Article

On the Design of Efficient and Secure Hierarchic Architecture for Software Defined Vehicular Networks

Muhammad Adnan[1], Noor Ul Amin [1], Jawaid Iqbal [1], Abdul Waheed [1,2,*], Mahdi Zareei[3], Shidrokh Goudarzi [4], Asif Umar [1]

1 Department of Information Technology, Hazara University Mansehra, Mansehra 21120, Pakistan
2 School of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea
3 Tecnologico de Monterrey, School of Engineering and Sciences, Zapopan 45201, Mexico
4 Centre for Artificial Intelligent (CAIT), Universiti Kebangsaan Malaysia, 43600 Bangi, Malaysia
* Correspondence: abdul@netlab.snu.ac.kr, shidrokh@ukm.edu.my

Version November 22, 2020 submitted to Journal Not Specified

Abstract: Modern vehicles are equipped with various sensors, onboard units, and devices such as Application Unit (AU) that support routing and communication. In VANETs, traffic management, Quality of Service (QoS), and vulnerabilities are the main research dimensions to be considered while designing VANETs architectures. To cope with the issues of QoS and vulnerabilities faced by the VANETs, we design an efficient and secure SDN based architecture where we focus on QoS and security of VANETs. In this paper, QoS is achieved by a priority-based scheduling algorithm in which we prioritize traffic flow messages in safety queue and non-safety queue. In the safety queue, the messages are prioritized based on deadline and size using the New Deadline and Size of data method (NDS) with constrained location and deadline. In contrast, the non-safety queue is prioritized based on First Come First Serve (FCFS) method. Furthermore, it focuses on network vulnerabilities and addresses the identified threat vectors to secure the proposed Software Defined Vehicular Network (SDVN) architecture. In this architecture, we proposed a PKI-based digital signature scheme for the secure communication between Vehicle to Vehicle (V2V), public key authority infrastructure for Vehicle to Infrastructure (V2I), and a three-way handshake mechanism for the secure communication between main and sub-SDN controllers. For the simulation of our proposed scheduling algorithm, we used the CloudSim toolkit. The simulation results of safety messages show better performance than non-safety messages in terms of execution time. We validate our proposed security scheme using a new familiar simulation tool called AVISPA, which shows that our proposed security mechanisms for V2V, V2R, and V2I are secure.

Keywords: VANETs, QoS, SDVN, V2V and V2I Communications, AVISPA

1. Introduction

Recently, VANETs have got a great attraction in the research community. The researchers are developing protocols, applications, and simulation tools in different dimensions to make them smarter. In this connection, several architectures have been proposed but still facing some difficulties like less flexibility, less programmability, less scalability in the deployment of services in large-scale VANETs environment. Similarly, the network throughput problem becomes more sensitive when a large amount of information is simultaneously transferred between the hosts. The situation gets inferior when the network is congested with inefficient routing or bottlenecks. These issues create difficulty in the management of the network due to the dynamic behavior of the VANETs. Therefore, a new networking paradigm was introduced, known as Software Defined Networks (SDN). The basic idea behind SDN is the decoupling of the network control plane from the data plane. The data plan defines forwarding data while the control plane is responsible for controlling the entire network. The decoupling of the network control plane from the data plane provides a simpler programmable environment and provides external software opportunity to define a network's behavior.

The integration of SDN and VANETs can play a vital role in developing a new, improved VANETs architecture. With the in-depth study of literature review and comprehensive analysis of these two networking trends (VANETs and SDN), we move towards designing a new SDN-based VANETs architecture where the VANETs will be managed in a programmable and centralized way. SDN splits the data plane from the control plane, having centralized network controllers, which conclude how traffic flow will be forwarded within the entire network [1]. For the better performance of these two networking trends (VANETs and SDN), we believe that QoS in traffic management and its security are unavoidable and challenging concerns. There are several security issues and threat vectors in SDVN that may be victims of attacks on vulnerabilities. There may be a possibility of a man-in-the-middle attack in the first threat vector, and the second threat vector, there may be a possibility of existing forged or bogus traffic flows in the data plane. The third vector may be a victim of attacks on vulnerabilities in Road Side Units (RSUs). The third vector permits the attacker to cause disorder in the network by the weakness of forwarding devices. The most critical ones due to which the network operation can be compromised are threat vectors four and five. The attacker can easily control the network due to attacks on the control plane communication and SDN controllers due to attacks on controllers and some controllers' vulnerabilities. The last threat vector can cause due to the requirement of trusted resources for forensics and remediation, which can agree for investigations and exclude quick and secure recovery modes for carrying the network back into a safe operating condition.

