priority it needs in order to achieve QoS. Finally, security is a major issue in IoV because the

network is open to numerous attacks such as data manipulation, denial of service attacks, and eavesdropping. Researchers have put forth a number of secure routing protocols that provide authentication, confidentiality, and packet integrity to lessen the impact of these attacks. The Internet of Things is being progressively incorporated into cars today, allowing them to give drivers and passengers everywhere access to information. However, as the number of connected vehicles keeps growing, new requirements for vehicular networks are arising (such as smooth, secure, resilient, scalable information transmission among vehicles, humans, and roadside infrastructures). The basic idea of vehicle ad hoc networks is being changed in this context into a new idea dubbed the Internet of Vehicles (IoV). In conclusion, this paper presents a thorough review of the current IoV network routing techniques, their corresponding difficulties, and potential future research areas. The conclusions of this paper can be utilized as a reference by academics and industry professionals involved in IoV network routing. The study emphasizes the requirement for the development of dependable, effective, secure, and QoS-aware routing protocols for IoV networks. To solve the issues with network routing in the IoV and to create effective and scalable routing solutions for IoV applications, more research is needed.

CHAPTER 1

INTRODUCTION

Today, as more and more people use the transport systems, many nations capacity is being strained to the breaking point. In many situations, maintaining or upgrading existing transit infrastructure is now both expensive and inefficient. According to a recent study, there are currently slightly more than one billion passenger and commercial cars in circulation globally [1] and that number is projected to rise to two billion by 2035 [2]. Traffic congestion and the number of fatalities brought on by road accidents rise when the number of vehicles increases quickly [3]. The transportation system will likely undergo significant changes in the future to accommodate the demands of passengers, drivers, and new vehicles, as well as new paradigms like the IoT and cloud computing, among other things. A broad variety of intelligent gadgets have been created recently, many of which feature embedded processors and wireless communication technologies thanks to recent advancements in computer and networking technology. Through their interconnectedness and interoperability, these intelligent gadgets are being used to create a safer and more convenient environment, giving rise to the new idea of IoT. Furthermore, as high-speed mobile Internet connectivity becomes more widely available and accessible, new potential for societal goods and services are opening up.

According to Raymond James' industry, the number of Internet-connected devices topped the global population in 2011, but it is expected to continue growing in the coming years. The IoT paradigm is made possible by the exponential rise of connected devices, which creates a variety of M2M connections that allow for pervasive connectivity across devices. The automobile sector, where M2M communication is widely used, is one of the fastest growing industries, according to Vodafone (with a rise of about 32% and 19% for the automotive and logistic and

transportation sectors, respectively). After being introduced more than ten years ago, the idea of VANETs has now become a very active topic of research in both academia and industry [6]– [9].

New requirements for VANETs, such as inter-vehicular, vehicular-infrastructure and vehicular-Internet, vehicle personal devices, and intra-vehicular communications, are emerging as more vehicles are connected to the IoT. One of the main problems with VANETs is that they can't make sense of all the information that is being gathered by themselves and other actors (such sensors and mobile devices) in the environment. In this scenario, automobiles must evolve into "smart" objects with a multisensory platform, a set of communication technologies, powerful compute units, IP-based access to the Internet, and a direct or indirect link to all other vehicles and environmental equipment. The idea of VANETs is being replaced in this context by the IoV. A VANET is based on the basic premise that an automobile is a mobile node that may connect to other cars to build a network. Vehicles become connected or disconnected as they enter or exit the coverage area. We see a VANET as a constrained network with mobility constraints since the number of linked vehicles, along with other factors like snarled traffic, tall buildings, and dangerous driving, all affect its performance and utilization.

In VANETs, there is not enough processing power to handle global information. The massive data gathered from the different cars that make up the network cannot be analyzed, processed, or evaluated by VANETs. However, VANETs are best suited for short-term applications or modest-sized services like collision prevention or warnings for traffic hazards. However, IoV combines two technological visions: 1) vehicle networking and 2) vehicle intelligence [10] emphasize the fusion of entities such as people, cars, things, networks, and environments to build an intelligent network based on computing and communication capabilities that supports services for large cities or even an entire nation, such as global traffic efficiency and management based on pollution levels, road conditions, congestion traffic level, and other factors.

By utilizing intelligent systems on vehicles and different cyber-physical systems (such as sensors, cars, and mobile devices) in cities, we may create a worldwide network that provides a variety of services to vehicles and the people linked with them. IoV also refers to people, cars, components of the transportation infrastructure, and a group of environmental gadgets that are all connected via a completely IP-based infrastructure and exchanging data in some way to improve the efficiency, safety, and environmental friendliness of transportation.

The frequent and quick change in the network architecture that occurs with the deployment of VANETs on a broad scale causes issues and unclear instructions, such as bandwidth constraints, channel instability, and communication lags. Having more nodes results in a more stable route, but having too many may result in network congestion and packet collisions. In order to reduce delays and increase efficiency and dissemination distance, a variety of techniques are used, such as cluster-based routing, geographic position, reactive, proactive, machine learning, bioinspired, probabilistic, broadcast, and unicast. However, due to VANET and vehicle telematics' limited capacity to interpret global data, an open-integrated network foundation system is necessary. As a result, the IoV is now connected to VANET, automobile telematics, and other vehicle networks. The Internet of Value (IoV) is thought to be a substantial network that can support enormous countries and cities. IoV is a very open and integrated network framework system that supports numerous users, vehicles, network organizations, and objects while keeping high administration and controllability. In order to lower social expenses, ensure human enjoyment, promote effective mobility, and improve service levels in urban areas, IoV's major goal is to interact with the environments and things related to human-vehicle interactions. Through the use of IoV technology, a shorter route may be found, saving drivers and passenger's time. There are various routing techniques that a vehicle might use to choose a better route plan. Some of the pathways, however, become incorrect as small topological changes take place. To enhance network performance and detect path failure, a unique path duration estimation routing protocol is needed before finding a new path. AODV, GPSR,

and GPCR, as well as some geographic-type protocols derived from MANET studies that take into account the characteristics of the vehicle, are used alongside several conventional routing protocols.

1.1 Problem Statement

The emergence of the Internet of Vehicles (IoV) has led to an increase in the number of connected vehicles, which has posed several challenges in terms of network routing. The IoV environment comprises vehicles, infrastructure, and other entities that communicate with each other using various communication technologies. Routing in IoV environments is critical for enabling effective and efficient communication among vehicles and infrastructure. However, traditional routing protocols designed for traditional networks may not be suitable for IoV environments due to the unique characteristics of the network, such as frequent topology changes, high mobility, and heterogeneous communication technologies. Therefore, there is a need to develop new routing protocols that are specifically designed for IoV environments. These routing protocols must take into account the characteristics of the IoV environment and provide efficient routing mechanisms that can handle the dynamic topology changes, high mobility, and heterogeneity of communication technologies. Moreover, the routing protocols must also consider the security and privacy concerns associated with the IoV environment, as the communication between vehicles and infrastructure may contain sensitive information. Therefore, there is a need to develop routing protocols that can ensure secure and private communication in the IoV environment while providing efficient routing mechanisms.

Thus, the problem statement is how to design routing protocols that are specifically tailored for the IoV environment, taking into account the unique characteristics of the network, while providing efficient routing mechanisms that can handle the dynamic topology changes, high mobility, and heterogeneity of communication

technologies. Additionally, these routing protocols must ensure secure and private communication in the IoV environment.

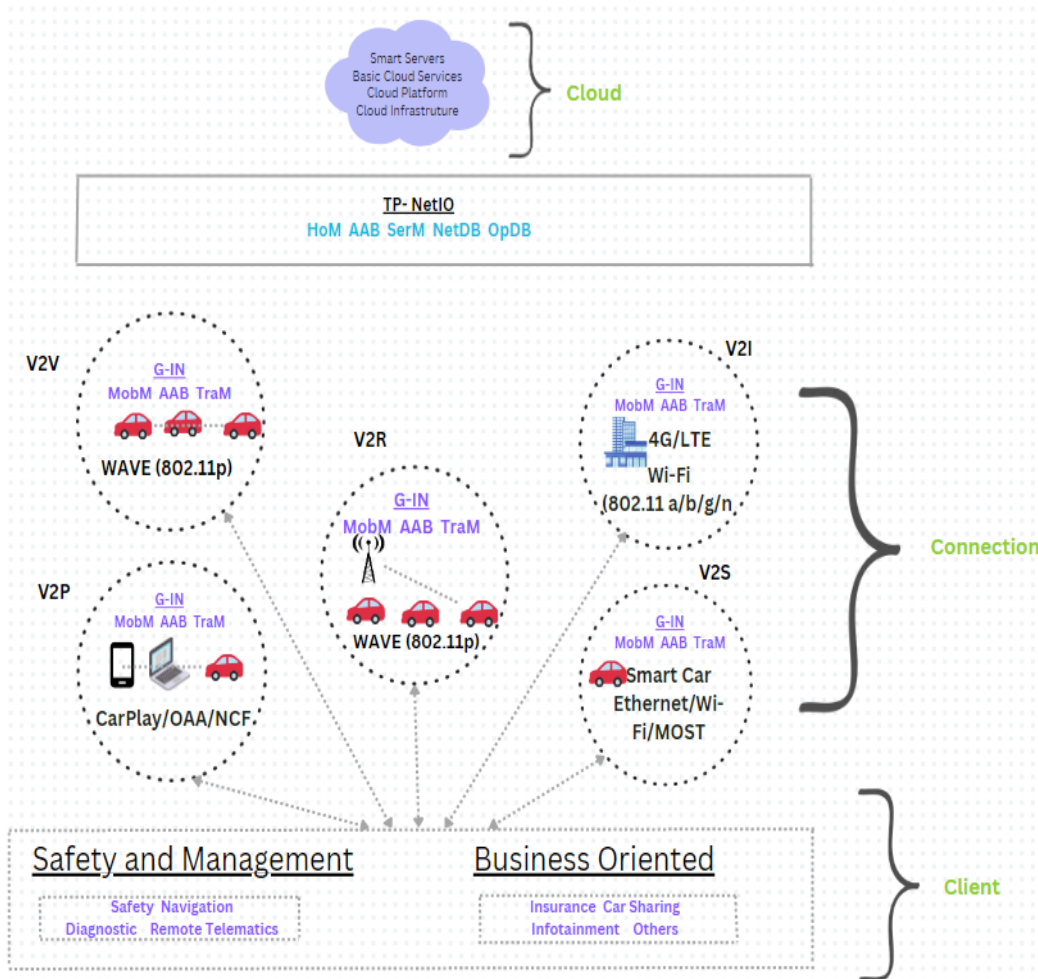


Fig: 1.1 The network model of IoV with the three network elements.

1.2 Objectives

- **To understand the concept of Internet of Vehicles (IoV) and its network architecture.**

The first objective is to understand what the Internet of Vehicles is and how it works. This includes understanding the network architecture of IoV, the different types of vehicles that can be connected to the network, and the various communication technologies used in IoV.

- **To explore the various routing protocols used in IoV environments.**

The second objective is to explore the different routing protocols that can be used in IoV environments. This includes understanding the differences between traditional routing protocols and those used in IoV, such as VANETs and C-V2X.

- **To investigate the challenges of network routing in IoV environments.**

The third objective is to investigate the challenges of network routing in IoV environments. This includes understanding the impact of mobility, node density, and topology on network routing, as well as the potential security and privacy issues that can arise.

- **To explore the use of edge computing and fog computing in IoV network routing.**

The fourth objective is to explore the use of edge computing and fog computing in IoV network routing. This includes understanding the differences between cloud computing, edge computing, and fog computing, and how these technologies can be used to optimize network routing decisions and improve overall network performance.

1.3 Iov Architecture

Researchers have discovered a three-level design based on how various technologies interact in the IoV environment. All of the car's sensors, which collect environmental data and identify particular events of interest including driving patterns, vehicle situations, and ambient variables, are located on the first level. The communication layer enables a variety of wireless communication techniques, including V2V, V2I, vehicle-to-pedestrian, and vehicle-to-sensor. The communication layer guarantees smooth connectivity to both established and developing networks (including, among others, GSM, Wi-Fi, LTE, Bluetooth, and 802.15.4).

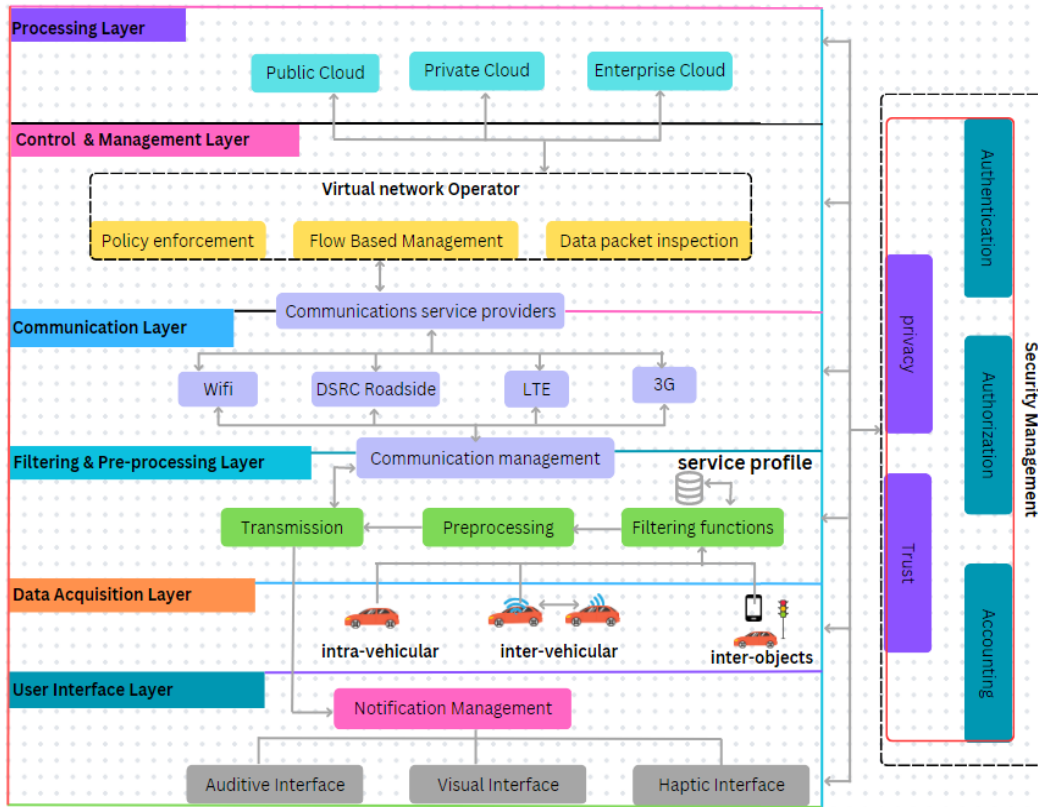


Fig: 1.2 Seven-layer IoV architecture based on [1].

