Secure Distribution of Certificate Revocation List (CRL) in Vehicular Ad-hoc Network (VANET)

Submitted By

Md. Imamul Islam
ID: 2014-1-60-027

Shuvo Barua
ID: 2014-1-60-051

Md. Ariful Islam
ID: 2014-1-60-078

Supervised By

Rashedul Amin Tuhin
Senior Lecturer
Dept. of Computer Science and Engineering
East West University

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Department of Computer Science and Engineering
East West University
Dhaka-1212, Bangladesh

April 2018

Declaration

We, hereby, declare that the work presented in this thesis is the outcome of the investigation performed by me under the supervision of Rashedul Amin Tuhin, Senior Lecturer, Department of Computer Science and Engineering, East West University. We also declare that no part of this thesis has been or is being submitted elsewhere for the award of any degree or diploma. This thesis complies with the regulations of this University and meets the accepted standards with the respect to originality and equality. We hereby release this thesis to the public. We also authorize the university or other individuals to make copies of this thesis as needed for scholarly research.

Countersigned

_____

(Rashedul Amin Tuhin)
Supervisor

Signature

_____

(Md. Imamul Islam, ID: 2014-1-60-027)

Signature

_____

(Shuvo Barua, ID: 2014-1-60-051)

Signature

_____

(Md. Ariful Islam, ID: 2014-1-60-078)

Letter of Acceptance

The thesis entitled “Secure Distribution of Certificate Revocation List (CRL) in Vehicular Ad-hoc Network (VANET)” submitted by Md. Imamul Islam, ID 2014-1-60-027, Shuvo Barua, ID 2014-1-60-051 & Md. Ariful Islam ID 2014-1-60-078 to the Department of Computer Science & Engineering, East West University, Dhaka 1212, Bangladesh is accepted as satisfactory for partial fulfillment for the degree of Bachelor of Science in Computer Science & Engineering in April 2018.

Board of Examiners

_____

Rashedul Amin Tuhin
Senior Lecturer — Supervisor
Department of Computer Science and Engineering
East West University, Dhaka, Bangladesh

_____

Dr. Ahmed Wasif Reza
Associate Professor and Chairperson — Chairperson
Department of Computer Science and Engineering
East West University, Dhaka, Bangladesh

Abstract

In this study, we have implemented Public Key Infrastructure (PKI) with Certification Authority (CA) that will issue and verify the digital certificate of any entity or vehicle in a Vehicular Ad-hoc network (VANET). After revoking a certificate for whatever reasons, the CA generates a Certificate Revocation List (CRL), where there will be the list of revoked certificates. Distribution of CRL is crucial for the quick removal of faulty or misbehaving nodes or any entity whose key pair has been compromised. Secure distribution of CRLs can be ensured with the help of the PKI, to the VANET. We have evaluated the performance of three VANET protocols. These are Ad hoc On-Demand Distance Vector (AODV), Dynamic Source Routing Protocol (DSR) and Destination Sequence Distance Routing Protocol (DSDV). We have evaluated their performance regarding average Lost Packet Ratio (LPR) and average End to End Delay to investigate which protocol performs efficiently in propagating the CRL message to the all nodes. Our results help that there is no best protocol that performs well in every situation, rather depending on the situation, the routing protocol can be selected for the quickest distribution of CRLs.

Acknowledgement

As it is true for everyone, we have also arrived at this point of achieving a goal in our life through various interactions with and help from other people. However, written words are often elusive and harbor diverse interpretations even in one's mother language. Therefore, we would not like to make efforts to find best words to express my thankfulness other than simply listing those people who have contributed to this thesis itself in an essential way. This work was carried out in the Department of Computer Science and Engineering at East West University, Bangladesh. First of all, we would like to express my deepest gratitude to the almighty for His blessings on us. Next, our special thanks go to our supervisor, "Rashedul Amin Tuhin", who gave us this opportunity, initiated us into the field of "Secure Distribution of Certificate Revocation List (CRL) in Vehicular Ad-hoc Network (VANET)" and without whom this work would not have been possible. His encouragements, visionaries and thoughtful comments and suggestions, unforgettable support at every stage of our B.Sc study was simply appreciating and essential. His ability to muddle us enough to finally answer our own question correctly is something valuable what we have learned and we would try to emulate if ever we get the opportunity.

We would like to thank "Bijan Pal" sir for his helpful suggestions in solving tricky technical problems. Last but not the least, we would like to thank our parents for their unending support, encouragement, and prayers.

Md. Imamul Islam,
April 2018

Shuvo Barua,
April 2018

Md. Ariful Islam,
April 2018

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

An ad hoc network [1] is a network that is composed of individual devices communicating with each other directly. It is a decentralized type of wireless network [1] [2]. The term Ad-hoc entitles spontaneous construction because these networks are often bypassed by the gatekeeping hardware or central access point. There are many Ad-hoc networks which are local area networks, where devices are enabled to send data directly to one another rather than going through a centralized access point.

The idea of an Ad-hoc network is not often familiar to end users who have only seen small residential or business networks that use a router to send wireless signals to individual computers. However, the Ad-hoc network is used in new types of wireless engineering. For example, in a mobile ad hoc network, it involves mobile devices to communicate directly with each other. Another type of ad hoc network is the vehicular ad hoc network. It involves placing communication devices in cars. Both of these are examples of ad hoc networks that use huge collection of individual to freely communicate without a top-down or hierarchical communication structure [16].

Experts have pointed out that ad hoc networks can be cheaper to build for small local area networks because they do not need much hardware. However, others make the point that a large number of devices can be difficult to manage without a larger and more concrete infrastructure [16]. Tech leaders are looking at ways to enable more vibrant network functionality with these peer-to-peer networks [16].

1.1 Introduction to Vehicular Ad-hoc Network (VANET)

Vehicular Ad-hoc network (VANET) [3] is considered as a special type of Mobile Ad-hoc Network (MANET) [3] [4], in which each node is a vehicle (i.e., car, bus, truck). This kind of networks has to face up new challenges. They are characterized by very high node mobility and topology changing that dependent using wireless technologies. These features make VANETs very prone to transmission errors, topology changes, and intermittent connectivity. It is an effect of a high moving speed of nodes and highly dynamic operating environments. So the primary goal for VANET is to achieve high packet delivery rates and low packet latency.

Last few years, car manufacturers have given vehicles the ability to generate and analyze large amounts of data, although this data associated only with a single vehicle. With the VANET technologies vehicles have the new ability to connect each over as well as to network infrastructures that companies in the last year have equipped into the highways (also a hybrid combination of them is possible). Roadside infrastructure also can be used as a gateway to the Internet. Thus data and context information can be collected, stored and processed somewhere (cloud computing). Communication can be either done directly between vehicles as one-hop communication or the vehicles those can retransmit messages thereby enabling multi-hop communication. Furthermore, packets can be transited in a unicast or multicast way.

Despite the difficult challenges, VANET represents an emerging wireless technology, allowing efficient communication among vehicles and fixed devices positioned along the street with a very favorable area of safety, traffic control, and user applications. Ongoing research is exploring protocol stacks and network architectures to efficiently afford these challenging issues.

The area of ad-hoc networks has experienced many years that contains a huge number of the variety of technologies (e.g., MANETs). Although VANET is one kind of MANETs that has some different feature and that makes it more challenging from other. Those challenges can be described as following.

• **High dynamic topology:** Vehicle change its position very frequently because of its speed. On the other hand, the topology is very large to cover and topology change frequently.

• **Frequently disconnected network:** VANET has a high dynamic topology and topology change often. As the topology change that makes the nodes out of range frequently. On the other hand when the topology density change that can make the problem also.

• **Unlimited battery power and storage:** As every vehicle act as a node, the power of every node provided by the vehicle that comes from fuel or gas of the car.

• **On board sensors:** Nodes consists of sophisticated sensors which provide very effective information such as GPS which gives location information.

1.2 Types of Communication in VANET

There are three types of VANET communication that can be established in a VANET network. Those are as follow.

1. Vehicle to Vehicle (V2V)
2. Vehicle to Infrastructure (V2I)
3. Hybrid communication.

1.2.1. Vehicle to Vehicle (V2V)

Vehicle to Vehicle (V2V) [3] communication in VANET networks is known as the vehicle to vehicle communications. It is an ad-hoc network that works among the vehicles such as cars, buses, trucks, etc. V2V allows the vehicles to send and receive the data from the vehicles inside the network. In this network system, a vehicle has some components that are used to detect position and movement of other nodes or vehicles. A simple antenna is used to transfer or receive the signal, and GPS is used for the detecting position and movement. As it is a vehicle to vehicle communication, the vehicles are directly connected with each other and they don't need to transfer or receive data from another component like roadside unit or roadside infrastructure. In this communication method, vehicles could share the information about the blond spot, accident information, and road condition. In this network, system vehicle can

anticipate and react to any dangerous driving situation informing drivers. If the driver does not act according to the information or responds to the alerts, then the vehicle can stop itself to avoid the collision.

V2V communication networks can communicate upmost quarter a mile and this networks equipped with a long-range scanning sensor for adaptive cruise control, forward vision sensor for object detection, mid-range blind spot detection sensor and long-range change assist sensor [3]. In this communication system, vehicle alerts the driver about the blind spot by focus a steady amber light in the side mirror. If the turn signal is activated, then there will be smooth seat vibration of the driver. It is good enough to grab the attention of the driver in a dangerous situation. V2V communication not only alerts in the dangerous situation but also in regardless of lanes, the vehicle ahead or any vehicle ahead break hard, allowing the driver to change the lane or stop the vehicle if needed.

In general, V2V is researched for applications linked to road safety but also to include entertainment applications. The technology draws on several disciplines, including transport engineering, electrical engineering, automotive engineering and computer science.

1.2.2 Vehicle to Infrastructure (V2I)

V2I is known as the vehicle to infrastructure. It is a little bit different than V2V communication. V2I is used to build up for a large network where vehicles are not very close to each other. As there are fewer chances to communicate with each other directly so, a roadside unit is used to transfer and receive the signal sent by the vehicles. In this communication system, the vehicles communicate with each other via a third component known as the roadside unit. As it has a roadside unit and it is a wireless network (WiFi, WiMAX or cellular). As there will be some roadside unit, so there should be an integrated communication system that is needed to manage those roadside units, and that is known as the vehicle infrastructure integration or VII.

This technology includes some engineering techniques to make it work. Transport engineering, electrical engineering, and computer science. The purpose of this communication system is in providing safety and efficiency. This idea can be

implemented in other modes also. For example, the airway can be directed, allowing autopilot to fly the plane without human intervention.

Finally, VII is that branch of engineering, which deals with the study and implementation of a series of techniques to achieve communication among vehicles and infrastructure bases to improve road safety.

1.1.3. Hybrid communication

Hybrid communication: Hybrid communication is the idea of building a network using both V2V and V2I technology. In this communication system, the vehicle is connected to each other as well as the roadside unit. It means this communication has the ability to communicate with a vehicle to vehicle and roadside unit. In this communication system, VANET can be benefited from both V2V and V2I.