1.1. Contributions

The main contributions of this paper are as follows;

1. In this paper, we have proposed a novel efficient, and secure architecture for SDVN to improve the QoS using a priority-based scheduling algorithm. We prioritize traffic flow messages both in safety and non-safety queues.
2. In the safety queue, the messages are prioritized based on deadline and size using the New Deadline and Size of data method (NDS) with constrained location and deadline.
3. In contrast, the non-safety queue is prioritized based on First Come First Serve (FCFS) algorithm.
4. Our proposed scheme highlights network vulnerabilities and addresses the identified threat vectors to design a novel efficient and secure hierarchic architecture for SDVN with efficient network resources utilization.
5. In our proposed novel and secure hierarchic architecture, we have improved the secure communication between vehicle to vehicle, vehicle to RSU, and vehicle to infrastructure using Public Key Infrastructure (PKI) based digital signature, and protected networks form adversary's attacks.
6. Additionally, we have used the concept of a three-way handshake mechanism to establish a reliable connection between main SDN and sub SDN controller for a secure key generation along with onward secure data dissemination.
7. We have used the CloudSim toolkit concept to simulate the proposed priority-based scheduling algorithm in hierarchic SDVN architecture.
8. We have proved the security of our proposed efficient and secure architecture using a familiar simulation tool called Automated Validation of Internet Security Protocols and Applications (AVISPA).
9. Moreover, we have validated our proposed architecture's fundamental security properties using a formal security method.

1.2. Paper Organization

The structure of this paper is categorized as follows. Section II consists of related work about VANETs and its traffic management, the background of SDN based VANETs, Priority-based scheduling, and SDN based VANETs security. Section III describes the issues and vulnerabilities in SDVN. Section IV consists of the proposed scheme, and priority-based scheduling algorithms are discussed in Section V and Section VI. The proposed scheme security analysis for SDVN describe in Section VI, where section VII discussed the simulation and evaluation. The last section VIII consists of a conclusion.

2. Related Work

Considering the QoS and security requirements in SDVN, we move towards an efficient and secure SDVN architecture. For this purpose, a comprehensive literature survey is presented, covering the VANETs background of SDN, SDN based VANETs, QoS factors in terms of traffic management and its security.

Recently, by the rapid development of wireless communication technology and the increased demand in the transportation field's information technology, the VANETs is an integral element of the Intelligent Transportation System (ITS). VANETs can equip hundreds or thousands of nodes in wireless communication. VANETs is a new type of Ad hoc network and is a particular part of its and is a subclass of Mobile Ad hoc Networks (MANET) with distinctive properties [2] like dynamic topology, limited bandwidth, limited energy, and many more. At the same time, the VANETs has some different characteristics such as mobility, dynamic topology, restricted geographical topology, the density of vehicle that is changeable concerning time, no constraints on network size, restrictions of road pattern, and so on. VANETs have three communication modes, which are V2V, Vehicles-to-Roadside (V2R) unit, and V2I. VANETs plays an essential role in safety as well as non-safety applications [3]. Driver drowsiness prevention system, emergency warning system, collision avoidance, automatic emergency braking system are included in the safety applications. On the other side, the traffic information systems like direction changer, cooperative entertainment, toll service, Internet access fall under the non-safety applications [2]. Significant applications of VANETs include road information dissemination that provides help to the driver as well as car safety based on sensor data, accident avoidance, regional weather forecast, information regarding the next available parking space, map location, driverless vehicles, fuel prices offered by the nearest station and many more [3]. To make possible these applications, different protocols have been deployed. The researchers are attracted to developing protocols, applications, and simulation tools in VANETs to improve efficiency and secure communication.