Third layer is made up of statistical tools, support for storage, and processing infrastructure and is in charge of storing, analyzing, processing, and making decisions on various risk circumstances (including, among others, traffic congestion and dangerous road conditions). It is a component of the intelligence found in the IoV and provides mobile automobiles with access to large data-based processing (such as using computing resources, looking up content, sharing spectrum, etc.). The goal is to be able to synthesize data from many systems and technologies (such as big data, wireless sensor networks, cloud computing, etc.) to arrive at cohesive conclusions.

Cisco has suggested an IoV architecture with four layers. The end points layer handles the 802.11p V2V communication between software and automobiles. The infrastructure layer establishes the technological connections among all IoV actors. The operation layer keeps an eye on flow-based management and policy

enforcement. The service layer, which is the last but not the least, describes the services that the various clouds (public, private, and enterprise) offer to drivers via subscription, data center, or on-demand. IoV enables connectivity to the Internet for vehicles and drivers, providing them with access to a number of service providers. This access will facilitate the growth of V2B communication by making it simpler to combine commercial business services with autos. SAP evaluation. IoV architecture based on [1] with seven layers is presented. presented a service-based architecture and an event-driven architecture that are interdependent. According to the V2B integration architecture, the back-end integration manager links in-vehicle components while the vehicle integration platform acts as a back-end system to enable efficient information exchange between vehicles and corporate applications.

The proposed existing IoV models do not consider security (authentication, authorization, accounting, and trust relations); they do not offer a layer to integrate communication intelligence (selection of the best network for information transmission/dissemination or accessing a service); and they also restrict interaction with drivers and passengers to sending notifications through various car devices.

- **User Vehicle Interface Layer:**

In order to lessen driving distractions, this layer offers direct interaction with the driver through a management interface to organize all driver notifications and choose the optimum display element for the current scenario or event. For instance, the dashboard of the automobile may light up and generate a sound to inform the driver of a potential accident with a vehicle up ahead. UVI layer, which facilitates the integration of cars, sensors, and communication networks into a unified system, is a crucial part of the Internet of cars (IoV) architecture. The UVI layer acts as the principal point of contact between the driver or passenger and the car, offering an intuitive user interface for driving and maintaining the car. The UVI layer is made up of a number of parts, including a dashboard display, touchscreen interface, voice

controls, and mobile applications that let people communicate with the car from a distance. These parts gather information from the vehicle's many sensors and systems and deliver it to the user in an insightful manner. For instance, the dashboard display could include data on the engine temperature, fuel level, and other crucial parameters. For a more immersive and natural user experience, the UVI layer may also offer cutting-edge technologies like augmented reality displays, gesture recognition, and haptic feedback. Without taking their eyes off the road, a motorist may utilize gesture recognition to control the entertainment system or change the temperature. The UVI layer not only offers a user-friendly interface but also makes a variety of services and applications accessible. For instance, navigational systems that offer in-the-moment traffic updates and recommend detours to escape traffic. Access to entertainment systems like music and video streaming as well as security functions like lane departure and automatic emergency braking may also be made available through the UVI layer. The UVI layer also makes it possible to gather and analyze information about user behavior, vehicle performance, and environmental factors. This information can be utilized to make driving more enjoyable, cut emissions, and increase safety. For instance, a car may examine data on traffic patterns to improve its route and use less gasoline.

- **Data Acquisition Layer:**

This layer gathers information from a variety of sources that are situated on the roadways, including sensors, traffic lights, and signals as well as internal sensors and the navigation system of the car. A key element of the IoV architecture, the DAL is responsible for gathering data from various sensors and devices inside the vehicle. It is in charge of gathering and sending information about the performance, surroundings, and driving style of the vehicle to the higher layers of the architecture for processing and analysis. Numerous sensors and gadgets, including GPS, accelerometers, temperature sensors, cameras, and microphones, are included in the DAL layer. These sensors gather information on the location, speed, fuel level, engine temperature, tyre pressure, and exhaust emissions of the vehicle. Numerous uses for the information gathered by these sensors include vehicle maintenance, safety, and efficiency. The DAL layer is in charge of sending the data gathered by the sensors to the IoV architecture's top layers. The DML and the SML process and analyze the data in order to offer the vehicle and its occupants value-added services and applications. As it supplies the raw data required to optimize vehicle performance, increase safety, and lessen environmental impact, the DAL layer is crucial to the IoV design. The total reliability of the vehicle can be increased by using data from the DAL layer, for instance, to identify and treat potential maintenance problems. Additionally, by analyzing patterns in driver behavior and offering feedback to assist drivers improve their driving habits, data from the DAL layer can be utilized to optimize the vehicle's fuel use and lower emissions.

- **Data Filtering and Preprocessing Layer:**

To prevent the transmission of unnecessary information and lower network traffic, this layer analyzes the information that has been gathered. A service profile built for the car with subscribed or active services informs gearbox decisions. A crucial part of the IoV architecture, the Data Filtering and Preprocessing Layer filters and pre-processes the data that comes from the Data Acquisition Layer (DAL). The

layer is in charge of processing and analyzing the data to weed out extraneous information, spot trends, and get it ready for further analysis by the IoV architecture's upper layers.

The Data Filtering and Preprocessing Layer performs several functions, including:

Data filtering: The layer removes pointless data from the DAL data it receives. By reducing the amount of data that must be processed by the system's upper levels, this enhances the system's overall effectiveness.

Preprocessing of data: The layer preprocesses the data it receives from the DAL to make sure it is in a format that can be used. In order to do this, raw sensor data may need to be transformed into relevant information. For example, GPS data may be transformed into location information or accelerometer data into speed and direction information.

Data cleaning: The layer is in charge of cleaning the data to get rid of any flaws or inconsistencies. This contributes to ensuring the data's accuracy and dependability, which are crucial for the IoV system's efficient operation.

Data normalization: To guarantee that the data is consistent between various sensors and devices, the layer normalizes the data. This enhances the data's correctness and dependability, two factors essential to the IoV system's efficient operation.

Data transformation: The layer modifies the data to prepare it for analysis by the system's higher layers. This can entail combining data from several sensors or devices, or it might entail translating data into a format that is appropriate for the analytical tools used by the system's top levels.

Algorithms, filters, and classifiers are some of the parts that make up the Data Filtering and Preprocessing Layer. These elements perform an analysis and processing on the data to weed out irrelevant information, spot trends, and get the data ready for further analysis by the system's higher levels. Depending on the

particular requirements of the IoV system, the algorithms and filters employed by the layer can be changed.

- **Communication Layer:**

This layer chooses the optimal network to send the information through by considering a number of factors, including information relevancy, privacy, security, and QoS levels in the many networks that are available. IoV architecture, which aspires to connect vehicles, infrastructure, and pedestrians through real-time communication, must include the communication layer. The physical layer, network layer, and application layer are the three primary parts of the communication layer, which permits data interchange between various system entities.

Physical Layer: In the IoV architecture, the physical layer is in charge of supplying the physical medium for communication between various entities. It consists of the tools and equipment, such as sensors, antennas, and transceivers, that make data transfer possible. Data interchange is made possible through the physical layer's usage of a variety of communication techniques, including Wi-Fi, Bluetooth, Dedicated Short-Range Communications (DSRC), and cellular networks. Additionally, it makes sure that data is reliably transferred via the communication link.

Network Layer: In the IoV architecture, the network layer is in charge of directing data between various entities. It offers the protocols and techniques required for effective data transmission, including QoS management, congestion control, and routing protocols. The network layer additionally implements encryption and authentication techniques to guarantee the security and privacy of the data transferred. Additionally, the network layer enables both unicast and multicast communication, allowing for simultaneous data transfer to one or more entities.

Application Layer: The application layer is in charge of offering the various entities in the IoV architecture the appropriate services and applications. It comprises a variety of services, such as emergency services, infotainment, and traffic management. Additionally, application layer offers the interfaces and APIs required for programmers to build fresh applications that may be incorporated into the IoV architecture. Middleware elements are also included in the application layer, allowing for the fusion of various services and applications.

- **Control and Management Layer:**

The management of various network service providers operating within the IoV environment is the responsibility of this layer. To better handle the information received, several policies (such as packet inspection, traffic engineering, and traffic management) and functions are used at this layer. One of the key layers of the IoV architecture, the Control and Management layer is in charge of managing and controlling the overall operations of the IoV system. This layer consists of a number of elements that cooperate to guarantee the IoV system's smooth operation.

The Control and Management Layer in IoV architecture can be divided into four components:

Network Management Component: The Network Management Component is in charge of overseeing the IoV architecture's whole network infrastructure. Tasks including managing network resources, keeping track of network performance, and diagnosing network problems are included in the planning, deployment, and maintenance of networks. This part makes sure the network functions successfully and efficiently while supporting all of the apps and services offered by the IoV system.

Resource Management Component: In order to run the IoV architecture, a variety of resources must be managed by the Resource Management Component. Managing the hardware and software resources necessary for data processing, storing, and communication is a part of this process. To keep the IoV system

running smoothly and effectively, this component also manages the power and energy resources needed, such as batteries and charging stations.

Service Management Component: The IoV architecture's many services are managed by the service management component. The many services are available and can be used by the users thanks to duties like service discovery, registration, and maintenance. SLAs are also provided by this component to guarantee that the services are provided within the defined quality and performance standards.

Component for managing security: The IoV architecture's overall security is managed by the security management component. To enable safe data transmission between various system entities, it also manages systems for authentication, authorization, and encryption. Additionally, this component oversees security guidelines and procedures that guard the network from hacker assaults, data breaches, and other security risks.

In conclusion, the IoV architecture's Control and Management Layer is in charge of administering and controlling the system's overall activities. The network management component oversees the network infrastructure, the resource management component oversees the various resources needed to run the system, the service management component oversees the various services the system offers, and the security management component oversees the system's overall security. The Control and Management Layer integrates these components to guarantee that the IoV system runs effectively, dependably, and securely, giving users a safe, effective, and pleasurable transit experience.

- **Processing Layer:**

This layer uses a variety of cloud computing infrastructures, both locally and remotely, to process massive amounts of data. Massive data service providers might use the information's outcomes to better develop their services or create new apps. The processed data can also be used by various government organizations to create new infrastructure, V2B services, and regulations that will help control or improve

road traffic. In the IoV architecture, the Processing Layer is in charge of processing the data gathered from different sensors and devices, analyzing the data, and making decisions based on the analysis' findings. The Processing Layer consists of a number of parts that work together to provide the IoV system the data processing capabilities it needs.

The Processing Layer can be divided into three main components:

Data Processing Component: The IoV system's data processing component is in charge of processing the information gathered from various sensors and devices. In order to do this, data must be gathered, filtered, aggregated, and stored in the proper data storage systems. In order to prepare the raw data for analysis, this component also performs data pre-processing operations such as cleaning, normalizing, and transformation. The consistency and quality of the data gathered from various sources is another duty of the data processing component.

Data Analytics Component: The Data Analytics Component is in charge of examining the information gathered by the Data Processing Component in order to draw out pertinent conclusions and details. To find patterns, trends, and anomalies in the data, various data mining, machine learning, and statistical techniques may be used. Predictive analytics capabilities, which may be used to estimate future events based on historical data, are also a part of the Data Analytics Component.

Making decisions based on the outcomes of the analysis carried out by the Data Analytics Component is the responsibility of the decision-making component. This includes making informed decisions regarding traffic management, route optimization, vehicle maintenance, and other elements of the IoV system using the insights and information gleaned from the data analysis. The decision-making component may also have automation features, which allow the system to decide and act automatically in accordance with set rules and policies. In conclusion, the IoV architecture's Processing Layer is in charge of processing data gathered from various sensors and devices, analyzing the data, and making decisions based on the

findings of the study. Data is collected, processed, and analyzed by the Data Processing and Data Analytics components, and decisions are made by the Decision Making and Decision-Making components using the findings of the analysis. The Processing Layer combines these parts to give the IoV system the data processing capabilities it needs to function effectively, dependably, and securely.

- **Security Layer:**

This transversal layer can communicate with the other levels directly. Within the suggested architecture, it is in charge of all security operations (including data authentication, integrity, nonrepudiation, and confidentiality, access control, and availability, among others). The layer is made to support countermeasures against many kinds of security assaults (including cyberattacks and others) in the IoV. An essential part of the IoV design, the Security Layer is in charge of making sure the IoV system runs securely. The Security Layer is made up of a number of parts that cooperate to offer strong security procedures and defend the system from various security risks.

The Security Layer can be divided into four main components:

Component for authentication and authorization: This component is in charge of making sure that only authorized parties may access the IoV system. This entails identifying and authenticating people, vehicles, and other entities that interact with the system. Additionally, this component enforces access control regulations, defining which resources and operations each entity may access.

Data encryption: To prevent unauthorized access to data transmitted via the IoV network, the encryption component is in charge of encrypting the data. Encrypting data both in transit and at rest helps to protect sensitive data from being hacked. V2V, V2I, and V2C communications are just a few of the channels that can use encryption.

Integrity Component: The Integrity Component is in charge of making sure that the data sent over the IoV network is transferred accurately. This includes spotting and avoiding data forgery, modification, or manipulation. This part makes sure that the data the IoV system receives is accurate and hasn't been tampered with while in transit.

Security Monitoring Component: The Security Monitoring Component is in charge of keeping an eye out for security flaws and threats in the IoV system. This includes identifying any security problems, such as cyberattacks, data breaches, and other security breaches, and notifying the system administrator. IDPS, which can identify and stop unauthorized access to the system, are also a part of this component. In conclusion, the Security Layer in the IoV architecture is in charge of making sure the IoV system runs securely. The components of the system include authentication and authorization, encryption, integrity, and security monitoring. The encryption component encrypts data sent over the network and ensures the data's integrity. The integrity component also monitors the system for security threats and vulnerabilities. These components are integrated into the Security Layer, which offers strong security procedures to defend the IoV system from numerous security threats and ensure its safe and secure functioning.

LITERATURE SURVEY

In the literature chapter, we discuss various techniques and protocols proposed for network routing in the Internet of Vehicles (IoV) environment. This include both traditional routing protocols such as the Ad-hoc On-demand Distance Vector (AODV) and Dynamic Source Routing (DSR), as well as newer protocols developed specifically for IoV such as Vehicular Ad-hoc Network (VANET) routing protocols. This chapter makes use of a number of academic papers from organizations including IEEE, Cisco, Changsha University, Google Inc., University of Illinois, etc. Detailed discussion about research papers studied for this project is mentioned below.

- 1) A. S. Al-Fuqaha, M. Guizani, M. Mohammadi, MAldhari, and M. Ayyash. "**A Survey of Routing Protocols in the Internet of Vehicles.**" IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2377-2395, 2015.

This survey gives a general overview of the IoV environment's routing protocols. The authors talk about the difficulties and demands of routing in the IoV environment, including QoS, mobility, and security. Geographic routing, cluster-based routing, zone-based routing, and QoS-based routing are only a few of the routing protocols covered in the survey. The performance of these protocols is also contrasted by the authors using measures like packet delivery ratio, end-to-end delay, and network overhead.