1.3 Application of Vehicular Ad-Hoc Network

All services available into a VANET can be classified into three diff erent groups: safety, traffic control and user applications [5]. Firstly, safety message should be exchanged through the networks vehicle. It will help to avoid the probability of the accident. Secondly, traffic control is used for improving the quality and the control the vehicle movement of vehicle traffic. As this help to minimize the traffic jam and congestions. Thirdly, the user application is the service to the passenger which is mostly an entertaining part. A passenger can listen to music, chat with other friends, can watch the video like movie or serial as they want. Safety and traffic control is that kind of application that is nearly tied related because they can be combined to achieve the maximum road safety that will lead to the minimum road accident. This application can be used for two purposes one is car accident prevision and the second one is minimizing the road congestions.

• **Car accidents prevision:** As the vehicle moves very fast. Compare with the speed of the vehicle the reaction time of a driver is very low. So, if a driver does not reflex strongly for a few seconds, an accident can happen. So the safety and the traffic control that provide the warning and informations about the road situation and another factor that can lead to the accident.

• **Road congestion:** Safety and traffic control can be used to provide the best route to reach the destination. It also used to consider the traffic jam and other car accident. As a result the road congestions will be less in number and the traffic will flow smoothly. It could also lead to less car accident because the driver would be less frustrated and advised to keep the traffic rules. User Applications These applications could make the traveling more interesting and entertaining, given to passengers a pleasant trip.

There are two basic types:

1. Internet connection and
2. Peer-to-Peer (P2P).

1.3.1 Internet Connection

The web is a part of people day-life nowadays, so more and more people need to have a constant connection with social networks, e-mail, and all the useful services. VANET is challenging with this issue that also will make new opportunity for already or new-one services by the web.

1.3.2. Peer to Peer (P2P)

It is another interesting idea that will allow passengers to share news, music, videos, and any information among vehicles. Furthermore, it will be used for online chatting and gaming.

1.4 Background

In this section, we have discussed other existing papers, books, journals that we have followed regarding our thesis work. These papers, books, and journals helped us to evaluate our thesis work in the right direction.

In this paper [6], the authors worked on the misbehaving nodes. Existing networks rely mainly on node certificate revocation for attacker eviction, but the lack of an omnipresent infrastructure in VNs may unacceptably delay the retrieval of the most recent and relevant revocation information this will especially be the case in the early deployment stages of such a highly volatile and large-scale system. The authors addressed this specific problem. They have proposed some protocols, as components of a framework, for the identification and local containment of misbehaving or faulty nodes, and then for their eviction from the system. They have tailored their design to the VN characteristics and analyze our system. The results show that the distributed approach to contain nodes and contribute to their eviction is efficiently feasible and achieves a sufficient level of robustness. They have proposed a framework to thwart internal attackers in vehicular networks. The eviction of faulty or attacking nodes is crucial to the robustness of vehicular communication systems. As revocation is the primary means to achieve this, they have designed two protocols tailored to the characteristics of the VN environment. To eliminate the vulnerability window, due to the latency for the authority to identify faulty or misbehaving nodes and distribute revocation information, they have designed a scheme that can robustly and efficiently achieve their isolation, as well as contribute to their eventual revocation. It is done with the help of a misbehavior detection module and a distributed eviction protocol.

In this paper [7], the authors proposed a scheme that allows only the connected authentic vehicles to disseminate a message. It identifies the replay attackers, fake nodes and also the attackers that are disseminating false message as misbehaving vehicles. A certificate revocation list is generated by inserting the identification of all the misbehaving vehicles. Finally, the proposed scheme revokes the misbehaving vehicles to ensure network security. The performance of the proposed scheme is studied by a delay in authentication

verification, delay in message dissemination and delay in revocation. It outperforms the existing schemes.

In this paper [8], the authors have worked with a distributed authentication scheme called as Group Authentication Protocol (GAP) which is proposed to resolve the most conflicting security requirements such as group authentication and conditional privacy. GAP uses a session-based pseudonym to support anonymous communication. The proposed batch verification scheme as a part of the protocol poses a significant reduction in the message delay. Furthermore, it enables the vehicles to verify a large number of messages in the case of high vehicular density and also improves message loss ratio. A Trusted third party called Trusted Authority has the complete control of tracing the vehicles both benign and misbehaving vehicles. The proposed protocol also scales well when the number of messages has increased and improved the service rate.

In this paper [9], the authors have presented discovery of a routing protocol that will mitigate the detrimental effects of such malicious behavior, as to provide correct connectivity information. The proposed protocol guarantees that fabricated, compromised, or replayed route replies would either be rejected or never reach back the querying node. Furthermore, the protocol responsiveness is safeguarded under different types of attacks that exploit the routing protocol itself. The sole requirement of the proposed scheme is the existence of a security association between the node initiating the query and the sought destination. Specifically, no assumption is made regarding the intermediate nodes, which may exhibit arbitrary and malicious behavior. The scheme is robust in the presence of some non-colluding nodes and provides accurate routing information promptly. The protocol introduces a set of features, such as the requirement that the query verifiably arrives at the destination, the explicit binding of network and routing layer functionality, the consequent verifiable return of the query response over the reverse of the query propagation route, the acceptance of route error messages only when generated by nodes on the actual route, the query/reply identification by a dual identifier, the replay protection of the source and destination nodes and the regulation of the query propagation. The resultant protocol is capable of operating without the existence of an online certification authority or the complete knowledge of keys of all network nodes. Its sole requirement is that any two nodes that wish to communicate securely can simply establish a priori a shared secret, to

be used by their routing protocol modules. Moreover, the correctness of the protocol is retained irrespective of any permanent binding of nodes to IP addresses, a feature of increased importance for the open, dynamic, and cooperative MANET environments.

1.5 Problem Statement

In our thesis work, we have shown that under Public Key Infrastructure (PKI) when a Certificate Authority (CA) detects that anyhow a node or vehicle starts misbehaving or becomes malicious, the CA revokes the digital certificate of that misbehaving node by creating a certificate revocation list (CRL). The criteria of a node or vehicle to be malicious if messages that are outdated (aged), received beyond their expected area of propagation, or contradictory to the node's state are considered false [6]. When the CA revokes the certificate and creates a CRL, it broadcast the updated CRL to all nodes or vehicles so that, they can reject any request sent from the malicious or misbehaving vehicle.

We have also evaluated and analyzed the performance of three different VANET routing protocol. The main purpose of this analysis is to determine when the CA broadcast the updated CRL to all nodes which protocol is more efficient to propagate the message. These protocols are Ad Hoc On-Demand Vector (AODV), Destination Sequenced Distance Vector (DSDV) and Dynamic Source Routing (DSR). We have evaluated their performance by comparing average lost packet ratio and average end to end delay time. We have also used different numbers of nodes to see the expected significant result.

1.6 Goals and Objectives

The main goal of our thesis work is to implement Public Key Infrastructure (PKI) for the secure distribution of certificate revocation list (CRL) and perform a simulation so that the three protocols can be evaluated.

1.7 Scopes

In our thesis work, we have worked with self-generated Certificate Revocation List and implemented the Public Key Infrastructure (PKI) with the help of a certificate authority software. In future, we can propose to work with a software OpenCA which will help us to generate an actual CRL and implement PKI. The mobility model we have worked with is a Random Way Point mobility model. In future, we will introduce an actual mobility model with real-time scenarios.

1.8 Overview

In chapter 2 we have discussed the methodology of our work. In chapter 3 we have discussed the implementation of Public Key Infrastructure (PKI) and the simulation of the three protocols. In chapter 4 we have shown the result and analysis of our thesis work. In chapter 5 we have concluded our work.

Chapter 2

Methodology

In this chapter, we have briefed about the protocols and the simulation tools we have used in our thesis work. We have worked with three VANET routing protocols. Routing protocols specify how routers communicate with each other, distributing information that enables them to select routes between any two nodes on a computer network. Routing algorithm determines the specific choice of route.

2.1 Routing Protocols

We have worked with three different routing protocols. These are:

1. Ad hoc On-Demand Distance Vector (AODV)

2. Dynamic Source Routing Protocol (DSR)

3. Destination Sequence Distance Routing Protocol (DSDV).

2.1.1. Ad hoc On-Demand Distance Vector (AODV):

An Ad Hoc On-Demand Vector (AODV) is a routing protocol which is designed for wireless and mobile ad hoc networks (MANET). It is the descendant of another routing protocol Destination-Sequenced Distance Vector Routing (DSDV) which is based on Bellman-Ford Algorithm. AODV is therefore considered an on-demand algorithm that does not create any extra traffic for communication. Its algorithm enables dynamic, self-starting, multi-hop routing between participating mobile nodes that are wished to establish and maintain an ad hoc network. This protocol establishes routes to a destination on demand and support both unicast and multicast routing. It uses bi-directional links. The AODV protocol builds routes between nodes only if they are requested by the source nodes. The routes are maintained as long as they are needed by the source node. AODV allows mobile nodes to respond to link breakage and change in the network topology promptly. AODV is a loop-free algorithm, and by avoiding the Bellman-Fond “counting to infinity” problem, it offers a quick merged way when the ad hoc network topology changes. One of the distinguishing

features of AODV routing protocol is that it uses a destination sequence number for each route entry. The sequence number is created by the destination which includes along with any route information it sends to requesting modes [10]. By using the destination sequence numbers AODV ensures loop-freedom and a simple programmable routing protocol.

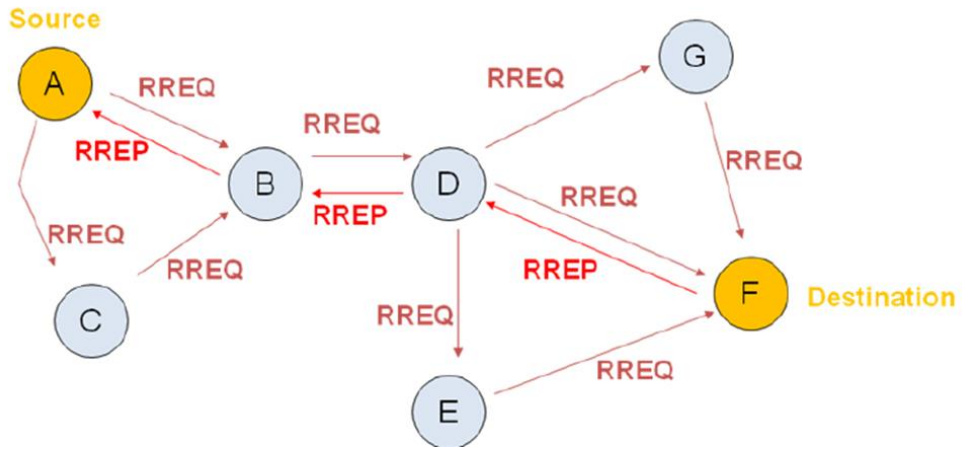

Figure 2.1.1: Ad hoc On-Demand Distance Vector (AODV) routing mechanism

Route Request (RREQs), Route Replies (RREPs) and Route Errors (RERRs) are the defined message types by AODV [10]. These message types are received via User Datagram Protocol (UDP), and normal IP header processing applies. To broadcast messages, the IP limited broadcast address (255.255.255.255) is used [10]. It concludes that these messages are not forwarded blindly. Therefore, AODV operations require certain messages (e.g., RREQ) to be propagated widely, throughout the ad hoc network.

AODV does not play any role as long as the endpoints of a converse connection that have valid routes connected to each other. When a route to a new destination needed, the source node continuously broadcasts a Route Request (RREQs) to find a route to the particular destination [10]. A route can be persuaded when the RREQ reaches either the desired destination itself or an intervening node that has a 'fresh enough' route to the destination. A 'fresh enough' route is a valid route entry for the destination whose associated sequence number is at least as great as that contained in the RREQ [10]. The route is made available by unicasting a Route Replies (RREP) back to the origin of the RREQ. Each node that receives the request caches a route back to the source that requested, so that the RREP can be unicast from the

destination along a path to the originator or any intervening node which can satisfy the request [10].

The link status of next hops in active routes is monitored by the intervening node. When a line break is detected, a Route Errors (RERR) message is used to send to notify the other nodes that the loss of that link has occurred [10]. The RERR message indicates that those destinations are no longer reachable by way of the broken link. To enable the reporting mechanism, each node keeps a "precursor list", which contains the IP address for each of its neighbors those who are likely to be used as a next hop heading for each destination [10]. The information in the "precursor lists" is most easily collected at the time of processing for generation of an RREP message, which has to be sent to a node in a precursor list. An RREQ message may also be apprehended for a multicast IP address. As an example, the source of an RREQ message for multicast may have to follow special rules.

AODV is a routing protocol that deals with some routing table management. All the routing table information must be kept even for short-lived routes, such as those which are created to temporarily store paths heading towards nodes that are the originator of RREQs. The following fields are used in AODV with each routes table entry [10]:

1. Destination IP address

2. Destination Sequence Number

3. Valid Destination Sequence Number Flag

4. Other state and routing flags (e.g., valid, invalid, repairable, being repaired)

5. Network Interface

6. Hop Count (number of hops needed to reach the destination)

7. Next Hop

8. List of Precursors

9. Lifetime (expiration or validation time of the selected route)

It is very essential for managing the sequence number to avoid routing loops, even when the links break and a node is no longer accessible to supply its information about its sequence number. A destination becomes no longer accessible when a link

breaks or is deactivated. When these type of conditions occur, the selected route is deleted or invalidated by operation that involves the sequence number and making the route table entry as invalid.

2.1.2. Dynamic Source Routing Protocol (DSR):

The Dynamic Source Routing protocol (DSR) [17] [18] is a simple and efficient routing protocol which is designed specifically for use in multi-hop wireless ad hoc networks of mobile nodes. Using DSR, the network is completely self-organizing and self-configuring, which does not require any existing network infrastructure or administration. The protocol is suppressed of the two main mechanisms of "Route Discovery" and "Route Maintenance", which work together that allow the nodes to discover and maintain routes to arbitrary destinations in the ad hoc network [12]. Network nodes co-operate to forward packets for each other to allow communication over multiple "hops" between nodes which are not directly within wireless transmission range of one another. As the nodes in the network move about or join or leave the network, and as wireless transmission conditions such as sources of interference change, all routing is automatically determined and maintained by the DSR routing protocol. Since the number or sequence of intermediate hops needed to reach any destination may change at any time, the resulting network topology may be quite rich and rapidly changing. The DSR protocol is composed of two main mechanisms that work together to allow the discovery and maintenance of source routes in the ad hoc network [12]:

- Route Discovery is the mechanism by which a node S wishing to send a packet to a destination node D obtains a source route to D. Route Discovery is used only when S attempts to send a packet to D and does not already know a route to D [12].

- Route Maintenance is the mechanism by which node S can detect while using a source route to D, if the network topology has changed such that it can no longer use its route to D because a link along the route no longer works. When Route Maintenance indicates a source route is broken, the node S also can attempt to use any other route it happens to know to D, or it can invoke Route Discovery again to find a new route for subsequent packets to D. Route Maintenance for this route is used only when S is actually sending packets to D.

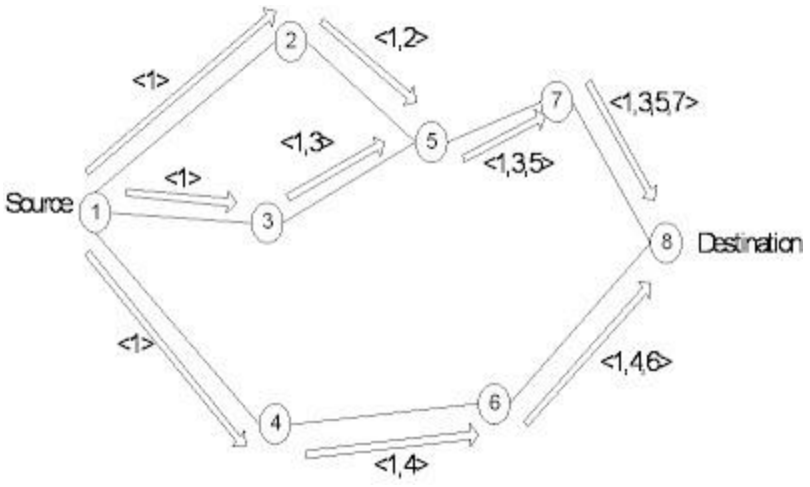

Figure 2.1.2(a): The Dynamic Source Routing Protocol (DSR) for multihop Wireless Ad Hoc Networks [12]

In DSR, Route Discovery and Route Maintenance both operate entirely "on demand". In particular, unlike other protocols, DSR requires no periodic packets of any kind at any layer within the network. For example [12], DSR does not use any periodic routing advertisement, link status sensing, or neighbor detection packets and does not rely on these functions from any underlying protocols in the network. These completely on-demand behavior and lack of periodic activity allow the number of overhead packets caused by DSR to scale all the way down to zero when all node is approximately stationary concerning each other and all routes are needed for current communication have already been discovered. As nodes begin to move more or as communication patterns change, the routing packet overhead of DSR automatically scales to only what is needed to track the routes currently in use. Network topology changes not affecting routes currently in use are ignored and do not cause a reaction from the protocol [12].

The operation of both Route Discovery and Route Maintenance in DSR are designed to allow unidirectional links and asymmetric routes to be supported. In particular, in wireless networks, it is possible that a link between two nodes may not work equally well in both directions, due to differing transmit power levels or sources of interference [12].

When some source node originates a new packet addressed to some destination node, the source node places in the header of the packet a "source route" giving the sequence of hops that the packet is to follow on its way to the destination [12].

Normally, the sender will obtain a suitable source route by searching its "Route Cache" of routes previously learned; if no route is found in its cache, it will initiate the Route Discovery protocol to dynamically find a new route to this destination node. In this case, we call the source node the "initiator" and the destination node the "target" of the Route Discovery. For example [12], suppose a node A is attempting to discover a route to node E. The Route Discovery initiated by node A in this example would proceed as follows:

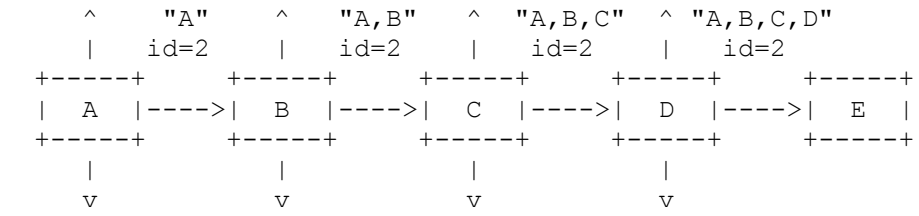

Figure 2.1.2(b): DSR Route Discovery [12]

To initiate the Route Discovery, node A transmits a "Route Request" as a single local broadcast packet, which is received by (approximately) all nodes currently within wireless transmission range of A, including node B in this example [12]. Each Route Request identifies the initiator and target of the Route Discovery, and also contains a unique request identification (2, in this example), determined by the initiator of the Request. Each Route Request also contains a record listing the address of each intermediate node through which this particular copy of the Route Request has been forwarded [12]. This route record is initialized to an empty list by the initiator of the Route Discovery. In this example, the route record initially lists only node A.

When another node receives this Route Request [12] (such as node B in this example), if it is the target of the Route Discovery, it returns a "Route Reply" to the initiator of the Route Discovery, giving a copy of the accumulated route record from the Route Request; when the initiator receives this Route Reply, it caches this route in its Route Cache for use in sending subsequent packets to this destination [12].

Otherwise, if this node receiving the Route Request has recently seen another Route Request message from this initiator bearing this same request identification and the target address, or if this node's address is already listed in the route record in the Route Request, this node discards the Request. Or, this node appends its address to the route record in the Route Request and propagates it by transmitting it as a local broadcast packet. In this example [12], node B broadcast the Route Request, which

is received by node C; nodes C and D each also, in turn, broadcast the Request, resulting in receipt of a copy of the Request by node E.

In returning the Route Reply to the initiator of the Route Discovery, such as in this example, node E replying back to node A, node E will typically examine its own Route Cache for a route back to A and, if one is found, will use it for the source route for delivery of the packet containing the Route Reply. Otherwise, E SHOULD perform its own Route Discovery for target node A, but to avoid possible infinite recursion of Route Discoveries, it MUST in this case piggyback this Route Reply on the packet containing its own Route Request for A. It is also possible to piggyback other small data packets, such as a TCP SYN packet [19], on a Route Request using this same mechanism.

When originating or forwarding a packet using a source route, each node transmitting the packet is responsible for confirming that data can flow over the link from that node to the next hop. For example, [12] in the situation shown below, node A has originated a packet for node E using a source route through intermediate nodes B, C, and D:

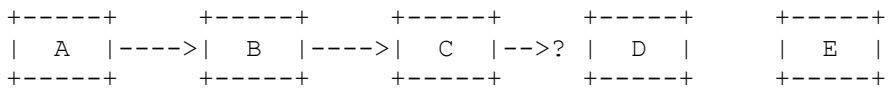

Figure 2.1.2(c): Basic DSR Route Maintenance [12]

In this case [12], node A is responsible for the link from A to B, node B is responsible for the link from B to C, node C is responsible for the link from C to D, and node D is responsible for the link from D to E.

2.1.3. Destination Sequence Distance Routing Protocol (DSDV):

Destination sequenced distance vector routing (DSDV) is adapted from the conventional Routing Information Protocol (RIP) to ad hoc networks routing. It adds a new attribute, sequence number, to each route table entry of the conventional RIP. Using the newly added sequence number, the mobile nodes can distinguish stale route information from the new and thus prevent the formation of routing loops.

In DSDV, each mobile node of an ad hoc network maintains a routing table, which lists all available destinations, the metric and next hop to each destination and a sequence number generated by the destination node. Using such routing table stored

in each mobile node, the packets are transmitted between the nodes of an ad hoc network. Each node of the ad hoc network updates the routing table with advertisement periodically or when significant new information is available to maintain the consistency of the routing table by the dynamically changing topology of the ad hoc network [13].