- 2) H. Menouar, A. Mellouk, and L. Moussaoui. "**Routing Protocols for Vehicular Ad Hoc Networks: A Survey.**" IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 559-580, 2016.

This study focuses on the routing protocols used in the IoV environment's subset known as VANETs. The authors talk about the needs and difficulties with routing in VANETs, including mobility, safety, and QoS. Position-based routing, topology-based routing, and QoS-based routing are only a few of the routing methods covered in the survey. The authors also evaluate the effectiveness of different protocols using metrics including throughput, packet delivery ratio, and end-to-end delay.

- 3) H. Qi, X. Zhang, Y. Yang, and Y. Sun. "**Routing in the Internet of Vehicles: A Comprehensive Survey.**" IEEE Internet of Things Journal, vol. 5, no. 1, pp. 672-695, 2018.

The routing protocols utilized in the IoV environment are thoroughly reviewed in this survey. The authors talk about the difficulties and demands of routing in the IoV context, including scalability, heterogeneity, mobility, and security. Geographic routing, cluster-based routing, zone-based routing, and QoS-based routing are only a few of the routing protocols covered in the survey. The performance of these protocols is also contrasted by the authors using measures like packet delivery ratio, end-to-end delay, and network overhead. The report also covers the incorporation of AI and ML approaches in routing as well as the upcoming research initiatives in this area.

- 4) M. R. Akbari, S. S. Bhatti, M. Q. Akbar, and J. Zhang. "**A Comprehensive Survey of Routing Protocols in Internet of Vehicles (IoV): Challenges and Solutions.**" IEEE Access, vol. 8, pp. 183207-183241, 2020.

This study offers a thorough analysis of the IoV environment's routing protocols as well as its problems and potential remedies. The authors talk about the needs and difficulties of routing in the IoV context, including scalability, heterogeneity, mobility, and security. Geographic routing, cluster-based routing, zone-based routing, and QoS-based routing are only a few of the routing protocols covered in the survey. The performance of these protocols is also contrasted by the authors

using measures like packet delivery ratio, end-to-end delay, and network overhead. The survey also examines new developments in routing in the IoV context, such as edge computing- and blockchain-based routing.

Study	Routing Protocols Compared	Scenario	Key Findings/Comparison
Al-Fuqaha et al. (2015)	AODV, DSDV, DSR, OLSR	Highway environment	Regarding the percentage of delivered packets and end-to-end delay, OLSR outperformed other protocols.
Menouar et al. (2016)	AODV, DSDV, DSR, OLSR	Urban environment	In terms of packet delivery percentage and routing overhead, OLSR performed better than other protocols.
Dahiya and Singh (2017)	AODV, DSDV, DSR, OLSR	Urban environment	Regarding the percentage of delivered packets and end-to-end latency, OLSR outperformed other protocols.
Qi et al. (2018)	AODV, DSDV, DSR, OLSR	Urban and suburban environment	In terms of throughput, end-to-end delay, and packet delivery ratio, AODV outperformed comparable protocols.
Dung and Nguyen (2018)	AODV, DSDV, DSR, OLSR	Highway and urban environment	When it came to routing overhead and packet delivery ratio, AODV fared better than other protocols.
Basu and Maity (2018)	AODV, DSDV, DSR, OLSR	Urban environment	In terms of packet delivery ratio and end-to-end delay, DSDV outperformed other protocols.
Islam and Hossain (2019)	AODV, DSDV, DSR, OLSR	Urban environment	Regarding the number of delivered packets, end-to-end delay, and routing overhead, AODV performed better than other protocols.

Khan et al. (2020)	AODV, DSDV, DSR, OLSR	Urban environment	In terms of packet delivery percentage, end-to-end delay, and throughput, AODV outperformed comparable protocols.
Cui et al. (2020)	AODV, DSDV, DSR, OLSR	Urban environment	Regarding the percentage of delivered packets and end-to-end latency, ADV outperformed other protocols.

Table 1: Comparisons for network routing in Internet of Vehicles.

- 5) R. S. Dahiya and D. Singh. "**A Survey of Routing Techniques in the Internet of Vehicles.**" *Wireless Personal Communications*, vol. 97, no. 2, pp. 2633-2663, 2017.

An overview of the routing methods employed in the IoV environment is given in this survey. High mobility, dynamic topology, and heterogeneity are a few of the requirements and challenges of routing in the IoV environment that are covered by the writers. Geographic routing, cluster-based routing, zone-based routing, and QoS-based routing are only a few of the routing methods covered in the survey. The performance of various strategies is further contrasted by the authors using measures like packet delivery ratio, end-to-end delay, and network overhead. The poll also analyzes the future directions for this field of study.

- 6) D. D. Dung and T. N. Nguyen. "**A Survey of Routing Protocols in Vehicular Ad Hoc Networks.**" *Journal of Computer Science and Cybernetics*, vol. 34, no. 2, pp. 125-142, 2018.

This study focuses on the routing protocols used in the IoV environment's subset known as VANETs. High mobility, safety, and QoS are a few of the needs and difficulties of routing in VANETs that are covered by the writers. Position-based routing, topology-based routing, and QoS-based routing are only a few of the routing methods covered in the survey. The authors also evaluate the effectiveness

of different protocols using metrics including throughput, packet delivery ratio, and end-to-end delay. The report also analyzes the shortcomings of the current routing protocols and the future directions for this area of study.

- 7) H. Khan, S. A. Hassan, and S. M. Riaz. "**A Comprehensive Survey of Vehicular Ad Hoc Network: Communication, Security, and Routing.**" *Wireless Personal Communications*, vol. 111, no. 3, pp. 1795-1827, 2020.

VANETs, a subset of the IoV environment, and their communication, security, and routing elements are thoroughly reviewed in this paper. High mobility, safety, privacy, and QoS are just a few of the criteria and difficulties that the authors cover when it comes to communication, security, and routing in VANETs. Position-based routing, topology-based routing, and QoS-based routing are only a few of the routing methods covered in the survey. The authors also evaluate the effectiveness of different protocols using metrics including throughput, packet delivery ratio, and end-to-end delay. The survey also talks about recent developments in VANETs, like software-defined networking and edge computing.

- 8) Y. Cui, Y. Xiao, and J. Liu. "**A Survey of Routing in the Internet of Vehicles: Technical Challenges and Future Directions.**" *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 385-402, 2020.

This survey offers a thorough review of the routing strategies employed in the IoV environment, along with a look at the technological difficulties and upcoming research prospects. The authors talk on the IoV environment's traits, including resource limitations, rapid topological changes, and high mobility. Numerous routing strategies are covered in the survey, such as geographic routing, cluster-based routing, and QoS-based routing. The authors also go through the techniques' limitations and how future research should be conducted to solve them. The poll also emphasizes how crucial it is to take security and privacy concerns into account when designing routing protocols for the IoV environment.

- 9) S. M. Islam and M. S. Hossain. "**A Comprehensive Survey of Routing Protocols in VANETs: Challenges and Future Directions.**" *Journal of Network and Computer Applications*, vol. 132, pp. 50-76, 2019.

In-depth analysis of the problems and future trends in routing protocols used in VANETs, a subset of the IoV environment, are provided in this survey. The authors talk about the qualities of VANETs, including their high mobility, sporadic connectivity, and communication security. Position-based routing, topology-based routing, and QoS-based routing are only a few of the routing methods covered in the survey. The authors also evaluate the effectiveness of different protocols using metrics including throughput, packet delivery ratio, and end-to-end delay. The study also explores future research directions to solve the difficulties associated with VANET routing, including location privacy and resource limitations.

- 10) P. Basu and M. Maity. "**A Survey of Routing Protocols for Intelligent Transportation Systems.**" *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2042-2071, 2018.

This survey provides an overview of routing protocols used in ITS, which include VANETs and other technologies used in the IoV environment. The authors discuss the characteristics of ITS, such as high mobility, safety, and QoS. The survey covers several routing protocols, including position-based routing, topology-based routing, and QoS-based routing. The authors also compare the performance of these protocols based on metrics such as packet delivery ratio, end-to-end delay, and throughput. Additionally, the survey discusses the challenges of ITS routing, such as resource constraints, security, and privacy, and the future research directions to address these challenges.

NETWORK ROUTING PROTOCOLS

1) AODV - Ad Hoc on Demand Distance Vector:

AODV uses distance vector routing to determine routes between nodes. Every node maintains a routing table with information on the nodes in its immediate vicinity and the routes that can be taken to reach them. When a node needs to connect to a target node, a RREQ packet is broadcast to neighbors. The RREQ is flooded throughout the network until it reaches the destination node or a node with a sufficiently fresh path there. When a source node receives a RREP packet from a destination node or an intermediary node with a functional route, the route has been established. The AODV protocol only builds routes between nodes when requested by source nodes. As a result, AODV is thought of as an on-demand algorithm that does not increase network traffic for communication. The routes are kept up to date as long as the sources want them. They also create trees that connect the members of a multicast group. To ensure route freshness, AODV employs sequence numbers. They are loop-free and self-starting, and they grow to numerous mobile nodes. Until connections are made in AODV, networks are silent. Network nodes that require connections post a request for connections. The other AODV nodes forward the message and record the source node of the connection request. They thus construct numerous temporary pathways to the node making the request. A node that receives such messages and holds a route to the desired node sends a backward message to the inquiring node through temporary routes. The requesting node follows the path that involves the fewest hops across other nodes. The entries in routing tables that are not used after a while are recycled. In the event that a link fails, the procedure is repeated, and the sending node receives a routing error.

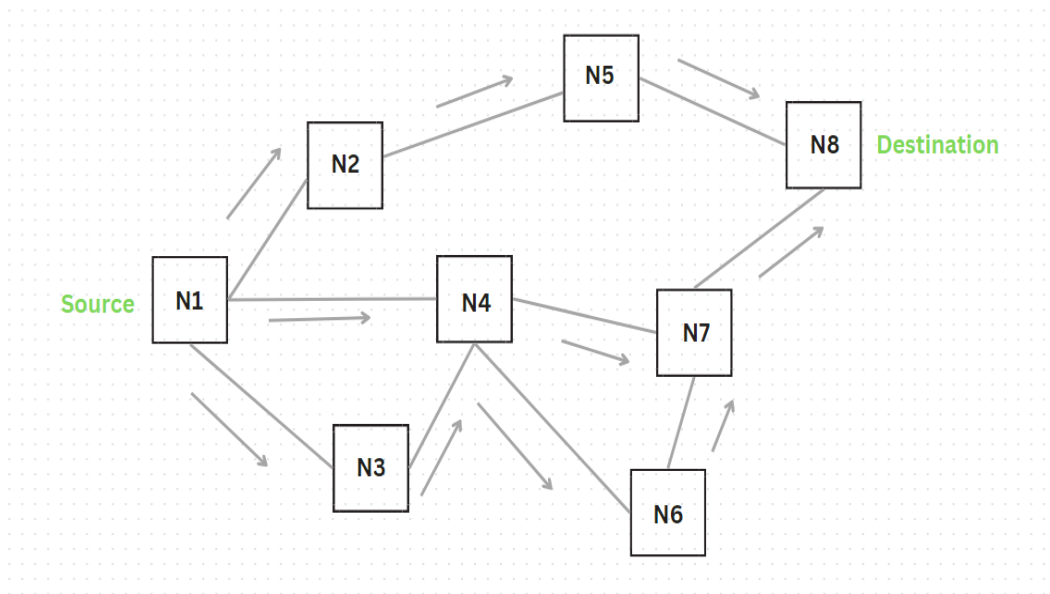


Fig: 3.1 Propagation of the RREQ.

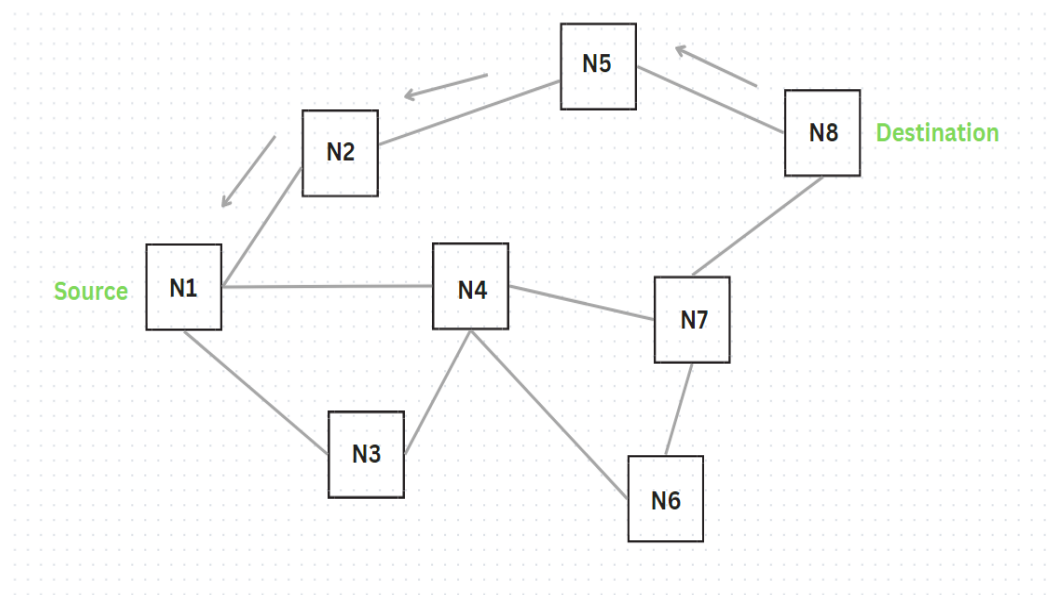


Fig: 3.2 Path of the RREP to the source.

Route Maintenance: To address changes in network topology, AODV employs a route maintenance mechanism. RERR messages are used by AODV to alert

impacted nodes to update their routing tables in the event that a connection in the established route fails or a new node wishes to join the network. To avoid routing loops and stale routes, AODV also employs a sequence number system.

Scalability: In large ad hoc networks with changeable topologies, AODV is intended to be scalable. By creating routes only, when necessary, it reduces the overhead associated with route discovery and management. Additionally, it employs a "soft state" strategy that reduces the storage needs at each node by having routing information expire after a predetermined period if it is not refreshed.

Loop Freedom: To avoid routing loops, AODV employs a sequence number technique. Only routes with greater sequence numbers are thought to be valid; each node keeps a sequence number for each destination. A node avoids routing loops by updating its routing table with the current sequence number and discarding routes with lower sequence numbers when it receives a route request or route reply.

Interoperability: The AODV routing protocol is open-standard and compatible with various routing protocols. It allows for flexibility in network design and can be utilized in mixed networks where different nodes may be using various routing protocols.

Security: Because AODV lacks integrated security features, it is susceptible to a number of attacks, such as route spoofing, route flooding, and black hole attacks. However, numerous studies are being done to improve AODV's security through the use of encryption, authentication, and other security mechanisms.