Periodically or immediately when network topology changes are detected, each mobile node advertises routing information using broadcasting or multicasting a routing table update packet. The update packet starts out with a metric of one to direct connected nodes [13]. This indicates that each receiving neighbor is one metric (hop) away from the node. It is different from that of the conventional routing algorithms. After receiving the update packet, the neighbors update their routing table with incrementing the metric by one and retransmit the update packet to the corresponding neighbors of each of them. The process will be repeated until all the nodes in the ad hoc network have received a copy of the update packet with a corresponding metric [13]. The update data is also kept for a while to wait for the arrival of the best route for each particular destination node in each node before updating its routing table and retransmitting the update packet. If a node receives multiple update packets for the same destination during the waiting period, the routes with more recent sequence numbers are always preferred as the basis for packet forwarding decisions, but the routing information is not necessarily advertised immediately if only the sequence numbers have been changed [13]. If the update packets have the same sequence number with the same node the update packet with the smallest metric will be used and the existing route will be discarded or stored as a less preferable route. In this case, the update packet will be propagated with the sequence number to all mobile nodes in the ad-hoc network. The advertisement of routes that are about to change may be delayed until the best routes have been found. Delaying the advertisement of a possibly unstable route can damp the fluctuations of the routing table and reduce the number of rebroadcasts of possible route entries that arrive with the same sequence number [13].

The elements in the routing table of each mobile node change dynamically to keep consistency with the dynamically changing topology of an ad hoc network. To reach this consistency, the routing information advertisement must be frequent or quick enough to ensure that each mobile node can almost always locate all the other mobile

nodes in the dynamic ad-hoc network. Upon the updated routing information, each node has to relay the data packet to other nodes upon request in the dynamically created ad hoc network.

In the routing [13] information updating process, the original node tags each update packet with a sequence number to distinguish stale updates from the new one. The sequence number is a monotonically increasing number that uniquely identifies each update from a given node. As a result, if a node receives an update from another node, the sequence number must be equal or greater than the sequence number of the corresponding node already in the routing table, or else the newly received routing information in the update packet is stale and should be discarded. If the sequence number of one node in the newly received routing information update packet is same as the corresponding sequence number in the routing table, then the metric will be compared and the route with the smallest metric will be used [13].

In addition to the sequence number and the metric for each entry of the update packet, the update route information contains also both the address of the final destination and the address of the next hop. There are two types of update packets, one is called full dump, which carries all of the available routing information. The other is called incremental, which carries only the routing information changed since the last full dump.

Links can be broken when the mobile nodes move from place to place or have been shut down etc. The broken link may be detected by the communication hardware or be inferred if no broadcasts have been received for a while from a former neighbor. The metric of a broken link is assigned infinity. When a link to next hop has broken, any route through that next hop is immediately assigned an infinity metric and an updated sequence number. Because link broken qualifies as a significant route change, the detecting node will immediately broadcast an update packet and disclose the modified routes.

To describe the broken links, any mobile node other than the destination node generates a sequence number, which is greater than the last sequence number received from the destination. This newly generated sequence number and a metric of infinity will be packed in an update message and flushed over the network. To avoid nodes themselves and their neighbors generating conflicting sequence numbers when the network topology changes, nodes only generate even sequence numbers for

themselves, and neighbors only generate odd sequence numbers for the nodes responding to the link changes.

The routes to a lost node will be re-established when the lost node comes back to the network and broadcasts its next update message with an equal or later sequence number and a finite metric. The update message will be disseminated over the whole network to indicate that the broken links have come back into service again. In any case, the entry containing a finite metric and an equal or later sequence number will supersede the corresponding entry with a metric of infinity in the routing table of a node.

DSDV also contains substantially more procedures for handling layer-2 and layer-3 routing and for dealing with the extension of base station coverage. The details of these procedures are referred to [13].

2.2 Simulation Tools

We have used some simulation tools to help with our thesis work. These are:

1. Network Simulator 2 (NS-2)

2. OpenSSL

2.2.1. Network Simulator 2 (NS-2):

NS-2 is a discrete-event network simulator, targeted primarily for research and educational use. NS-2 is free software, licensed under the GNU GPLv2 license and is publicly available for research, development, and use. The goal of the NS-2 project is to develop a preferred, open simulation environment for networking research: it should be aligned with the simulation needs of modern networking research and should encourage community contribution, peer review, and validation of the software [14]. NS-2 is available for Linux, Mac OS and MS Windows using Cygwin [15].

The NS-2 project is committed to building a solid simulation core that is well documented, easy to use and debug that caters to the needs of the entire simulation work-flow, from simulation configuration to trace collection and analysis. Furthermore, the NS-2 software infrastructure encourages the development of simulation models which are sufficiently realistic to allow NS-2 to be used as a real-

time network emulator, interconnected with the real world and which allows many existing real-world protocol implementations to be reused within NS-2.

The NS-2 simulation core supports research on both IP and non-IP based networks. However, the large majority of its users focuses on wireless/IP simulations which involve models for Wi-Fi, WiMAX or LTE for layers 1 and 2 a variety of static or dynamic routing protocols such as OLSR and AODV for IP-based applications. NS-2 also supports a real-time scheduler that facilitates some simulation-in-the-loop" use cases for interacting with real systems. For instance, users can emit and receive NS-2-generated packets on real network devices, and NS-2 can serve as an interconnection framework to add link eff ects between virtual machines. Another emphasis of the simulator is on the reuse of real application and kernel code. Frameworks for running unmodified applications or the entire Linux kernel networking stack within NS-2 are presently being tested and evaluated.

Because creating a network simulator that supports a sufficient number of high quality validated, and maintained models requires a lot of work, NS-2 attempts to spread this workload over a large community of users and developers. Every three months, it ships a new stable version of NS-2 with new models developed, documented, validated, and maintained by enthusiastic researchers. It encourages the open validation of these models by third parties on its mailing-lists to ensure that the models it ships and stay of the highest quality possible.

2.2.2. OpenSSL:

OpenSSL is a full-featured toolkit for the Transport Layer Security (TLS) and Secure Sockets Layer (SSL) protocols. SSL stands for Secure Sockets Layer, the protocol which provides the encryption. SSL certificates are typically installed on pages that require end-users to submit sensitive information over the internet. SSL certificates provide secure encrypted communication between a website and a webserver. It is also a general-purpose cryptography library. OpenSSL is written in the C programming language and relies on different ciphers and algorithms to provide encryption. The product is dual licensed under an Apache license and a Berkeley Software Distribution license. It is a free and open-source cryptographic library that provides several command-line tools for handling digital certificates. Some of these tools can be used to act as a certificate authority.

Chapter 3

Implementation

In information technology (IT) security is the defense of digital information and it assets against internal and external, malicious and accidental threats which includes detection, prevention and response to threats through the use of security policies, software tools and IT services. In this chapter, we have shown how we have implemented Public Key Infrastructure (PKI) and the implementation of the protocol in NS-2. A Public Key Infrastructure (PKI) is a set of roles, policies and procedure needed to create, manage, distribute, use, store, revoke the digital certificate, and manage public key encryption. The purpose of a PKI is to facilitate the secure electronic transfer of information for a range of network activities such as e-commerce, internet banking, and confidential email.

3.1 Certificates authorities (CA)

Certificate Authorities or CAs, issue Digital Certificates. A digital certificate is an electronic "passport" that allows a person, computer or organization to exchange information securely over the Internet using the public key infrastructure (PKI). Just like a passport, a digital certificate provides identifying information, is forgery-resistant and can be verified because it was issued by an official, trusted agency. A digital certificate may also be referred to as a public key certificate. Digital Certificates are verifiable small data files that contain identity credentials to help websites, people, and devices represent their authentic online identity (authentic because the CA has verified the identity). CAs play a critical role in how the Internet operates and how transparent, trusted transactions can take place online. CAs issue millions of Digital Certificates each year, and these certificates are used to protect information, encrypt billions of transactions, and enable secure communication. The PKI of CA utilizes a hybrid model of asymmetric and symmetric encryption, in addition to hashing functions, to guarantee data confidentiality, data integrity, and server authenticity.

Working with certificates means trusting someone else because a certificate contains a foreign signature combining a public key with identity information. Here comes the concept of Root and Intermediate Certificate Authorities.

Root certificate authority is the highest signer in the modern world. It publishes a so-called self-signed certificate to display its special role. A self-signed certificate is created by using own private key to certify your identity. Often a root CA does not directly offer certificates to users and companies but rather certifies another CAs to do this job. This CAs called Intermediate CAs. So intermediate CA can give anyone authority to do a specific task on behalf of Root CA. If any user misbehaves, it can also generate Certificate Revocation Lists (CRLs). A certificate is irreversibly revoked if, for example, it is discovered that the certificate authority (CA) had improperly issued a certificate, or if a private-key is thought to have been compromised. Certificates may also be revoked for failure of the identified entity to adhere to policy requirements, such as publication of false documents, misrepresentation of software behavior, or violation of any other policy specified by the CA operator or its customer. The most common reason for revocation is the user no longer being in sole possession of the private key.

3.2 Certification Process by CA

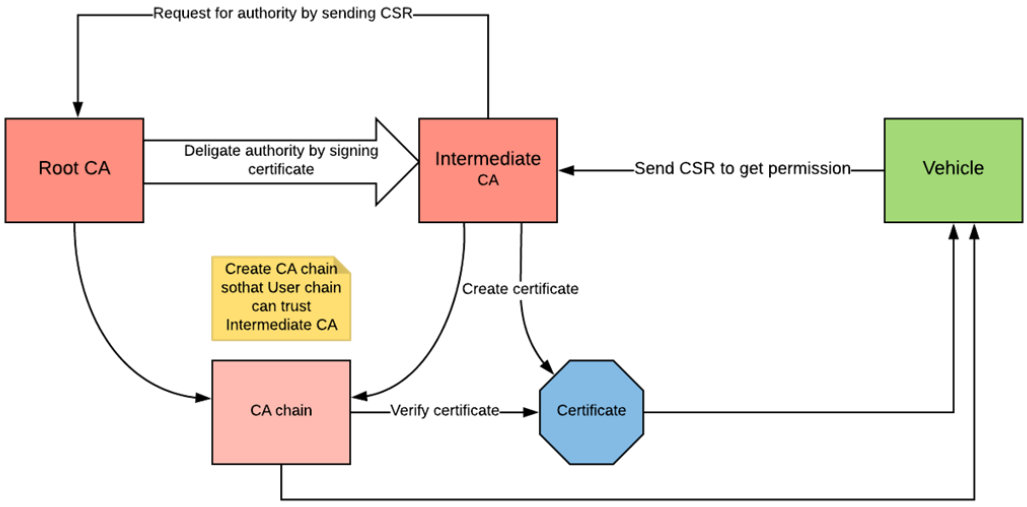

Figure 3.2: Certification process by Certification Authority (CA)

We already discussed before that we need Certification Authority (CA) for secure vehicular communication in VANET. In Certification Authority there is always top-level authority that is called Root CA. Root CA is the head of the certification process. As Root CA has the top authority so instead of giving a certificate to the user level, it delegates its authority to another CA. That CA is called intermediate CA.

When an intermediate CA get the permit to be a CA from Root CA, then it will handle all user-level certificate request. A vehicle/user will send a Certificate Signing Request (CSR) to the intermediate CA when a certificate is needed. If he is eligible for the certificate, then intermediate CA will sign the user request that means intermediate CA will create a certificate for that user. After creating the certificate, intermediate CA will send that certificate and CA-chain to that user. CA-chain is the combined certificate of Root CA certificate and Intermediate CA certificate. An Intermediate CA could be corrupted. The reason for CA-chain creation is user need to trust the Intermediate CA. When user sees the CA-chain along with his certificate, then he can trust that Intermediate CA. That means the user can ensure that his certificate is issued by a valid Intermediate CA.

3.3 Certificate Authority Delegation

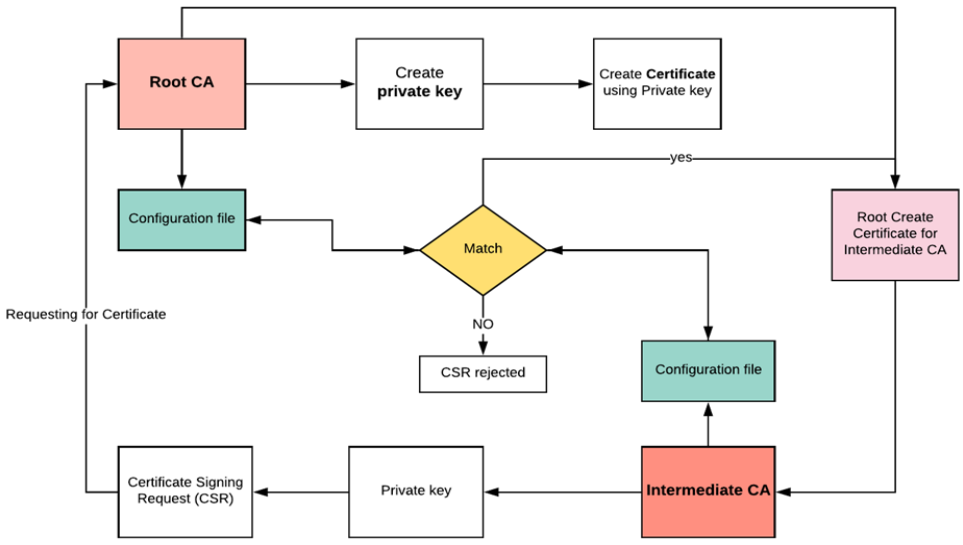

Figure 3.3: CA delegation

Here is the workflow how a Root CA delegate its authority to an Intermediate CA. At first Root CA creates its certificate by using his private key because there is no higher authority than him who can give him the certificate. Besides Root CA also create a configuration file that contains different default values for different variables and the directories of the keys and certificates.

Now a candidate who wants to become an intermediate CA will send his Certificate Signing Request (CSR) to the Root CA. His CSR is created with his private key. He also generates a configuration file for his settings. After getting his CSR root, CA will check the information that contains in CSR. If some particular information is matched with Root CA, then it will sign the CSR and create a certificate for that candidate. Then the certificate will be sent to that candidate. Now it can act as an Intermediate CA and sign any CSR that will come from the users. We implemented all these CA process using OpenSSL to see that how it works and we got the process steps as described before.

3.4 Certificate Revocation List (CRL) Distribution

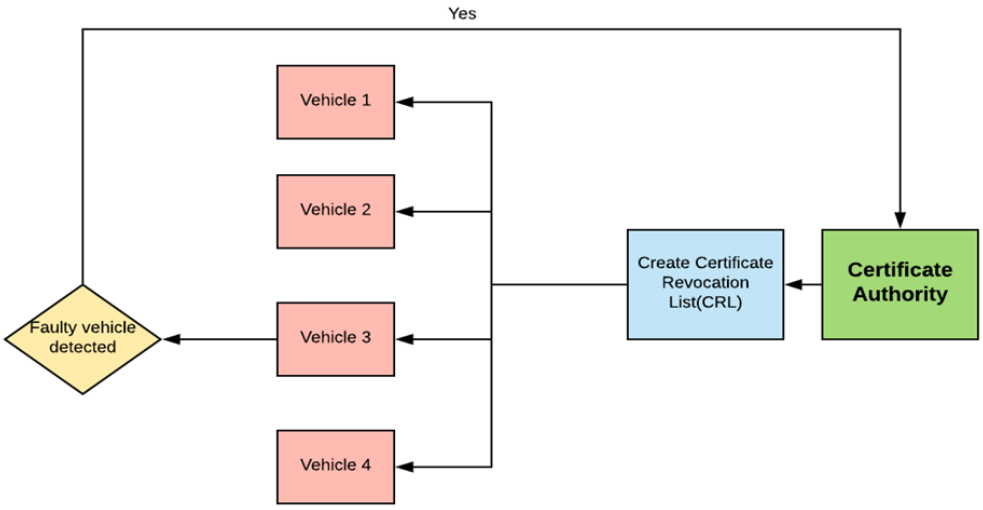

Figure 3.4: Certificate Revocation List distribution

CRL is created by a CA when it detects any vehicle is misbehaving or sending false information to the other vehicles. CRL contains all the faulty vehicles ID. When CRL is generated by CA, then it will be distributed to all the vehicles of that network. So that further no secure communication will be established with those faulty vehicles when all the vehicles of that network got that CRL.

3.5 Working Methodology of Simulation using NS-2

Ns-2 is a networks simulator that supports .tcl or .otcl file format. TCL is a tool command language that is used to transfer the command to the text editor, debugger or shell. When ns-2 execute the .tcl file, usually it generate two file one is NAM file another is TRAACE file. Nam is a Tcl/TK based animation tool for viewing network simulation traces and real world packet traces. It supports topology layout, packet level animation, and various data inspection tools [2]. .tr or trace file is nothing but a log file. It contains the information of data sent receive packet size events and other information about simulation of the .tcl file. Another type of file we are using to simulate is the .tr file known as the .awk script.

AWK is a high-level programming language which is used to process text files. .awk file has some fixed rules to write that script the procedure of writing .awk program structure contains mainly three parts, Begin, Content and End.

For simulating purpose, we have written a .awk file name as pdr.awk see appendix B. In the .awk file, we have collected the information from the trace file log. In that .awk file, we have calculated the sent packet, received packet, packet delivery ratio and Lost packet ratio. For calculating the packet delivery and lost packet ratio we have used two equation respectively:

$$Lost\ Packet\ ratio = \frac{\text{Receive packet}}{\text{Sent Packet}} * 100$$

$$Lost\ packet\ ratio = \frac{\text{Sent packet} - \text{Receive packet}}{\text{Sent packet}} * 100$$

If we draw a flowchart of the procedure of our simulation using ns-2. The below figure shows the flowchart of the execution of the procedure.

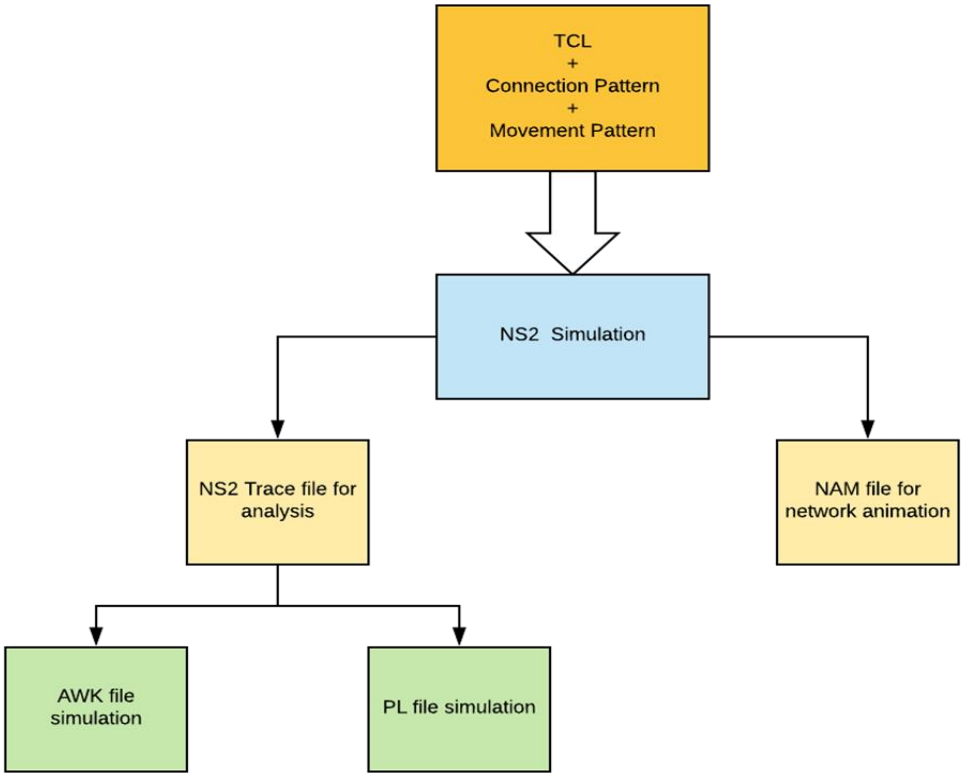

Figure 3.5: Working methodology of simulation using ns-2

3.6 Simulation Parameters of NS-2

For comparing the performance of three protocol, we used some predefine parameters those are the same for all protocol except the protocol name.

TABLE 3.6: Parameters' Description

Parameter name	Parameter description
Channel	Type's transaction of a data base on medium wired or wireless communication.
Propagation model	Radio propagation model two Ray Ground or Radio wave propagation.
Network interface type	Types of network interface physical or wireless.
Mac Type	Types of medium control access We have used ITEE802.11
Interface queue type	In our comparison, we use FIFO (First in First Out) interface queue type.
Link layer	Which layer the packet will transfer. We work on data link layer for transferring packet.
Antenna model	Antenna can be Omni-directional or dipole antenna. We have used Omni-directional antenna.
Maximum interface queue value	10,50,30
Vehicle node	5,15,30
Routing protocol	AODV, DSDV, DSR.
X dimension value	Integer value to declare the topology area in vertical.
Y dimension value	Integer value to declare the topology area in horizontal.
Time of simulation end	40,45,50,55,75,100,125,150,175,200,225,250,275,300

Chapter 4

Result and Analysis

In this chapter, we have shown our finding of the protocols' performances. We have used this parameter (discussed in section 3.6) to determine the performance of three protocol. We have to set this values as a parameter to determine the performance of three protocol. After setting the parameter, we have simulate the .tcl file for a various parameters such as vehicle number and simulation duration. For calculating the packet delivery and lost packet ratio we have used two equation respectively:

$$Lost\ Packet\ ratio = \frac{\text{Receive packet}}{\text{Sent Packet}} * 100$$

$$Lost\ packet\ ratio = \frac{\text{Sent packet} - \text{Receive packet}}{\text{Sent packet}} * 100$$

4.1 Simulation Result for AODV

First, we simulate the .tcl file for vehicle number 5 and simulation duration time 40,45,50,55. If we simulate the .tcl file using NS-2 simulator then we have got some value as shown in table 4.1

TABLE 4.1(a): For Vehicle 5

AODV					
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
5	40	89	78	12.3935955	246.493
5	45	508	485	4.52756	207.606
5	50	926	903	2.4838	222.933
5	55	1344	1321	1.71131	230.613
Average				5.279066375	226.91125

First, we execute the simulation for AODV protocol. Here we get some value for five vehicles when we change the simulation duration 40,45,50,55 then we execute the pdr.awk. See Appendix B. From the simulation we get the lost packet ratio and average end to end delay by executing delay.awk (see Appendix B). Here we have a different parameter one is vehicle number or node number and a different simulation duration. So we get the average in lost packet ratio and average end to end delay.