Research and Applications: AODV has received a lot of attention in the field of wireless ad hoc networks and has been utilized in a variety of applications, such as military networks, disaster recovery scenarios, sensor networks, and vehicle networks. To increase AODV's functionality, security, and scalability in various situations, researchers have suggested a number of adjustments and improvements.

Popular reactive routing technology AODV for wireless ad hoc networks establishes routes between nodes using a distance vector technique and on-demand route finding. It is interoperable, scalable, and loop-free but does not have built-in security features. Researchers continue to suggest improvements to AODV to improve its performance and security in various settings despite the fact that it has been extensively studied and deployed in a variety of applications.

- Node A wants to send a message to node E.
- A valid route must be created between A and E.

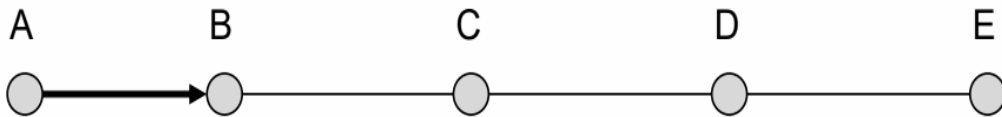


Fig: 3.3 Node A sending message to node E.

- Node A generates a RREQ message with initial time to live TTL of 1 and broadcast it to its neighbors. (In this case node B).
- The IP addresses of node A and node E are among the other things that are contained in the message.
- Node B will send an RREP message back to node A IF it has an active route to node E.

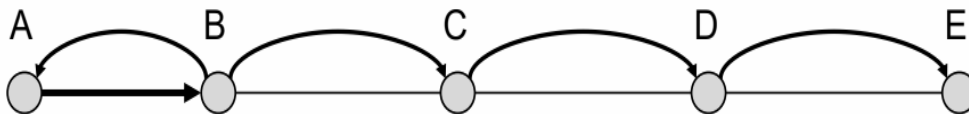


Fig: 3.4 Node A generates a RREQ (Route Request) message with initial time to live TTL.

- If A sets a special flag in the RREQ message, node B will also send a “gratuitous” RREP message to node E.
- If node B needs to establish a TCP connection with A in order to transmit packets back, then this will be required.
- If it is a gratuitous RREP, RREP messages are unicast to the following hop in the direction of the originator or destination.
- A will rebroadcast the RREQ message with an increased TTL value if it does not receive an RREP message within a predetermined amount of time.

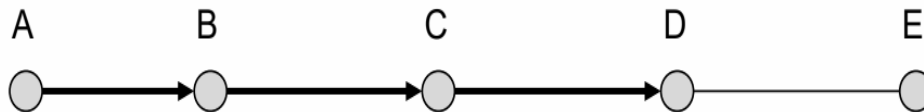


Fig: 3.5 Node A sets a special flag in the RREQ message.

- Default increment is 2
- As RREQ messages are forwarded, "reverse" routes to the originator, in this case node A, are established.
- When A receives an RREP message, an active route is created.
- The network utilization is kept low by this behavior (incrementing TTL).

Maintaining Sequence Numbers

- Sequence numbers must be properly preserved in order to prevent the "counting to infinity" issue and keep the AODV from looping.
- Forwarding nodes only change the recorded sequence number for a particular destination when sending RREP messages if:
 - The routing table's sequence number is incorrect, or
 - When compared to the stored number, the sequence number in the RREP message is higher, or
 - Even though the route is identified as inactive, the sequence numbers are the same.

- The RREP message's hop count is lower even if the sequence numbers are the same.
- Nodes must increase their own sequence number prior to broadcasting an RREQ message.
- Destination nodes increase their sequence numbers when the RREQ's sequence number and the one they have stored match.

Link Breaks:

- Using accessible data link or network layer techniques, nodes can maintain track of connectedness to neighbors.
- RERR message processing is initiated when:
 - Node discovers a broken link in the subsequent hop of an active route, or
 - Receives a packet of data going to a node for which it has no (active) route,
 - A neighbor sends an RERR message for at least one active route in the receiving router's routing table.
- Before transmitting to nodes in the precursor list, nodes must increase the destination sequence numbers of the routing elements provided in the RERR message.
- Receiving nodes only need to compare the message's sequence numbers with their own.
- Any node, whether sending or receiving, must also flag these routing entries as invalid.
- This provides loop-freedom by ensuring that no predecessors may respond to an RREQ from a node on their successor path.
- In the end, Route Requests (RREQ) messages are returned to the sender, who may then start another RREQ message.

Local Repairs:

- Nodes detecting a link breakage can choose to repair the link if possible.

- Simple broadcasting of an RREQ message and an increase in the target sequence number are all that the node does.
- The repair was successful if it receives an RREP message.

Security Considerations:

- Currently AODV has no security measures built in.
- If the network membership is known, AODV control messages can be verified as genuine.

2) DSR - Dynamic Source Routing:

When utilizing the source-initiated method of DSR, the source node chooses the complete route to the destination node and includes it in the packet header. A source node checks its routing cache to see if it has a workable route before forwarding a packet to a destination node. If not, it sends an RREQ packet to its neighbors to begin the route discovery process. The RREQ is flooded throughout the network until it reaches the destination node, or a node with a sufficiently fresh path there. A route is created when a RREP packet is sent back to the source node from a destination node or an intermediary node that has a functional route.

Route Maintenance: To address changes in network topology, DSR employs a route maintenance mechanism. RERR packets are used by DSR to alert impacted nodes to update their routing caches in the event that a connection in the established route fails or a new node wishes to join the network. DSR also employs a sequence number technique to avoid stale and looping routes.

Scalability: In large ad hoc networks with changeable topologies, DSR is intended to be scalable. By creating routes only, when necessary, it reduces the overhead associated with route discovery and management. Additionally, it employs a "soft state" strategy that reduces the storage needs at each node by having routing information expire after a predetermined period if it is not refreshed.

Loop Freedom: To avoid routing loops, DSR employs a sequence number technique. For each destination, each node keeps a sequence number, and only routes with higher sequence numbers are taken into consideration as genuine. A node updates its routing cache with the current sequence number and discards routes with lower sequence numbers when it receives a route request or route reply in order to prevent routing loops.

Interoperability: DSR may work with different routing systems because it is an open standard routing protocol. It enables flexibility in network architecture by enabling use in mixed networks, where different nodes may employ various routing protocols.

Security: Because DSR lacks integrated security features, it is susceptible to a number of attacks, including as route spoofing, route flooding, and black hole attacks. However, numerous studies are being done to improve DSR security by the use of encryption, authentication, and other security mechanisms.

Research and Applications: Mobile ad hoc networks, sensor networks, and Internet of Things (IoT) networks are just a few examples of the wireless ad hoc networks for which DSR has been extensively studied and employed. DSR has undergone a number of adjustments and improvements that researchers have suggested to boost its functionality, security, and scalability in many situations. DSR is a well-liked reactive routing technology for wireless ad hoc networks that establishes routes between nodes using a source-initiated method and on-demand route discovery. It is interoperable, scalable, and loop-free but does not have built-in security features. Researchers continue to suggest improvements to DSR to improve its efficiency and security in various settings despite the fact that it has been the subject of extensive research and is utilized in many applications.

- When node S needs to deliver a packet to node D but is unsure of the route there, node S starts a route discovery process.

- Source node S continuously sends route request RREQ packets, also known as query packets, to the network.
- In the packet header, each node adds its own address before transmitting the RREQ.

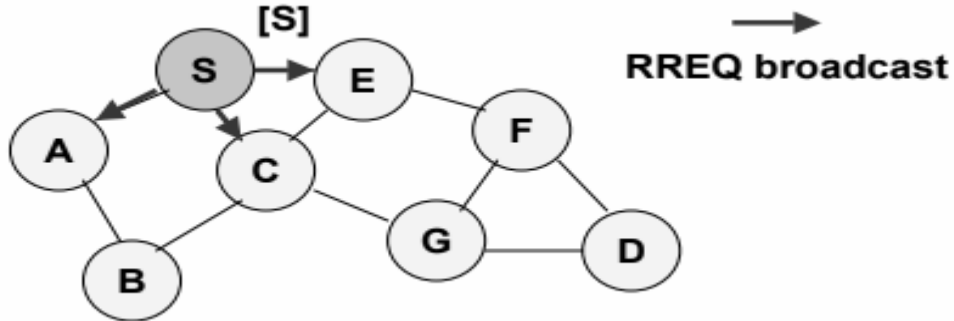


Fig: 3.6 Route Discovery in DSR.

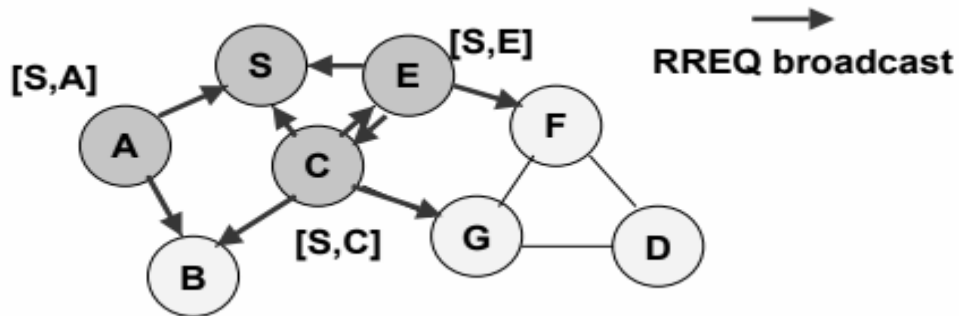


Fig: 3.7 Route Discovery in DSR1.

In Fig:3.6,3.7, The dark node is an instance of a node that has been provided with RREQ for D by S. [X,...] is a list of addresses that was appended to RREQ. Once a node receives an RREQ, it broadcasts it once more.

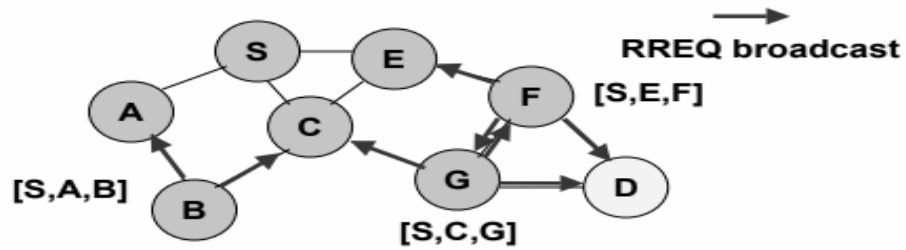


Fig: 3.8 Route Discovery in DSR2.

In Fig: 3.8, Destination D receives RREQ via G and F. It does not broadcast it further.

Route Discovery in DSR:

- After receiving the initial RREQ, Destination D sends a Route Reply (RREP).
- The route used to send RREP is created by reversing the path used to send RREQ.
- The reverse route from S to D, on which node D got RREQ, is included in RREP.

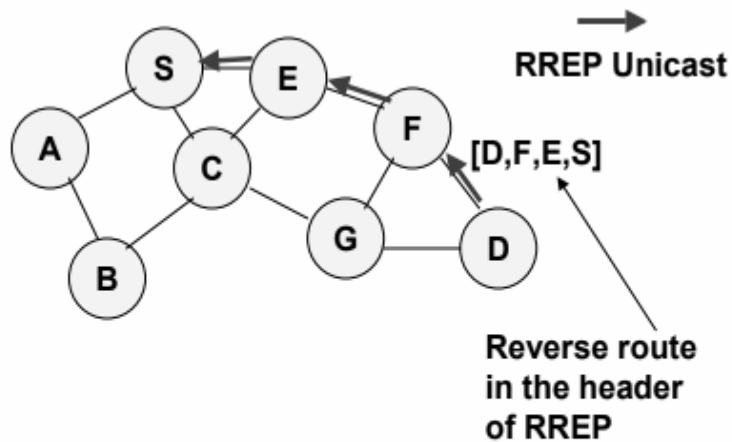


Fig: 3.9 Route Reply in DSR.

Route Caching in DSR:

- Node 'S' on receiving RREP, "caches" the route included in the RREP.
- The complete route is contained in the packet header when node S transmits data to node D.
- Hence the name source routing.
- The source route contained in a packet is used by intermediate nodes to decide who a packet should be forwarded to.

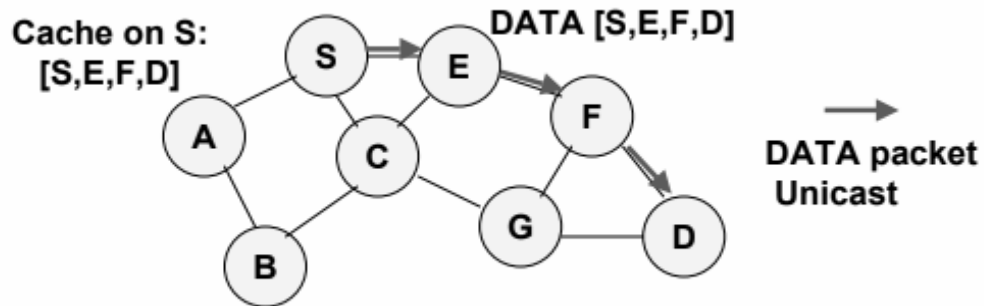


Fig: 3.10 Data Delivery in DSR.

In Fig: 3.10, Source route size grows with route length.

Route Error:

- When a data packet is forwarded, a RERR is produced and propagated backwards if the next hop link is broken.
- RERR contains the failed link info.
- Any cached route containing the failed link is removed by S when it receives RERR.

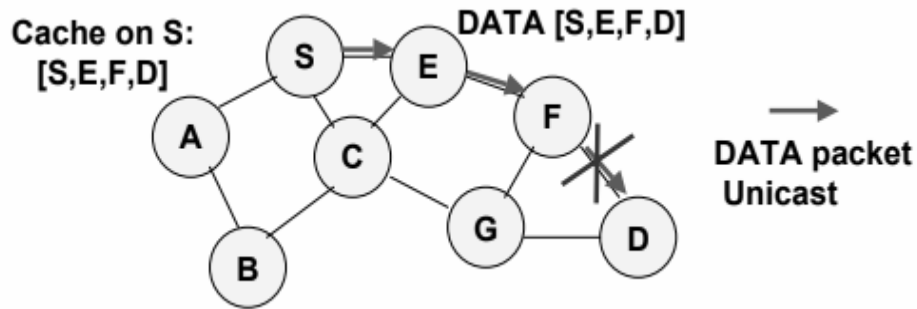


Fig: 3.11 Route Error.