Here is another table for vehicle number or node number 15. Here we get the same kind of table with different value in it. If we see those value of the simulation we have seen that, we have two different parameter so as before we make an average of lost packet ratio and average end to end delay.

TABLE 4.1(b): For Vehicle 15

Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
15	40	1552	1532	1.218866	250.987
15	45	1970	1950	1.01523	250.196
15	50	2388	2368	0.83752	250.141
15	55	2805	2785	0.71301	249.756
Average				0.9461565	250.27

Same procedure for 30 vehicle or nodes.

TABLE 4.1(c): For Vehicle 30

Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
30	40	1551	1531	1.28949	249.908
30	45	1971	1951	1.01471	249.155
30	50	2387	2367	0.83787	248.93
30	55	2805	2785	0.71301	249.095
Average				0.96377	249.272

As we change two parameter vehicle number (node number) and simulation duration for the next simulation purpose we will change only one parameter, and that will be simulation duration. Now the simulation will be executed on 30 vehicle or node numbers. For getting the result of the simulation, we have to follow the whole process for every simulation duration time. Now we execute the delay.pdr, analyze.pl and pdr.awk on protocol Name.tcl and put those data on a table.

TABLE 4.1(d): For Simulation Duration AODV

(AODV)					
Time (sec)	Sent Packet	Receive Packet	Packet Delivery Ratio	Lost Packet Ratio (%)	Avg. End to End delay (ms)
50	4279	4056	94.7885	5.2115	149.57
75	7626	7219	94.663	5.337	155.448
100	12423	11903	95.81422	4.18578	149.912
125	18253	17548	96.13762	3.86238	119.982
150	21321	20485	96.07898	3.92102	122.223
175	24043	23113	96.13193	3.86807	123.128
200	28737	27585	95.99123	4.00877	123.128
225	32929	31595	95.94886	4.05114	113.862
250	40574	39045	96.23158	3.76842	112.57
275	48110	46427	96.50177	3.49823	108.185
300	57135	55231	96.66754	3.33246	100.78

If we fixed the vehicle number 30 and change the simulation duration respectively 50,75,100,125,150,175,200,225,250,275,300. All the value of simulation duration in second.

4.2 Simulation Result for DSR

So, we have to go through the whole process for DSR protocol as well. If we simulate the .tcl file for DSR protocol with the same parameter, then we have got some result.

TABLE 4.2(a): For Vehicle 5, 15 and 30

DSR					
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
5	40	806	786	2.48139	326.189
5	45	1215	1195	1.64609	301.224
5	50	1621	1601	1.23381	290.41
5	55	2402	2381	0.87427	228.032
Average				1.55889	286.46375
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
15	40	1615	1595	1.23839	253.378
15	45	2022	2002	0.98912	252.463
15	50	2429	2409	0.82338	252.013
15	55	3099	3089	0.64537	217.015
Average				0.924065	243.71725
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
30	40	1693	1672	1.18203	253.348
30	45	2097	2077	0.95374	253.758
30	50	2423	2403	0.82542	253.371
30	55	33107	3087	0.64371	227.537
Average				0.901225	247.0035

As we change two parameter vehicle number (node number) and simulation duration for the next simulation purpose we will change only one parameter and that will be simulation duration. Now the simulation will be executed on 30 vehicle or node numbers. If we fixed the vehicle number 30 and change the simulation duration respectively 50,75,100,125,150,175,200,225,250,275,300. All the value of simulation duration in second.

TABLE 4.2(b): For Simulation Duration DSR

DSR					
Time (sec)	Sent Packet	Receive Packet	Packet Delivery Ratio	Lost Packet Ratio (%)	Avg. End to End delay (ms)
50	6087	6010	98.73501	1.26499	303.649
75	11209	11081	98.85806	1.14194	327.551
100	15815	15682	99.15903	0.84097	351.162
125	21437	21295	99.33759	0.66241	314.474
150	23292	23103	99.18856	0.81144	310.519
175	26388	26156	99.12081	0.87919	302.918
200	31490	31205	99.09495	0.90505	299.717
225	36856	36530	99.11548	0.88452	305.255
250	43565	43202	99.16676	0.83324	312.518
275	50919	50551	99.27728	0.72272	299.026
300	59759	59391	99.38419	0.61581	291.902

4.3 Simulation Result for DSDV

As we mentioned at the very beginning we are comparing three protocol. So, we have to go through the whole process for DSDV protocol as well. If we simulate .tcl file for DSDV protocol with the same parameter then we have got some result. If we show those result of the simulation then we will get the result.

TABLE 4.3(a): For Vehicle 5, 15 and 30

DSDV					
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio (%)	Avg. End to End Delay (ms)
5	40	317	298	5.99369	177.849
5	45	725	703	3.03448	212.672
5	50	1136	1118	1.58451	214.132
5	55	1736	1697	2.24654	184.216
Average				3.214805	197.21725
Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio	Avg. End to End Delay
15	40	6	2	66.6666	2540.62
15	45	6	2	66.6666	2540.62
15	50	315	296	6.03175	112.256
15	55	1133	1110	2.03001	71.3545
Average				35.34874	1316.212625

Number of Nodes	Simulation Duration	Sent Packet	Receive Packet	Lost Packet Ratio	Avg. End to End Delay
30	40	6	2	66.6666	2538.73
30	45	324	304	6.17284	166.401
30	50	720	697	3.19444	166.147
30	55	1375	1343	2.32727	126.618
Average				19.5902875	749.474

As we change two parameter vehicle number (node number) and simulation duration for the next simulation purpose we will change only one parameter and that will be simulation duration. Now the simulation will be executed on 30 vehicle or node numbers. If we fixed the vehicle number 30 and change the simulation duration respectively 50,75,100,125,150,175,200,225,250,275,300. All the value of simulation duration in second.

TABLE 4.3(b): For Simulation Duration DSDV

(DSDV)					
Time (sec)	Sent Packet	Receive Packet	Packet Delivery Ratio	Lost Packet Ratio (%)	Avg. End to End delay (ms)
50	5747	5637	98.08596	1.91404	65.009
75	11513	11274	97.92409	2.07591	56.6233
100	17494	17138	97.96502	2.03498	52.8804
125	23376	22894	97.93806	2.06194	51.4187
150	23770	23259	97.85023	2.14977	51.2314
175	24428	23894	97.81398	2.18602	51.9256
200	29519	28794	97.54395	2.45605	53.1964
225	35455	34591	97.56311	2.43689	52.2579
250	41473	40463	97.56468	2.43532	53.9103
275	46718	45523	97.4421	2.5579	58.5937
300	53862	52481	97.43604	2.56396	61.6204

4.4 *Graph Analysis*

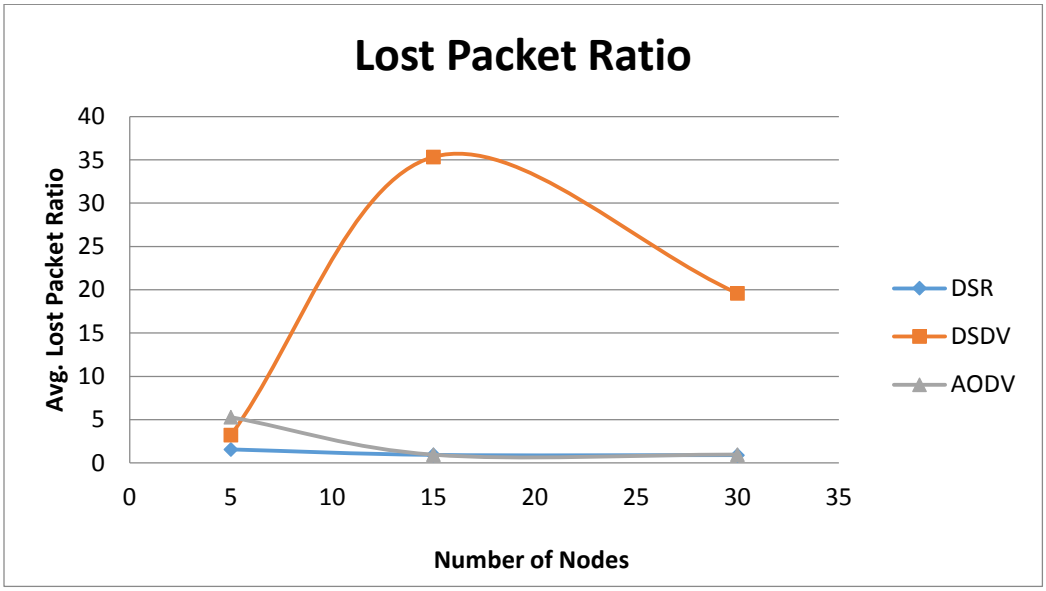

Figure 4.4(a): Lost packet ratio for a different number of nodes (5, 15, 30)

From the above figure, we can see that DSDV protocol has the most lost packet ratio whereas two other protocol has a nearly same number of lost packet ratio. So, for the various number of nodes DSDV has the worst performance. But if we concentrate on the figure, then we can see that DSR has the lowest number of lost packet ratio among three protocol.