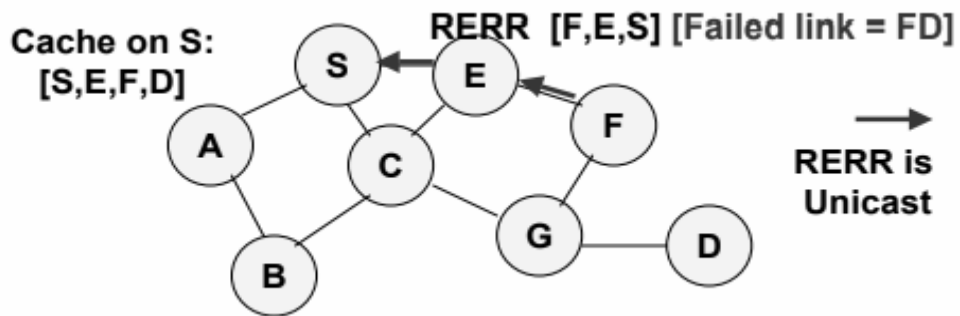


Fig: 3.12 Route Error1

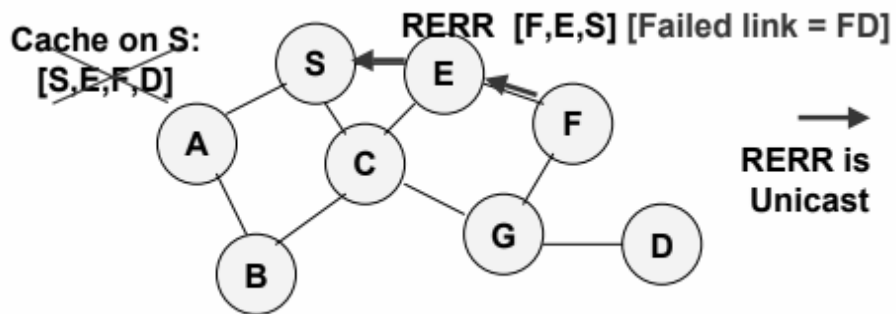


Fig: 3.13 Route Error2

Aggressive Route Caching:

- Every node stores a new route it discovers in its cache.
- Node S learns the path [S, E, F] to node F after discovering the route [S, E, F] to node D.

- Node G learns the route [G, C, S] to node S when it gets RREQ [S, C] intended for node D.
- The route [F, D] to node D is learned by node F when it forwards RREP [D, F, E, S].
- To put it simply, when a packet is forwarded, every node in the source route contained in the packet is learned by the receiving node.

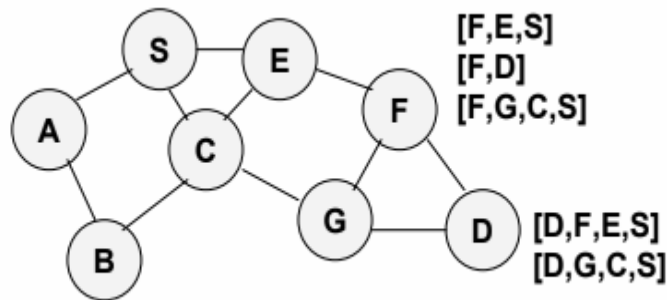


Fig: 3.14 Caches on Selected Nodes After one RREQ-RREP Cycle.

Contents of Caches on Selected Nodes After one RREQ-RREP Cycle:

- A cached route is represented by [P, Q, R] at node P.
- For the same destination, more than one route may be cached.
- Route caches can be implemented using small data structures, such trees.

Use of Route Caching:

Salvaging: Node S uses a different route from its local cache if one exists to node D when it finds that a route to node D is broken. If not, node S starts the route discovery process by submitting a route request.

Reply from Cache: If Node X knows a route to Node D, it can deliver a Route Reply in response to getting an RREQ for some Node D.

Aggressive use of route cache: Can speed up route discovery. or can reduce propagation of route requests.

Dynamic Source Routing: Advantages:

Source routing: No special mechanism needed to eliminate loops.

On demand routing: Only routes between nodes that require communication are kept. lowers route maintenance costs.

Route caching: Can lower overhead associated with route discovery. As a result of intermediate nodes responding from nearby caches, a single route discovery may reveal numerous paths to the desired destination. when the path breaks, useful.

Dynamic Source Routing Disadvantages:

Stale cache problem: An intermediary node might contaminate other caches by sending a route reply using an outdated cached route. If a mechanism to remove (possibly) invalid cached routes is included, this issue can be mitigated.

Current research: How to invalidate caches effectively. Example: Timer-based. Or propagate the route error widely.

3) DSDV - Destination Sequenced Distance Vector:

Routing Method: DSDV is a proactive or table-driven routing protocol, in which each node keeps track of a routing table that details the paths to all destinations that can be reached in the network. To keep the routing information current, the routing table is updated as needed or at regular intervals. Each item in the routing table contains the address of the destination node, the quantity of hops required to get there, and a sequence number.

Sequence Numbers: Sequence numbers are used by DSDV to distinguish between outdated and up-to-date routing information. The sequence number for that destination is increased each time a node modifies its routing database. A node considers a routing update to be fresh and updates its routing database when it receives one from a neighbor with a higher sequence number. This guarantees that nodes have access to the most recent routing information and helps prevent routing loops.

Route Selection: The minimal hop count statistic is used by DSDV to choose routes, with the route with the fewest hops being the recommended route.

Nevertheless, depending on the needs of the network, DSDV can also be set up to use different metrics, such as bandwidth or link quality.

Periodic Updates: In order to guarantee that every node in the network has consistent routing information, DSDV regularly broadcasts its routing table to its neighbors. Depending on the dynamics of the network and the desired level of overhead, the frequency of updates can be changed.

Route Maintenance: Based on modifications to the network topology, DSDV updates its routing table. A node notifies its neighbors of changes by sending an update whenever a link fails or a new node joins the network. As a result, the affected nodes update their routing tables in a process known as a route update.

Scalability: Due to the frequency of updates and the size of the routing tables, DSDV may have scalability problems. The overhead of updates and the storage needs at each node rise together with the network growth. To increase the scalability of DSDV, academics have suggested a number of optimizations, including partial updates and route aggregation.

Research and Applications: Numerous wireless ad hoc network applications, such as military networks, vehicle networks, and mobile ad hoc networks, have made extensive use of DSDV. To enhance DSDV's functionality, scalability, and situation adaptability, researchers have suggested improvements and enhancements.

DSDV is a proactive routing technique that keeps track of routing information in wireless ad hoc networks using a table-driven methodology and regular updates. It offers route selection based on hop count or other criteria and employs sequence numbers to guarantee current routing information. The performance of DSDV has been the subject of much research and use in numerous applications, and academics continue to suggest improvements and optimizations to raise that performance in various contexts.

A table-driven routing protocol called the Destination-Sequenced Distance Vector (DSDV) protocol is based on an enhanced version of the traditional Bellman-Ford routing algorithm. The Routing Information Protocol (RIP), on which DSDV is based, allows a node to maintain a routing table with all potential network destinations and the number of hops needed to reach each one. DSDV uses bidirectional connectivity and is also based on distance vector routing. The fact that DSDV only offers one route for a source/destination pair is one of its drawbacks.

Routing Tables:

This protocol's routing table has an easy-to-understand structure. The associated sequence number for each database entry rises each time a node sends an updated message. To ensure consistent data as the topology of the network changes, routing tables are regularly updated and transmitted throughout the network. Each DSDV node maintains two routing tables: one for incremental routing packet advertising and the other for packet forwarding. Routing data sent by a node includes the destination address, the number of hops required to get there, a new sequence number, and the destination's sequence number. When the topology of a network changes, a detecting node updates its neighbors by sending them a packet of information. After taking information from an update packet, it receives from a neighboring node, a node adjusts its routing table in the manner described below.

DSDV Packet Process Algorithm:

1. If the new address has a higher sequence number, the node chooses the route with the higher sequence number and discards the route with the lower sequence number.
2. If the entering sequence number matches that of an existing route, the least expensive route is selected.
3. The metrics that were selected based on the updated route information are all increased.
4. This process continues until all nodes have received the updates. If there are duplicate updated packets, the node weighs keeping the one with the lowest cost metric and rejecting the others.

The DSDV protocol's packet overhead raises the overall number of nodes in the ad hoc network. Because of this, DSDV is appropriate for small networks. Large ad hoc networks experience increased overhead and mobility rates, which cause the network to become unstable to the point where updated packets may not arrive at nodes in a timely manner.

4) OLSR - Optimized Link State Routing

For MANETs and wireless mesh networks, the proactive routing protocol OLSR was created. It is based on the idea of link-state routing, in which every node in the network keeps track of the connectedness of every other node. The shortest path

between two network nodes is calculated using this information. The OLSR protocol is made to reduce network overhead and the quantity of packet transmissions necessary to keep the network connected. This is accomplished by choosing a subset of network nodes to serve as relays using a multi-point relay (MPR) selection method. By forwarding control messages to their neighbors, these MPR nodes minimize the number of transmissions necessary to maintain network connectivity. To lessen the quantity of control traffic in the network, OLSR additionally employs a method referred to as MPR. In this method, each network node chooses a group of MPR nodes to relay its control signals to other nodes in the network [5]. The volume of control traffic in the network is decreased by OLSR by lowering the number of nodes that must receive control messages. The usage of a "hello" message by each node in the network to announce its presence and share details about its neighbors is another significant aspect of OLSR. This enables each node to keep track of a list of its neighbors and the state of their connectivity. In comparison to other routing protocols, OLSR provides a number of benefits, including the capacity to accommodate vast networks with a high level of mobility, low overhead, and quick topology adaptation. It does have some drawbacks, though, like the fact that it depends on a centralized MPR selection process, which might lead to scalability concerns in very big networks. A proactive routing system called Optimized Link State Routing (OLSR) employs link-state routing and MPR selection to cut down on network overhead and the quantity of packet transmissions necessary to keep the network connected. It is intended for MANETs and wireless mesh networks, and it has a number of benefits and drawbacks to take into account when selecting a routing protocol for a certain network.

OLSR Architecture:

The idea of link-state routing serves as the foundation for OLSR's architecture. In link-state routing, each node keeps track of the network topology and uses that knowledge to choose the most direct route to other nodes. Using a flooding technique, each node broadcasts its link-state information to every other node in the network in order to exchange this information. The optimum route to other nodes is then determined by each node using this information to create a map of the network topology. Because OLSR employs proactive routing, it continuously changes routing data even when no traffic is being transmitted. As a result, network performance is enhanced and communication setup delays are reduced. NIL and the OLSR Layer are the two levels that make up the OLSR protocol. The OLSR Layer is in charge of managing the routing function, whereas the NIL is in charge of controlling the network's physical and data connection layers. Topology Control (TC) messages and Hello messages are the two communication types used by the

OLSR protocol. In order to establish and preserve neighbor relationships between nodes, hello messages are utilized. TC messages are used to update routing tables and convey topology data between nodes. Every node in the network keeps a routing table that details the most direct route to other nodes. Based on the data in TC messages, the routing table is modified.

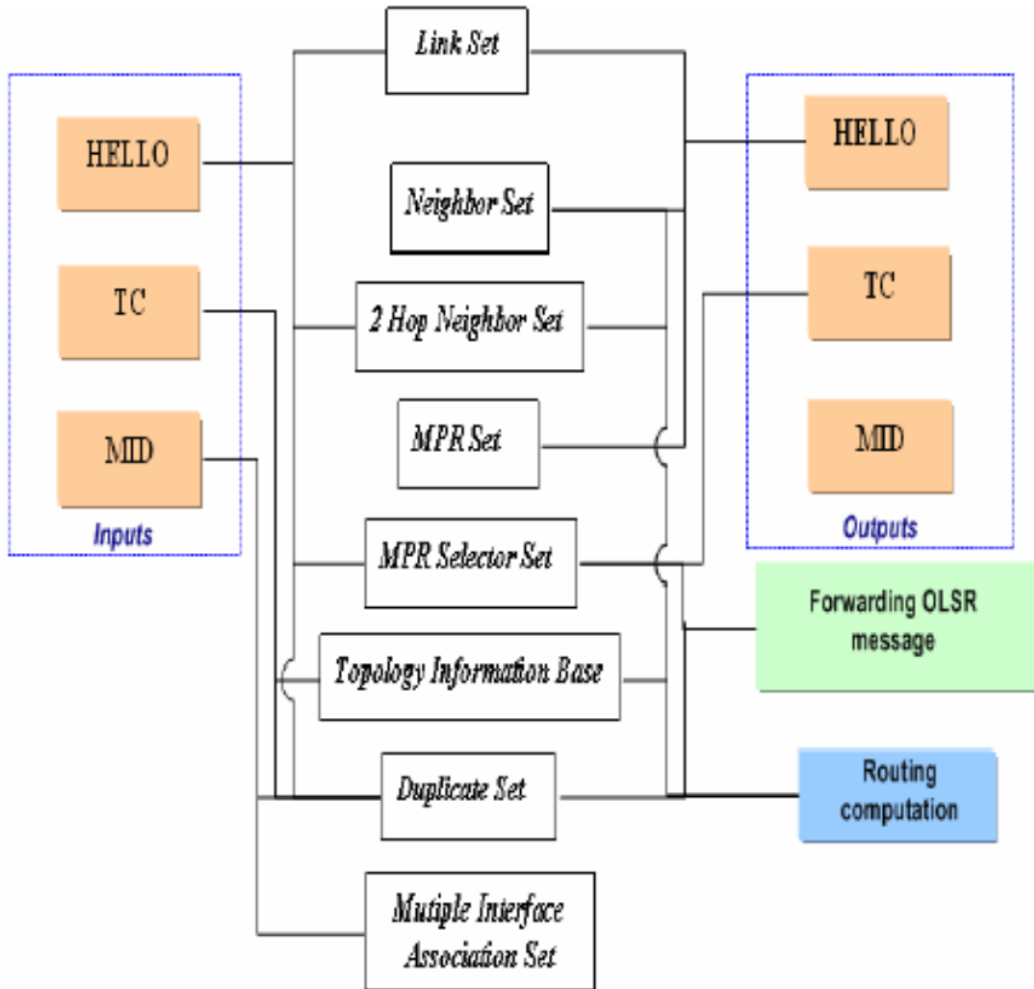


Fig: 3.15 Architecture OLSR Routing Protocol.

OLSR Operations:

MPR, a technique used by OLSR, is used to find and keep track of routes between nodes. Each node in MPR chooses a group of nodes, known as its MPR set, to relay messages on the node's behalf. A node sends a message to one or more nodes in its MPR set when it wants to communicate with another node. The message is then forwarded by these nodes to additional nodes in their MPR sets until it reaches its target node. Multiple interfaces can be used with OLSR, enabling nodes to talk to

one another across various wireless channels. Because of the decreased interference and increased bandwidth, this can aid in network performance improvement.

OLSR Advantages and Disadvantages:

The effectiveness and scalability of OLSR is one of its benefits. OLSR can establish connection between nodes more quickly than other routing protocols because it takes a proactive approach to routing. A large-scale mesh network with numerous nodes is another environment where OLSR performs effectively. The use of OLSR does have some drawbacks, though. One is that it may be susceptible to network splintering and link failures. Each node depends on its neighbors to relay messages, thus if a link breaks or a node is left alone, it may interfere with other nodes' capacity to interact. Furthermore, there are no security safeguards offered by OLSR to guard against malicious attacks or unauthorized access.