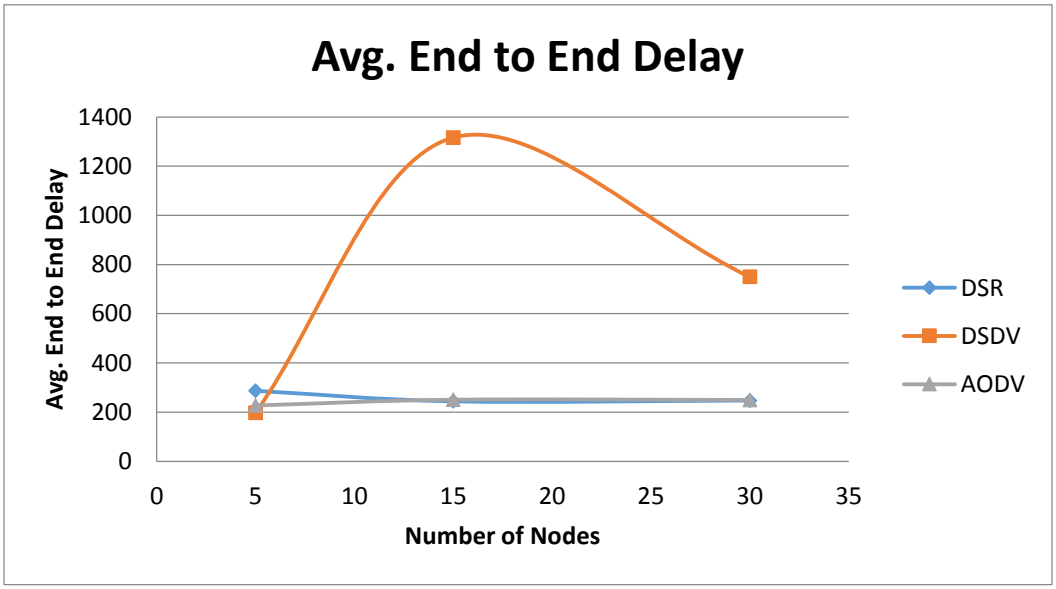

Figure 4.4(b): Avg. End to end delay for a different number of nodes (5, 15, 30)

From the figure, we can see that DSDV has the worst performance for both the average lost packet ratio and the average end to end delay. On the other hand AODV and DSR have slightly close performance on the average lost packet ratio and the average of average end to end delay. From this above graph we can conclude that for a lower number of vehicle or node DSDV is not perfect option to choose.

Now we have got the value of a sent packet, received packet number, packet delivery rate, lost packet ratio, and average end to end delay. Now if we draw the graph for sent packet with respect to simulation duration.

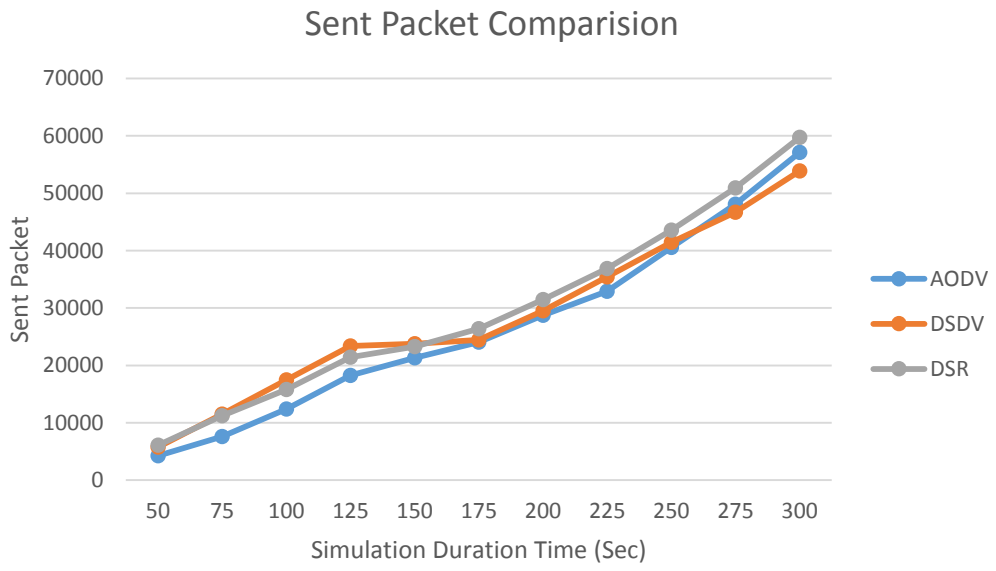

Figure 4.4(c): Sent packet ratio with respect to simulation duration

If we consider the graph, then we can see that as the simulation time goes up the number of packets sent. In VANET as the simulation time increase then the vehicle of the network transfer more data to communicate. As more data sent to communicate then more the data packet will generate.

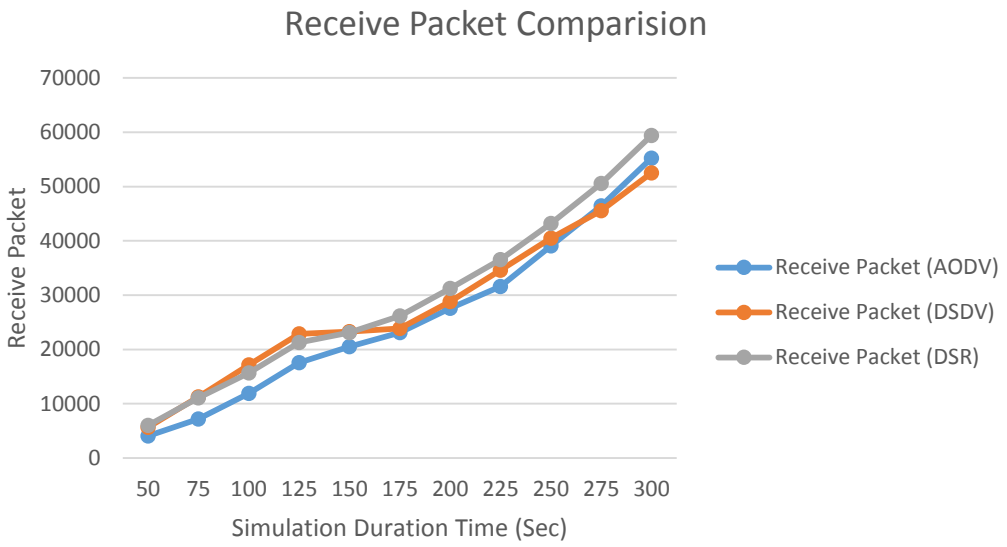

Figure 4.4(d): Receive packet ratio in terms of simulation duration time.

From the previous graph, we can see that when the simulation time goes up, the sent packet number goes up. No the logic behind the received packet is same as the vehicle node sent more data then it will receive more data as well because the message is sent as a broadcast. So, we can see that as the simulation time goes up the number of received packet increase exponentially.

Now as the same procedure if we draw a graph for lost packet ratio and an end to end delay, then the graph will look like as below in respect of simulation duration.

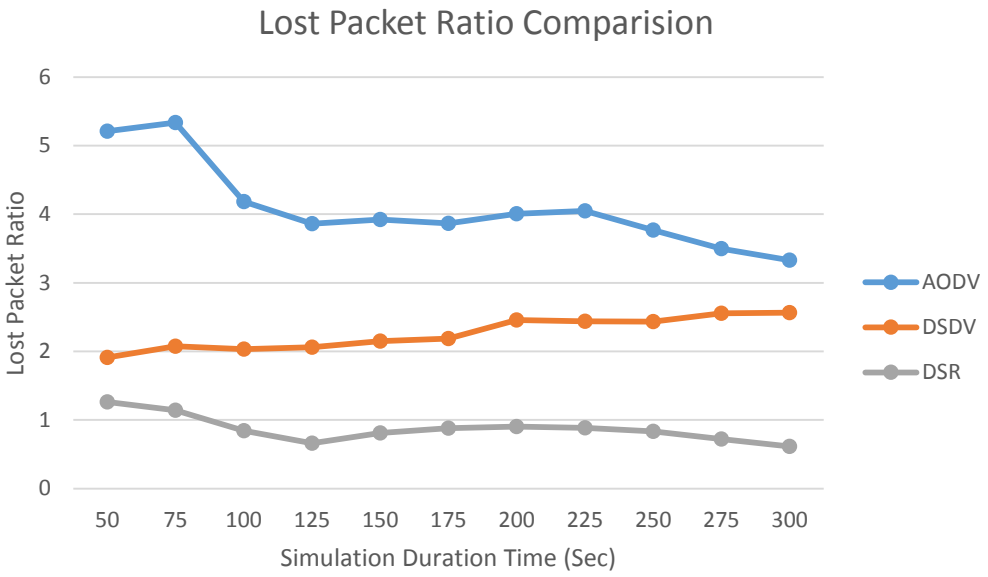

Figure 4.4(e): lost packet ratio with respect to simulation duration time (sec)

From this graph, we can conclude that DSR has the lowest packet lost ratio. On the other hand, AODV protocol has the maximum number of lost packet ratio. DSDV has acquired around the middle point among three protocol. If the networks designer wants to make sure nearly all packet transfer to all the node then DSR is the best option for transferring the packet. Another perspective is that AODV has decreased the lost packet ratio as the simulation time goes up. This also can be concluded that when the simulation time duration goes up, AODV will give a reasonable performance. But as far as we can see that according to lost packet ratio DSR has the best performance, because the lost packet ratio of this protocol nearly 1.5 maximum and less than 1 is the lowest lost packet ratio.

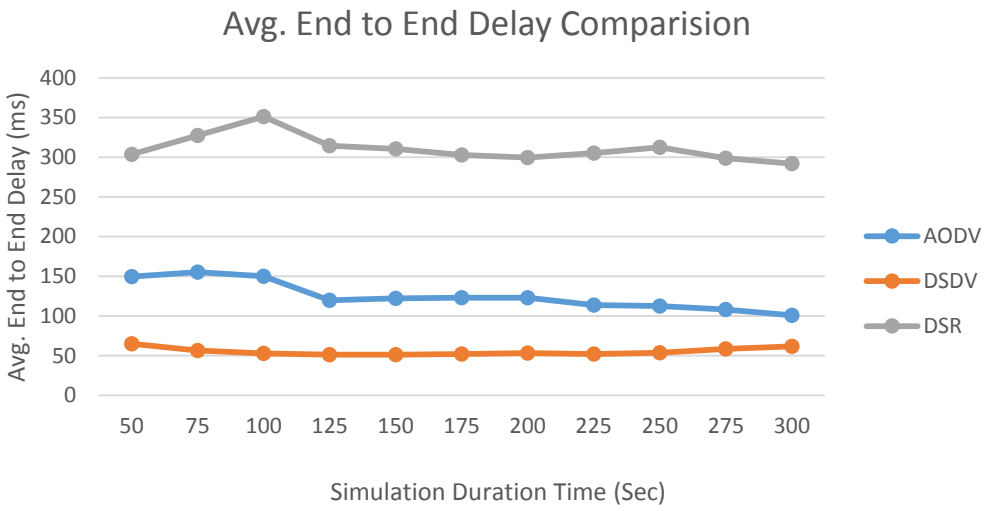

Figure 4.4(f): Average end to end delay (ms) with respect to simulation time duration (sec).

If we consider the figure which is drawn with the value of simulation duration time and the average End to End delay here DSR has the worst performance. As DSR took much time to send data packet one node to another. DSDV has the lowest average End to End delay and among three protocol AODV has got the middle point. DSDV took minimum time (ms) to send data one node to another.

Chapter 5

Discussion and Conclusion

5.1 Discussion

From the above simulation data and graph, we can conclude a point that is if the designer of the networks topology wants to transfer the packet fast then DSDV is the best option to distribute the CRL list to all the vehicle of that networks not considering the lost packet ratio. On the other hand, If the designer wants to send all the data to the and not worry about the time than DSR is the best option for distributing the CRL to all vehicle on that networks.

5.2 Conclusion

This works will implement revocation of misbehaving vehicles during data transmission. Such revokes will helps to reduce the number of misbehaving nodes during VANET movement. It will help to secure the movement of the vehicle. It will reduce the false vehicle entry robbery, or any false message will be reduced in number if this implementation in real life. However, given the importance of security and safety of human lives in VANET, the solution can be made more realistic by detecting other network attacks as well as other network models. The simulation experiment can be conducted in a distributed VANET to accommodate more vehicles and more events. Data security as it concerns the measures undertaken for the protection of data from accidental or intentional but unauthorized modification, destruction or disclosure through the use of physical security, administrative controls, logical controls, and other safeguards to limit accessibility. Data Governance is a key data management discipline. It is now more important than ever in time to adopt a strong data governance approach before the volume, variety and velocity of data increase further.

5.3 Future Works

In our work, we did not send the actual certificate revocation list (CRL). In future we will try to send an actual CRL. In this regard, we have implemented the public key infrastructure in OpenSSL not in OpenCA. We have limited the number of nodes in 30. In future, we will increase the number of nodes to get a better result. We have worked with Random Way Point Mobility Model. We did not implement any actual mobility models. In our future work we will use an actual Mobility Mode to see real time vehicle communication.

References

[1]. Toh, C. (2002). Ad hoc mobile wireless networks. Upper Saddle River: Prentice Hall.

[2]. C. Siva Ram Murthy. (2012). Ad Hoc Wireless Networks. Upper Saddle River, N.J.: Prentice hall.

[3]. Toh, Chai-Keong. Ad Hoc Mobile Wireless Networks: Protocols and Systems. Prentice-Hall, 2002.

[4]. Zanjireh, M. and Larijani, H. (2015). A Survey on Centralised and Distributed Clustering Routing Algorithms for WSNs. 2015 IEEE 81st Vehicular Technology Conference (VTC Spring).

[5]. Toor, Y., Muhlethaler, P., Laouiti, A. and La Fortelle, A. (2008). Vehicle Ad Hoc networks: applications and related technical issues. IEEE Communications Surveys & Tutorials, 10(3), pp.74-88.

[6]. Raya, M., Papadimitratos, P., Aad, I., Jungels, D. and Hubaux, J. (2007). Eviction of Misbehaving and Faulty Nodes in Vehicular Networks. IEEE Journal on Selected Areas in Communications, 25(8), pp.1557-1568.

[7]. Mondal, A. and Mitra, S. (2017). Revocation of misbehaving vehicles during data dissemination among connected vehicles in VANET. 2017 IEEE Region 10 Symposium (TENSYMP).

[8]. Priya, K. and Karuppanan, K. (2011). Secure privacy and distributed group authentication for VANET. 2011 International Conference on Recent Trends in Information Technology (ICRTIT).

[9]. Papadimitratos, P. and J. Haas, Z. (2002). Secure Routing for Mobile Ad hoc Networks. [ebook] San Antonio, Texas: Proc. SCS Comm. Networks and Distributed System Modelling Conf. (CNDS), 2002, 2002. Available at: https://www.researchgate.net/publication/2589156_Secure_Rotuing_for_Mobile_Ad_Hoc_Networks [Accessed 20 Apr. 2018].

[10]. Perkins C., Belding-Royer E. and Das S. (2003). Ad hoc On-Demand Distance Vector (AODV) Routing. [ebook] Santa Barbara: Nokia Research Center. Available at: https://tools.ietf.org/rfc/rfc3561.pdf.

[12]. Johnson, D., Maltz, D. and Hu, Y. (2007). The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4. [ebook] Network Working Group. Available at: https://tools.ietf.org/rfc/rfc4728.pdf.

[13]. Perkins Charles E., Bhagwat Pravin: Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, London England UK, SIGCOMM 94-8/94

[14]. NS-2. Network simulator. available at http://www.nsnam.org/, 2017

[15]. cygwin. Cygwin , linux feeling - on windows. available at http://www.cygwin.com/, 2011.

[16]. Techopedia.com. (n.d.). What is an Ad Hoc Network? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/5868/ad-hoc-network [Accessed 22 Apr. 2018].

[17] [JOHNSON96a] David B. Johnson and David A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, chapter 5, pages 153-181. Kluwer Academic Publishers, 1996.

[18] [JOHNSON96b] David B. Johnson and David A. Maltz. Protocols for Adaptive Wireless and Mobile Networking. IEEE Communications, 3(1):34-42, February 1996.

[19] [RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

Appendix A

Pdr File:

```
BEGIN {
        nSentPackets = 0;
        nReceivedPackets = 0;
        }
{
strEVENT = $1;
strAgt =   $4;
#printf( "string: %s\n",strEVENT);

if(strEVENT == "s" && strAgt == "AGT" ) {
        nSentPackets +=1;
        #printf( "nPacket: %d\n",nSentPackets );
}
if(strEVENT == "r" && strAgt == "AGT") {
        nReceivedPackets+=1;
        #printf( "nReceive: %d\n",nReceivedPackets );
        }
}
END {
        printf( "SentPacket: %d\n",nSentPackets );
        printf( "ReceivePacket: %d\n",nReceivedPackets );
        rPacketDeliveryRatio = nReceivedPackets / nSentPackets * 100;
        printf( "PacketDeliveryRatio: %.5f\n",rPacketDeliveryRatio );

        lpr = nSentPackets-nReceivedPackets;
        lpr = (lpr / nSentPackets) * 100;
        printf( "lpr: %.5f\n",lpr );
}
```

Appendix B

TCL file

```
# A 100-node example for ad-hoc simulation with AODV
# Define options
set val(chan)          Channel/WirelessChannel    ;# channel type
set val(prop)          Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)         Phy/WirelessPhy      ;# network interface type
set val(mac)           Mac/802_11           ;# MAC type
set val(ifq)         Queue/DropTail/PriQueue  ;# interface queue type
set val(ll)            LL                   ;# link layer type
set val(ant)           Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)        50                 ;# max packet in ifq
set val(nn)            20                   ;# number of mobilenodes
set val(rp)            AODV                     ;# routing protocol
set val(x)             500                  ;# X dimension of topography
set val(y)             400                  ;# Y dimension of topography
set val(stop)          100                    ;# time of simulation end
```

```
set ns          [new Simulator]
set tracefd       [open testAODV.tr w]
set windowVsTime2 [open win.tr w]
set namtrace      [open testAODV.nam w]


$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)

# set up topography object
set topo       [new Topography]

$topo load_flatgrid $val(x) $val(y)

create-god $val(nn)

#
#  Create nn mobilenodes [$val(nn)] and attach them to the channel.
#

# configure the nodes
 $ns node-config -adhocRouting $val(rp) \
-llType $val(ll) \
 -macType $val(mac) \
  -ifqType $val(ifq) \
  -ifqLen $val(ifqlen) \
-antType $val(ant) \
 -propType $val(prop) \
-phyType $val(netif) \
-channelType $val(chan) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace OFF \
-movementTrace ON

for {set i 0} {$i < $val(nn) } { incr i } {
            set node_($i) [$ns node]
            $node_($i) set X_ [ expr 10+round(rand()*480) ]
            $node_($i) set Y_ [ expr 10+round(rand()*380) ]
            $node_($i) set Z_ 0.0
}

for {set i 0} {$i < $val(nn) } { incr i } {
            $ns at [ expr 15+round(rand()*60) ] "$node_($i) setdest [ expr 10+round(rand()*480) ] [
            expr 10+round(rand()*380) ] [ expr 2+round(rand()*15) ]"

}

# Generation of movements
# $ns at 10.0 "$node_(0) setdest 250.0 250.0 3.0"
# $ns at 15.0 "$node_(1) setdest 45.0 285.0 5.0"
# $ns at 70.0 "$node_(2) setdest 480.0 300.0 5.0"
# $ns at 20.0 "$node_(3) setdest 200.0 200.0 5.0"
# $ns at 25.0 "$node_(4) setdest 50.0 50.0 10.0"
# $ns at 60.0 "$node_(5) setdest 150.0 70.0 2.0"
```

```
# $ns at 90.0 "$node_(6) setdest 380.0 150.0 8.0"
# $ns at 42.0 "$node_(7) setdest 200.0 100.0 15.0"
# $ns at 55.0 "$node_(8) setdest 50.0 275.0 5.0"
# $ns at 19.0 "$node_(9) setdest 250.0 250.0 7.0"
# $ns at 90.0 "$node_(10) setdest 150.0 150.0 20.0"

# Set a TCP connection between node_(2) and node_(8)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $node_(2) $tcp
$ns attach-agent $node_(8) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"



set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $node_(5) $tcp
$ns attach-agent $node_(0) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"

# Printing the window size
proc plotWindow {tcpSource file} {
global ns
set time 0.01
set now [$ns now]
set cwnd [$tcpSource set cwnd_]
puts $file "$now $cwnd"
$ns at [expr $now+$time] "plotWindow $tcpSource $file" }
$ns at 100.1 "plotWindow $tcp $windowVsTime2"

# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
        # 30 defines the node size for nam
        $ns initial_node_pos $node_($i) 30
}
# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
        $ns at $val(stop) "$node_($i) reset";
}

# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "stop"
$ns at 150 "puts \"end simulation\" ; $ns halt"
proc stop {} {
 global ns tracefd namtrace
 $ns flush-trace
```

```
close $tracefd
 close $namtrace
}$ns run
```

Appendix C

```
# ======================================================================

# AWK Script for calculating:

#     => Average End-to-End Delay.

# ======================================================================


BEGIN {

seqno = -1;

#    droppedPackets = 0;

#    receivedPackets = 0;

 count = 0;

}

{

if($4 == "AGT" && $1 == "s" && seqno < $6) {

 seqno = $6;

}
#            else if(($4 == "AGT") && ($1 == "r")) {

#            receivedPackets++;

#    } else if ($1 == "D" && $7 == "tcp" && $8 > 512){

#            droppedPackets++;

#  }

#end-to-end delay

 if($4 == "AGT" && $1 == "s") {

start_time[$6] = $2;

} else if(($7 == "tcp") && ($1 == "r")) {

 end_time[$6] = $2;
 } else if($1 == "D" && $7 == "tcp") {
```

```
  end_time[$6] = -1;
 }
}
END {
for(i=0; i<=seqno; i++) {
if(end_time[i] > 0) {
        delay[i] = end_time[i] - start_time[i];
   count++;
 }
else
{
        delay[i] = -1;
}
 }
for(i=0; i<=seqno; i++) {
         if(delay[i] > 0) {
          n_to_n_delay = n_to_n_delay + delay[i];
}
 }
 n_to_n_delay = n_to_n_delay/count;
 print "\n";
print "GeneratedPackets          = " seqno+1;
print "ReceivedPackets           = " receivedPackets;
 print "Packet Delivery Ratio     = " receivedPackets/(seqno+1)*100
"%";
 print "Total Dropped Packets = " droppedPackets;
 print "Average End-to-End Delay    = " n_to_n_delay * 1000 " ms";
 print "\n";
}
```