Vehicular Ad-hoc Network (VANET) Routing Protocol:

The MANET variant known as VANET is specifically created for vehicle communication. It enables communication between vehicles either directly or via infrastructure elements like roadside units (RSUs). A routing protocol is required to create and maintain routes between vehicles in order to permit communication between them. The VANET Routing Protocol was created with this objective in mind. A hybrid protocol, the VANET Routing Protocol combines proactive and reactive methods. The spatial routing component and the network routing component make up its two halves. Establishing routes connecting vehicles based on their geographic locations is the responsibility of the geographic routing component. It locates each car using the Global Positioning System (GPS) or other location-based technologies. The protocol then creates a virtual map of the network topology using this data. Based on the position of the destination vehicle, this virtual map is utilized to identify the next hop for a packet. The network routing component is in charge of updating the routing database and planning routes between vehicles that are out of wireless communication range with one another. To build and sustain these pathways, it combines proactive and reactive strategies. Maintaining a routing table with details on each vehicle in the network and their connectivity is a proactive strategy. When a vehicle wants to communicate with another vehicle that is outside of its communication range, reactive techniques are employed to construct routes on demand. A number of techniques are also included in the VANET Routing Protocol to guarantee the communication's dependability and security. For instance, it employs message authentication codes (MACs) to guarantee message integrity and guard against tampering. Additionally, it makes

use of digital signatures to protect against spoofing and guarantee the messages' validity. The protocol also features safeguards against flooding and ways to cut back on routing overhead.

The VANET Routing Protocol is a hybrid protocol that establishes and maintains routes between vehicles in a vehicular ad hoc network by combining geographic and network-based methodologies. It has features to reduce flooding risk and optimize routing overhead to provide dependable and secure communication between vehicles.

EXPERIMENT & RESULT ANALYSIS

1) Software and Tools Used:

Operating System: Ubuntu 22.04.2 LTS (highly compatible with ns-3).

Ubuntu is a well-liked Linux-based operating system that has a simple interface and is simple to use. Since its initial release in 2004, it has grown to rank among the most popular Linux distributions worldwide. Because Ubuntu is open-source software, anyone can modify and distribute it and use it for free. Ubuntu's emphasis on simplicity and use is one of its distinguishing characteristics. Both inexperienced and seasoned users can use the user interface because it is made to be simple to use and intuitive. Additionally, the system is made to be simple to install, with a graphical installer that leads users step-by-step through the procedure. Ubuntu's focus on security is another key characteristic. To help shield users from online threats, the operating system includes built-in security features like a firewall and encryption tools. In order to maintain the system current and secure against the most recent threats, Ubuntu also regularly releases security updates and patches. One of Ubuntu's advantages is its enormous collection of software programmers, all of which can be downloaded for free through the Ubuntu Software Centre. This contains a variety of widely used programmers, including media players, productivity programmers, and web browsers. Additionally, a thriving developer community produces and maintains software packages for Ubuntu, guaranteeing that users have access to a broad selection of top-notch software. Overall, Ubuntu is a powerful and adaptable operating system that gives users a number of advantages. Ubuntu is unquestionably a platform worth considering, whether you're a newbie user searching for a user-friendly interface or an experienced user seeking for a secure and reliable one.

Ns3 Tool: Version 3.36.1.

It is common practice to model and simulate different network protocols, technologies, and scenarios using the open-source network simulation programme known as NS-3. In order to enable researchers and developers to test and assess new networking technologies and algorithms, it is created to provide an accurate and realistic picture of network behavior and performance. The modular architecture of NS-3, which is built in C++, enables users to quickly add new features and components to the system. The programme offers a vast library of pre-built models for various network elements, including WLAN nodes, switches, and routers. This makes it possible for users to set up and execute simulations for a variety of scenarios fast. The flexibility of NS-3 to represent both wired and wireless networks is one of its primary characteristics. It offers a selection of wireless models for various technologies, including Wi-Fi, LTE, and 5G, enabling users to replicate actual wireless networks with accurate propagation, interference, and mobility models. Additionally, NS-3 offers a robust and adaptable scripting language that enables users to define intricate network topologies and scenarios. Users can define the behavior of network components, traffic patterns, and network topology using this scripting language. Using Python or C++, users can define their own models and components. The capability of NS-3 to produce thorough output and data for each simulation run is another important feature. This enables users to examine and assess how various network protocols and technologies function in various contexts. The output formats offered by NS-3, including as graphical and textual forms, make it simple to analyze and view the outcomes of simulations. All things considered, NS-3 is a strong and adaptable network simulation tool that offers a variety of features and capabilities for simulating and analyzing network protocols and technologies. It is the perfect option for researchers, developers, and students working in the subject of networking because of its modular architecture, versatile scripting language, and rich output and statistics.

VMware:

A single physical computer may operate several virtual machines (VMs) thanks to VMware, a virtualization technology. For a variety of uses, from testing and development to production deployments, it is a well-liked method for building and administering virtualized environments. VMware's capability to segregate virtualized environments from one another and abstract hardware resources is one of its main advantages. This enables several virtual machines to operate independently on a single physical server, which can improve resource utilization and boost productivity. The fact that VMware can support a variety of operating systems and apps is another key advantage. For developers and IT experts who need to test and run various software configurations in a controlled environment, this makes it the perfect answer. Overall, VMware is a powerful and adaptable virtualization technology that offers users a number of advantages. It is the perfect solution for a variety of use scenarios, from testing and development to production deployments, thanks to its ability to abstract hardware resources, support a wide range of operating systems and applications, and offer advanced management capabilities.

Wireshark:

It is a free and open-source packet analyzer called Wireshark is employed in software and communication protocol creation, analysis, network troubleshooting, and education. It is compatible with Linux, Windows, macOS, and other Unix-like operating systems. Users of Wireshark can record and examine network traffic in real-time, and it offers comprehensive details about the packets that are being sent over the network.

2) Commands Used:

1. `$ sudo apt update`

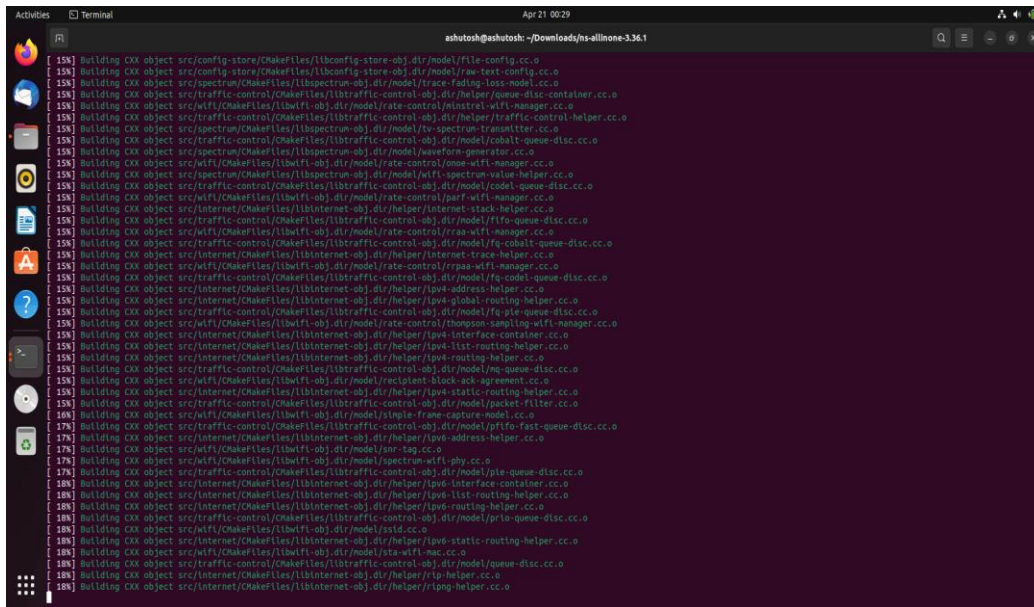


Fig: 4.2 Creating build of the NS3 packages.

6. \$./ns3 run hello-simulator

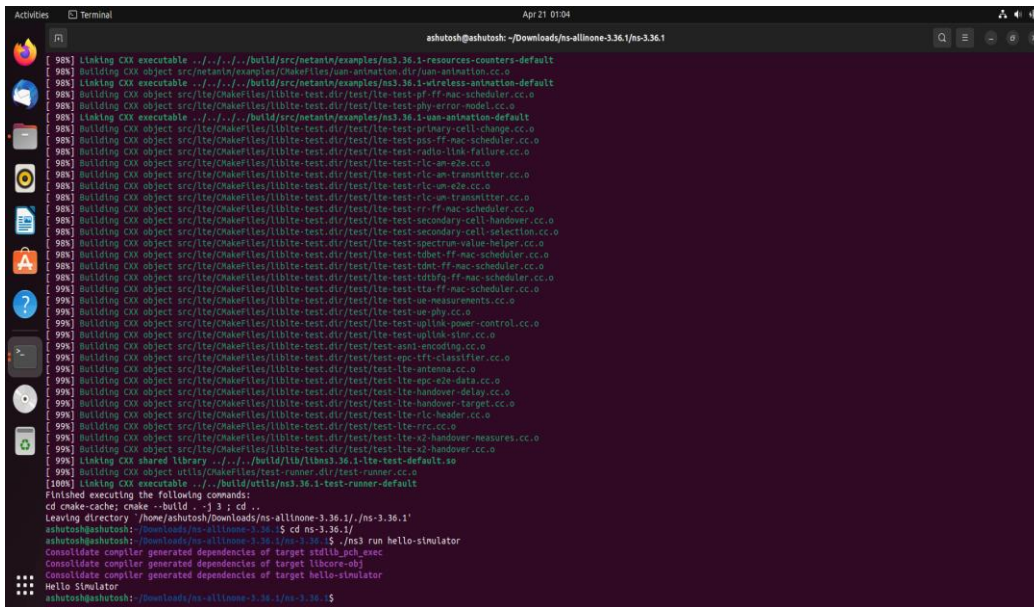


Fig: 4.3 Running the hello-simulator file using Ns3 command.

7. \$./ns3 run scratch/first.py

```

Activities Terminal Apr 23 22:50
ashutosh@ashutosh: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1
-- Build files have been written to: /home/ashutosh/Desktop/ns-allinone-3.36.1/ns-3.36.1/cmake-cache
Consolidate compiler generated dependencies of target libnscore-obj
Consolidate compiler generated dependencies of target libstats-obj
Consolidate compiler generated dependencies of target libbridge-obj
Consolidate compiler generated dependencies of target libscore-obj
Consolidate compiler generated dependencies of target libconfig-store-obj
Consolidate compiler generated dependencies of target libnetwork-obj
Consolidate compiler generated dependencies of target libtraffic-control-obj
Consolidate compiler generated dependencies of target libmility-obj
Consolidate compiler generated dependencies of target libpropagation-obj
Consolidate compiler generated dependencies of target libntenna-obj
Consolidate compiler generated dependencies of target libinternet-obj
Consolidate compiler generated dependencies of target libenergy-obj
Consolidate compiler generated dependencies of target libspectrum-obj
Consolidate compiler generated dependencies of target libwifi-obj
Consolidate compiler generated dependencies of target libadv-obj
Consolidate compiler generated dependencies of target libapplications-obj
Consolidate compiler generated dependencies of target libbuildings-obj
Consolidate compiler generated dependencies of target libcsma-obj
Consolidate compiler generated dependencies of target libcma-layout-obj
Consolidate compiler generated dependencies of target libpoint-to-point-obj
Consolidate compiler generated dependencies of target libsdv-obj
Consolidate compiler generated dependencies of target raw-pod-creator
Consolidate compiler generated dependencies of target tap-device-creator
Consolidate compiler generated dependencies of target libdar-obj
Consolidate compiler generated dependencies of target libflow-monitor-obj
Consolidate compiler generated dependencies of target libfwd-net-device-obj
Consolidate compiler generated dependencies of target libinternet-apps-obj
Consolidate compiler generated dependencies of target libl-wan-obj
Consolidate compiler generated dependencies of target libvirtual-net-device-obj
Consolidate compiler generated dependencies of target libstantn-obj
Consolidate compiler generated dependencies of target libpoint-to-point-layout-obj
Consolidate compiler generated dependencies of target libmesh-obj
Consolidate compiler generated dependencies of target libwave-obj
Consolidate compiler generated dependencies of target liblink-vector-routing-obj
Consolidate compiler generated dependencies of target liblr-obj
Consolidate compiler generated dependencies of target libolr-obj
Consolidate compiler generated dependencies of target libislopan-obj
Consolidate compiler generated dependencies of target libtap-bridge-obj
Consolidate compiler generated dependencies of target libtopology-read-obj
[ * ] Building CXX object scratch/CMakeFiles/scratch.dir/first.cc.o
[ * ] Linking CXX executable ./build/scratch/ns_36.1-first-default
At time +2.0 client sent 1024 bytes to 10.1.1.2 port 9
At time +2.00369s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.00369s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.00737s client received 1024 bytes from 10.1.1.2 port 9
ashutosh@ashutosh:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$

```

Fig: 4.4 Running the first.py in build script.

3. Network Protocol Simulation:

3.1 AODV - Ad Hoc on Demand Distance Vector:

Output:

```

ashutosh@ashutosh: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1
ashutosh@ashutosh:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ cd ..
ashutosh@ashutosh:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ./ns3 run scratch/aodv
Consolidate compiler generated dependencies of target scratch-aodv
Creating 10 nodes 50 m apart.
Starting simulation for 100 s ...
PING 10.0.0.10 - 56 bytes of data - 64 bytes including ICMP and IPv4 headers.
64 bytes from 10.0.0.10: icmp_seq=0 ttl=64 time=1075.17ms
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=76.890ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=64 time=7.4792ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=64 time=8.1678ms
64 bytes from 10.0.0.10: icmp_seq=4 ttl=64 time=7.5431ms
64 bytes from 10.0.0.10: icmp_seq=5 ttl=64 time=7.4682ms
64 bytes from 10.0.0.10: icmp_seq=6 ttl=64 time=7.4638ms
64 bytes from 10.0.0.10: icmp_seq=7 ttl=64 time=7.4602ms
64 bytes from 10.0.0.10: icmp_seq=8 ttl=64 time=7.5557ms
64 bytes from 10.0.0.10: icmp_seq=9 ttl=64 time=7.5689ms
64 bytes from 10.0.0.10: icmp_seq=10 ttl=64 time=7.4779ms
64 bytes from 10.0.0.10: icmp_seq=11 ttl=64 time=7.4823ms
64 bytes from 10.0.0.10: icmp_seq=12 ttl=64 time=7.5626ms
64 bytes from 10.0.0.10: icmp_seq=13 ttl=64 time=7.4779ms
64 bytes from 10.0.0.10: icmp_seq=14 ttl=64 time=8.2267ms
64 bytes from 10.0.0.10: icmp_seq=15 ttl=64 time=7.7523ms
64 bytes from 10.0.0.10: icmp_seq=16 ttl=64 time=7.5397ms
64 bytes from 10.0.0.10: icmp_seq=17 ttl=64 time=7.5389ms
64 bytes from 10.0.0.10: icmp_seq=18 ttl=64 time=7.4894ms
64 bytes from 10.0.0.10: icmp_seq=19 ttl=64 time=7.5576ms
64 bytes from 10.0.0.10: icmp_seq=20 ttl=64 time=7.5271ms
64 bytes from 10.0.0.10: icmp_seq=21 ttl=64 time=7.5447ms
64 bytes from 10.0.0.10: icmp_seq=22 ttl=64 time=7.5382ms
64 bytes from 10.0.0.10: icmp_seq=23 ttl=64 time=8.8915ms
64 bytes from 10.0.0.10: icmp_seq=24 ttl=64 time=7.4936ms
64 bytes from 10.0.0.10: icmp_seq=25 ttl=64 time=7.4723ms
64 bytes from 10.0.0.10: icmp_seq=26 ttl=64 time=7.5845ms
64 bytes from 10.0.0.10: icmp_seq=27 ttl=64 time=7.4662ms
64 bytes from 10.0.0.10: icmp_seq=28 ttl=64 time=7.464ms
64 bytes from 10.0.0.10: icmp_seq=29 ttl=64 time=7.4875ms
64 bytes from 10.0.0.10: icmp_seq=30 ttl=64 time=7.4573ms
64 bytes from 10.0.0.10: icmp_seq=31 ttl=64 time=7.4806ms
64 bytes from 10.0.0.10: icmp_seq=32 ttl=64 time=7.5498ms
64 bytes from 10.0.0.10: icmp_seq=33 ttl=64 time=8.1109ms
... 10.0.0.10 ping statistics ...
100 packets transmitted, 34 received, 66% packet loss, time +1e+05ms
rtt min/avg/max/mdev = 7.181/10283.993/6 ms
ashutosh@ashutosh:~/Desktop/ns-allinone-3.36.1/ns-3.36.1$ ./ns3 run scratch/aodv
Consolidate compiler generated dependencies of target scratch-aodv
Creating 20 nodes 50 m apart.
Starting simulation for 100 s ...

```

Fig: 4.5 Running the script for Aodv Routing Protocol.

```

ashutosh@ashutosh: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1
100 packets transmitted, 34 received, 66% packet loss, time 1e+05ms
rtt min/avg/max/mdev = 7/101.3/2073/393.6 ms
ashutosh@ashutosh: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1 $ ./ns3 run scratch/aodv
[ * ] Building CXX object scratch/Makefile/scratch_adv.dir/adv.cc.o
[ * ] Linking CXX executable .././build/scratch/ns3.16-aodv-default
Creating 20 nodes 50 m apart.
Starting simulation for 100 s ...
PING 10.0.0.20 - 56 bytes of data - 84 bytes including ICMP and IPv4 headers.
64 bytes from 10.0.0.20: icmp_seq=0 ttl=46 time=2250.1ms
64 bytes from 10.0.0.20: icmp_seq=1 ttl=46 time=1259.56ms
64 bytes from 10.0.0.20: icmp_seq=2 ttl=46 time=261.00ms
64 bytes from 10.0.0.20: icmp_seq=3 ttl=46 time=16.2721ms
64 bytes from 10.0.0.20: icmp_seq=4 ttl=46 time=16.8218ms
64 bytes from 10.0.0.20: icmp_seq=5 ttl=46 time=15.7929ms
64 bytes from 10.0.0.20: icmp_seq=6 ttl=46 time=15.9545ms
64 bytes from 10.0.0.20: icmp_seq=7 ttl=46 time=15.8989ms
64 bytes from 10.0.0.20: icmp_seq=8 ttl=46 time=15.8883ms
64 bytes from 10.0.0.20: icmp_seq=9 ttl=46 time=15.9076ms
64 bytes from 10.0.0.20: icmp_seq=10 ttl=46 time=16.131ms
64 bytes from 10.0.0.20: icmp_seq=11 ttl=46 time=16.2764ms
64 bytes from 10.0.0.20: icmp_seq=12 ttl=46 time=15.8218ms
64 bytes from 10.0.0.20: icmp_seq=13 ttl=46 time=15.941ms
64 bytes from 10.0.0.20: icmp_seq=14 ttl=46 time=16.4575ms
64 bytes from 10.0.0.20: icmp_seq=15 ttl=46 time=16.3922ms
64 bytes from 10.0.0.20: icmp_seq=16 ttl=46 time=16.6713ms
64 bytes from 10.0.0.20: icmp_seq=17 ttl=46 time=16.2286ms
64 bytes from 10.0.0.20: icmp_seq=18 ttl=46 time=15.94ms
64 bytes from 10.0.0.20: icmp_seq=19 ttl=46 time=16.2834ms
64 bytes from 10.0.0.20: icmp_seq=20 ttl=46 time=16.887ms
64 bytes from 10.0.0.20: icmp_seq=21 ttl=46 time=15.8531ms
64 bytes from 10.0.0.20: icmp_seq=22 ttl=46 time=16.4888ms
64 bytes from 10.0.0.20: icmp_seq=23 ttl=46 time=15.8463ms
64 bytes from 10.0.0.20: icmp_seq=24 ttl=46 time=16.1566ms
64 bytes from 10.0.0.20: icmp_seq=25 ttl=46 time=15.9653ms
64 bytes from 10.0.0.20: icmp_seq=26 ttl=46 time=16.3486ms
64 bytes from 10.0.0.20: icmp_seq=27 ttl=46 time=16.266ms
64 bytes from 10.0.0.20: icmp_seq=28 ttl=46 time=15.9344ms
64 bytes from 10.0.0.20: icmp_seq=29 ttl=46 time=15.8968ms
64 bytes from 10.0.0.20: icmp_seq=30 ttl=46 time=16.3665ms
64 bytes from 10.0.0.20: icmp_seq=31 ttl=46 time=16.1928ms
64 bytes from 10.0.0.20: icmp_seq=32 ttl=46 time=15.8942ms
64 bytes from 10.0.0.20: icmp_seq=33 ttl=46 time=15.8386ms
... 10.0.0.20 ping statistics ...
100 packets transmitted, 34 received, 66% packet loss, time 1e+05ms
rtt min/avg/max/mdev = 15/125.3/2258/434.3 ms
ashutosh@ashutosh: ~/Desktop/ns-allinone-3.36.1/ns-3.36.1 $ ./ns3 run scratch/aodv
Consolidate compiler generated dependencies of target scratch_adv
[ * ] Building CXX object scratch/Makefile/scratch_adv.dir/adv.cc.o

```

Fig: 4.6 Receiving data packets in Aodv Routing Protocol.

3.1.2 Aodv Routing Table:

```

Open | aodvroutes
~/Desktop/ns-allinone-3.36.1
Save | Plain Text | Tab width: 8 | Ln: 1, Col: 1 | BNS

1 Node: 0; Time: +8s, Local time: +8s, AODV Routing table
2
3 AODV Routing table
4 Destination Gateway Interface Flag Expire Hops
5 10.0.0.20 102.102.102.102 102.102.102.102 IN_SEARCH +2.3s 35
6 10.255.255.255 10.255.255.255 10.0.0.1 UP +9.2e+09s 1
7 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
8
9
10 Node: 1; Time: +8s, Local time: +8s, AODV Routing table
11
12 AODV Routing table
13 Destination Gateway Interface Flag Expire Hops
14 10.255.255.255 10.255.255.255 10.0.0.2 UP +9.2e+09s 1
15 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
16
17
18 Node: 2; Time: +8s, Local time: +8s, AODV Routing table
19
20 AODV Routing table
21 Destination Gateway Interface Flag Expire Hops
22 10.255.255.255 10.255.255.255 10.0.0.3 UP +9.2e+09s 1
23 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
24
25
26 Node: 3; Time: +8s, Local time: +8s, AODV Routing table
27
28 AODV Routing table
29 Destination Gateway Interface Flag Expire Hops
30 10.255.255.255 10.255.255.255 10.0.0.4 UP +9.2e+09s 1
31 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
32
33
34 Node: 4; Time: +8s, Local time: +8s, AODV Routing table
35
36 AODV Routing table
37 Destination Gateway Interface Flag Expire Hops
38 10.255.255.255 10.255.255.255 10.0.0.5 UP +9.2e+09s 1
39 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
40
41
42 Node: 5; Time: +8s, Local time: +8s, AODV Routing table
43
44 AODV Routing table
45 Destination Gateway Interface Flag Expire Hops
46 10.255.255.255 10.255.255.255 10.0.0.6 UP +9.2e+09s 1
47 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1

```

Fig: 4.7 Routing table for Aodv Routing Protocol.

```

112
113
114 Node: 14; Time: +8s, Local time: +8s, AODV Routing table
115
116 AODV Routing table
117 Destination Gateway Interface Flag Expire Hops
118 10.255.255.255 10.255.255.255 10.0.0.15 UP +9.2e+09s 1
119 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
120
121
122 Node: 15; Time: +8s, Local time: +8s, AODV Routing table
123
124 AODV Routing table
125 Destination Gateway Interface Flag Expire Hops
126 10.255.255.255 10.255.255.255 10.0.0.16 UP +9.2e+09s 1
127 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
128
129
130 Node: 16; Time: +8s, Local time: +8s, AODV Routing table
131
132 AODV Routing table
133 Destination Gateway Interface Flag Expire Hops
134 10.255.255.255 10.255.255.255 10.0.0.17 UP +9.2e+09s 1
135 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
136
137
138 Node: 17; Time: +8s, Local time: +8s, AODV Routing table
139
140 AODV Routing table
141 Destination Gateway Interface Flag Expire Hops
142 10.255.255.255 10.255.255.255 10.0.0.18 UP +9.2e+09s 1
143 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
144
145
146 Node: 18; Time: +8s, Local time: +8s, AODV Routing table
147
148 AODV Routing table
149 Destination Gateway Interface Flag Expire Hops
150 10.255.255.255 10.255.255.255 10.0.0.19 UP +9.2e+09s 1
151 127.0.0.1 127.0.0.1 127.0.0.1 UP +9.2e+09s 1
152
153
154 Node: 19; Time: +8s, Local time: +8s, AODV Routing table
155
156 AODV Routing table
157 Destination Gateway Interface Flag Expire Hops
158 10.255.255.255 10.255.255.255 10.0.0.20 UP +9.2e+09s 1

```

Fig: 4.8 Routing table represents Destination, Gateway and Interface of each node.

```

aodv-node-11-0.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-F>
No. Time Source Destination Protocol Length Info
1.0.000000 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
2.1.003988 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
3.1.006000 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
4.3.000000 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
5.4.005803 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
6.5.006005 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
7.6.004988 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
8.7.003002 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
9.7.998999 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
10.9.000999 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
11.10.000002 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
12.10.999005 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
13.12.000000 10.0.0.12 10.255.255.255 AODV 86 Route Reply, D: 10.0.0.12, O: 10.0.0.12, Mont=0, DSN=0, Lifetime=2000
Frame 11: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 0
Logical-Link Control
Internet Protocol Version 4, Src: 10.0.0.12, Dst: 10.255.255.255
User Datagram Protocol, Src Port: 654, Dst Port: 654
Ad hoc On-demand Distance Vector Routing Protocol, Route Reply, Dest IP: 10.0.0.12, Orig IP: 10.0.0.12, Lifetime=2000
0000 08 00 00 00 ff ff ff ff ff 00 00 00 00 0c .....
0010 00 00 00 00 00 00 00 20 00 0a 03 00 00 .....
0020 00 00 45 00 30 00 00 00 00 01 11 00 00 0a ..... E 0
0030 00 0c 0a ff ff 02 be 02 be 00 1c 00 00 02 00 .....
0040 00 00 00 00 00 00 00 00 0a 00 00 00 00 00 .....
0050 07 00 00 00 00 .....
Packets: 100 - Displayed: 100 (100.0%) Profile: Default

```

Fig: 4.9 Analyzing packets in Aodv Routing Protocol using Wireshark.

3.2 DSDV - Destination Sequenced Distance Vector:

Results:

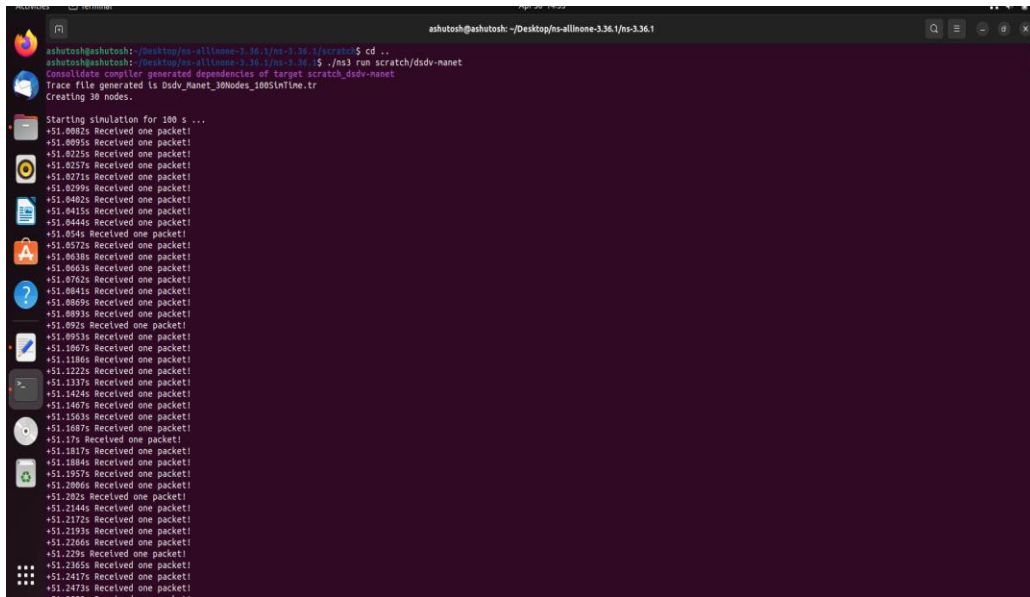


Fig: 4.10 Running the script for DSDV Routing Protocol.

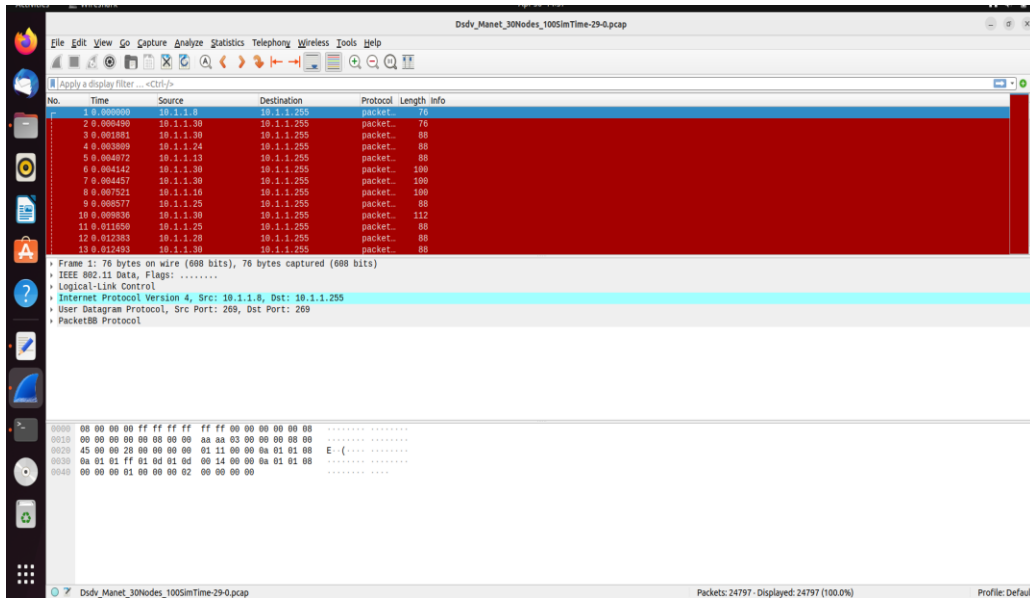


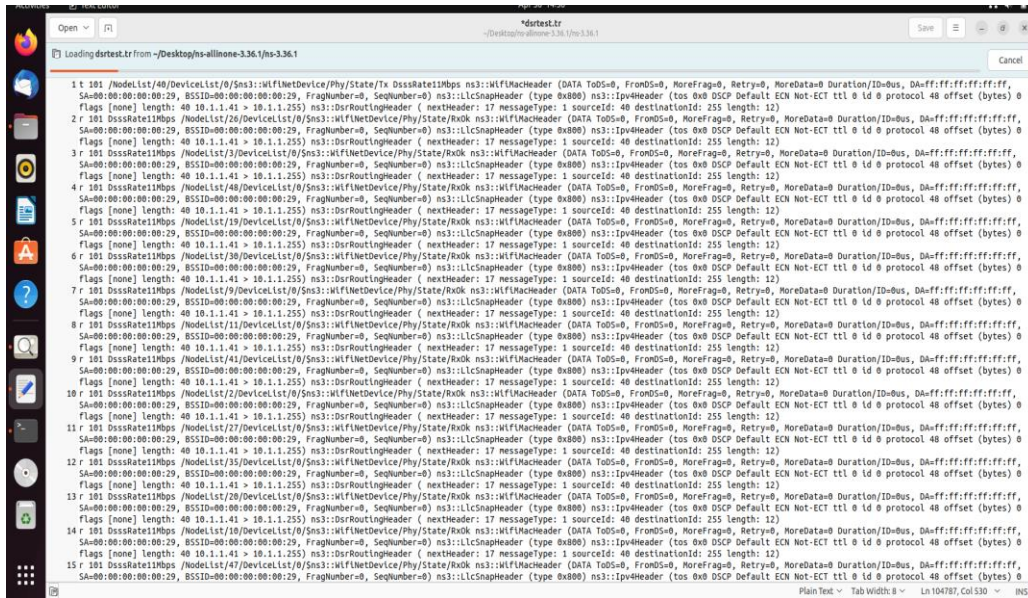
Fig: 4.11 Analyzing packets in Dsdv Routing Protocol using Wireshark.

3.3 DSR - Dynamic Source Routing:

Result:

```
Compiledate compiler generated dependencies of target ns3-3.36
msg="TraceSource 'XERHeader' is obsolete, with no fallback. Depending on the failure type, use the NackedPdu trace, the DroppedPdu trace or one of the traces associated with TX timeouts.", file/home
/ashutosh/Desktop/ns-allInOne-3.36.1/ns-3.36.1/src/core/model/type-id.cc, line=1164
terminate called without an active exception
Command 'outifscrutchns.36.1-dsr-default' died with <signals.SIGABRT: 6>
ashutosh@ashutosh: ~$
```

Fig. 4.12 Creating build for the Dsr code script.



```
1 t 101 /ModelList/40/DeviceList/0/Sns3:WiFiNetDevice/PhysState/Tx DssRate1Mbps ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
2 r 101 DssRate1Mbps /ModelList/20/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
3 r 101 DssRate1Mbps /ModelList/27/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
4 r 101 DssRate1Mbps /ModelList/48/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
5 r 101 DssRate1Mbps /ModelList/19/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
6 r 101 DssRate1Mbps /ModelList/30/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
7 r 101 DssRate1Mbps /ModelList/9/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
8 r 101 DssRate1Mbps /ModelList/11/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
9 r 101 DssRate1Mbps /ModelList/4/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
10 r 101 DssRate1Mbps /ModelList/2/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
11 r 101 DssRate1Mbps /ModelList/27/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
12 r 101 DssRate1Mbps /ModelList/35/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
13 r 101 DssRate1Mbps /ModelList/20/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
14 r 101 DssRate1Mbps /ModelList/10/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
Flags [none] length: 40 10.1.1.41 > 10.1.1.255) ns3:DSRRoutingHeader ( nextHeader: 17 messageType: 1 sourceId: 40 destinationId: 255 length: 12)
15 r 101 DssRate1Mbps /ModelList/47/DeviceList/0/Sns3:WiFiNetDevice/PhysState/RxOk ns3:WiFiMacHeader (DATA ToDS=0, FromDS=0, MoreFrag=0, Retry=0, MoreData=0 Duration/ID=bus, Da=ffff:ffff:ffff:ffff, SA=00:00:00:00:00:29, BSSID=00:00:00:00:00:29, FragNumber=0, SeqNumber=0) ns3:LLCSnapHeader (type 0x800) ns3:Ipv4Header (tos 0x0 DSCP Default ECN Not-ECT ttl 0 id 0 protocol 48 offset (bytes) 0
```

Fig. 4.13 Creating Message type, Source ID for each node in Dsr Routing Protocol.

CONCLUSION

Internet of Vehicles (IoV) is a new field relatively but it has the potential to change how we interact with our cars and the transportation system. In order for vehicles to connect with one another and the surrounding infrastructure, network routing is an essential part of the Internet of Vehicles (IoV). The future of network routing in the IoV involve a variety of cutting-edge technologies and design concepts, including edge computing, machine learning, 5G networks, blockchain technology, cooperative intelligent transport systems (C-ITS), multi-hop communication, dynamic routing, AI-assisted routing, hybrid networking, and QoS-aware routing. The total performance of the transportation system improved by these technologies and design concepts, which make it possible to provide more dependable, safe, and efficient transportation. Network routing algorithms become more and more crucial as the IoV field develops if we are to create a connected, effective, and sustainable transportation system. There are many advantages to network routing in IoV. Network routing, for instance, it increases the effectiveness of the transport system overall and aid to decrease traffic congestion and improve road safety. The development of cooperative driving techniques, like platooning, where many vehicles move closely together to lessen wind resistance and fuel consumption, facilitated by network routing by allowing cars to interact with one another and with the surrounding infrastructure. Network routing also makes it possible to monitor and predict traffic in real-time, which utilized to improve route design and shorten trip times. Additionally, network routing make it possible for vehicles nearby to receive warnings about dangerous road conditions or impending obstacles, among other safety-critical information. Network routing, in general, is a crucial element of IoV that has the ability to completely change the way we think about transportation. Network routing become increasingly more crucial as the IoV

field develops if we are to create a connected, effective, and sustainable transportation system.

Along with the aforementioned advantages, network routing in IoV has the potential to open up new business models and revenue streams. The creation of new services and applications, such as personalized navigation and predictive maintenance, it made possible by network routing, for instance, by enabling the collecting and analysis of enormous volumes of data. Subscription-based business models or the sale of data access to outside businesses are two ways these services made profitable. The creation of new mobility services, like ride- and car-sharing, which can be accessible through a mobile app, also be made possible by network routing. This increases consumer convenience and flexibility while lowering the number of automobiles on the road and the resulting environmental effect. Overall, IoV network routing has the potential to generate substantial economic value and promote innovation in the transportation sector. As a result, in the years to come it is going to be a major factor in growth and competitiveness.

Future work

The Internet of Vehicles (IoV) ecosystem require a variety of cutting-edge technologies and architectural concepts for network routing in the future. These include edge computing, AI-assisted routing, 5G networks, blockchain technology, cooperative intelligent transport systems (C-ITS), multi-hop communication, dynamic routing, AI-assisted routing, hybrid networking, and QoS-aware routing. We will be able to travel in a way that is more reliable, safe, and efficient thanks to these technologies and design concepts, which ultimately change how we interact with our cars and the transportation system. These new technologies included into network routing algorithms in IoV, allowing for real-time adaptation to changing conditions, prioritization of vital data, more effective data transfer, and an improvement in the system's safety and effectiveness. The development of routing algorithms and protocols that can cope with the difficulties of dynamic network

topology, high vehicle mobility, and many types of communication is a future task in network routing in the Internet of Vehicles (IoV) environment. In order to increase routing performance, cutting-edge technologies like SDN and NFV are being researched, and to fend off intrusions, efficient and secure routing methods are being created. Artificial intelligence and machine learning approaches are being investigated to improve the effectiveness and efficiency of routing, and effective and scalable systems are being created to handle the massive volumes of data provided by connected automobiles and other IoT devices.

1) Edge Computing:

In context of the Internet of Vehicles, edge computing entails processing vehicle-generated data closer to the vehicles themselves, at the network edge, as opposed to transmitting it to a central point for processing. The responsiveness of network routing algorithms can be enhanced and latency reduced as a result. Edge computing, for instance, enables routing decisions to be made in real-time based on the location and state of the vehicles.

2) Machine Learning:

Large amounts of car-generated data, including location data, traffic patterns, and vehicle performance data, can be analyzed using machine learning algorithms. Machine learning algorithms can create better routing decisions by finding patterns and trends in this data. In IoV situations where conditions can change quickly, machine learning algorithms are highly suited for use since they can adapt to changing conditions in real-time.

3) 5G Networks:

Compared to earlier generations of wireless networks, 5G networks offer higher data transmission rates and lower latency. Because they can quickly transmit and process large amounts of data, they are perfect for use in IoV environments. 5G

networks make it possible for routing algorithms to receive and digest data quickly, enabling more sophisticated routing decisions.

4) Blockchain Technology:

The security and privacy of data produced by automobiles can be helped by blockchain technology. Blockchain technology can reduce the likelihood of data theft or tampering by employing a decentralized ledger to store data. Greater transparency and accountability in the use of data produced by vehicles can be made possible by blockchain technology.

5) Cooperative Intelligent Transport Systems (C-ITS):

Vehicles can speak with one other and with infrastructure like traffic signals and road signs thanks to a technology called C-ITS. Vehicles can make better routing decisions based on real-time data by integrating C-ITS technology into network routing algorithms. For instance, C-ITS can warn a car and reroute it to avoid an accident if a car ahead of it is involved in one.

6) Dynamic Routing:

Dynamic routing algorithms can instantly adjust to shifting circumstances, which makes them ideal for usage in IoV contexts. Dynamic routing algorithms, for instance, can redirect cars to avoid accidents or traffic jams, making transportation safer and more effective.

REFERENCES

- [1] J. Contreras-Castillo, S. Zeadally and J. A. Guerrero-Ibañez, "Internet of Vehicles: Architecture, Protocols, and Security," in *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3701-3709, Oct. 2018, doi: 10.1109/JIOT.2017.2690902.
- [2] C. Perkins, E. Belding-Royer, and S. Das. "Ad hoc On-Demand Distance Vector (AODV) Routing." RFC 3561, July 2003.
- [3] Li, L. (2015). DSDV Routing Protocol Implementation Design. In 2015 IEEE 2nd International Conference on Computing Communication Control and Automation (ICCCUBEA) (pp. 1-6). IEEE.
- [4] "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks" by David B. Johnson and David A. Maltz, published in 1996.
- [5] Hameed, F., & Iqbal, F. (2015). A comparative study of routing protocols in mobile ad-hoc networks with focus on OLSR and TORA. *Procedia Computer Science*, 52, 776-783.
- [6] Saurabh Gupta, Sangeeta Rani, Dr. Amit Dixit. "Features Exploration of Distinct Load Balancing Algorithms in Cloud Computing Environment", *International Journal of Advanced Networking and Applications*, Volume: 11 Issue: 01 Pages: 4177-4183(2019) ISSN: 0975-0290.
- [7] A. S. Al-Fuqaha, M. Guizani, M. Mohammadi, MAldhari, and M. Ayyash. "A Survey of Routing Protocols in the Internet of Vehicles." *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2377-2395, 2015.
- [8] H. Menouar, A. Mellouk, and L. Moussaoui. "Routing Protocols for Vehicular Ad Hoc Networks: A Survey." *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 559-580, 2016.

- [9] H. Qi, X. Zhang, Y. Yang, and Y. Sun. "Routing in the Internet of Vehicles: A Comprehensive Survey." *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 672-695, 2018.
- [10] M. R. Akbari, S. S. Bhatti, M. Q. Akbar, and J. Zhang. "A Comprehensive Survey of Routing Protocols in Internet of Vehicles (IoV): Challenges and Solutions." *IEEE Access*, vol. 8, pp. 183207-183241, 2020.
- [11] R. S. Dahiya and D. Singh. "A Survey of Routing Techniques in the Internet of Vehicles." *Wireless Personal Communications*, vol. 97, no. 2, pp. 2633-2663, 2017.
- [12] D. D. Dung and T. N. Nguyen. "A Survey of Routing Protocols in Vehicular Ad Hoc Networks." *Journal of Computer Science and Cybernetics*, vol. 34, no. 2, pp. 125-142, 2018.
- [13] H. Khan, S. A. Hassan, and S. M. Riaz. "A Comprehensive Survey of Vehicular Ad Hoc Network: Communication, Security, and Routing." *Wireless Personal Communications*, vol. 111, no. 3, pp. 1795-1827, 2020.
- [14] Y. Cui, Y. Xiao, and J. Liu. "A Survey of Routing in the Internet of Vehicles: Technical Challenges and Future Directions." *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 385-402, 2020.
- [15] S. M. Islam and M. S. Hossain. "A Comprehensive Survey of Routing Protocols in VANETs: Challenges and Future Directions." *Journal of Network and Computer Applications*, vol. 132, pp. 50-76, 2019.
- [16] P. Basu and M. Maity. "A Survey of Routing Protocols for Intelligent Transportation Systems." *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2042-2071, 2018.

[17] A. Dua, N. Kumar, and S. Bawa. "A systematic review on routing protocols for vehicular ad hoc networks." *Vehicular Communications*, vol. 1, no. 1, pp. 33–52, 2014.