In [1], Kreutz *et al.* have pointed out that the SDN is superior to traditional networks due to some drawbacks. They do not have global information on the network, manual configuration, and high latency in path recovery. This new networking paradigm SDN is designed with a logical programmable central controller keeping global information. SDN decouples/separates the data plane from the control plane, having a logically centralized controller and global view of the entire network that decides how traffic flow will be handled within the network. With the *OpenFlow* protocol's help as southbound Application Programming Interface (API) and northbound API, the control plane's interaction is accomplished with the data plane and application plane correspondingly. In more detail [4], they say that centralized control enables rapid reconfiguration of the network, allocating network resources in dynamical ways, is more flexible, and makes troubleshooting more straightforward and more manageable. To overcome the challenging issues faced by vehicular communication, Yaqoob *et al.* [5] proposed a new networking paradigm with SDN's unique properties and benefits, called SDVN. They categorize the SDVN concept to create taxonomy-based vital characteristics. They identify and outline the key requirements for SDVNs and discuss several challenges that should be addressed to promote SDVN implementation.

Several architectures for SDVN, such as architecture with a central control host, selected server architecture with partial decentralization, and hierarchic architecture.

With time, new frameworks are developed to improve existing schemes. In [6], Sadio *et al*. proposed a topology-based routing protocol using SDN technology. This scheme consists of a routing algorithm through which path is selected, and flow tables are created based on path selection, which is accomplished with the help of the predicted topology. There are two models of communication that are unicast for data collecting and geocast for data dissemination. The performance analysis shows that the SDN is efficient than the other traditional routing protocols. Soufian *et al*. [7] worked on the architectural elements and placed dynamic controllers in SDVN. The author describes different approaches and proposes an architecture for dynamic controllers to the placement of controllers to readjust the controllers into road traffic situations adoptively. In a centralized SDN controller environment, there must be a burden on a controller due to continuous communication to the forwarding nodes, collecting information about the network state, and applying different forwarding rules and network policies. That is why they proposed dynamic controllers architecture for SDVN. The proposed dynamic controller's strategy is evaluated in a real traffic scenario and shows excellent results to reduce network changes. Lionel *et al*. [8] proposed a framework for SDVN based on Multi-access Edge Computing (MEC). This scheme consists of two algorithms: selecting the received information from neighboring in-vehicle messages from V2V and V2I communications. Moreover, the second one is implemented to *OpenFlow* protocol for the updation of flow tables for forwarding device. This architecture also comprises four logical layers through which it improves the path routing and reduces latency computation. Sadio *et al*. [9] proposed a prototype to design SDVN. In this scheme, an SDN environment based on the backbone is tested in real hardware that comprises *OpenFlow* switches. Then the SDN environment based on Radio Access is tested on a Wi-Fi access point comprised of *OpenFlow* switches and sustains click modular router. For better mobility management of V2V and V2I communications, routing algorithms for topology prediction are used on different SDN controllers. As a result, free bandwidth for routing is more suitable because it kept the flow balance through SDN switches.

In [10], Baihong *et al*. proposed SDN Based Vehicle Ad hoc On-Demand, Routing Protocol (SVAO). They compare SVAO with other ad hoc routing protocols such as Optimize Link State Routing (OLSR), Dynamic Source Routing (DSR), Destination Sequence Distance Vector (DSDV), and Distance Based (DB) routing protocol through simulation. Based on the packet reception and packet delay analysis, the SVAO performs better than the others in large-scale networks or high vehicle speeds. In [11], Balamurugan proposed a scheme for VANETs using SDN technology in which the deployment of SDN in VANETs and its importance are discussed. Software-defined VANETs lack the message priority, and it is essential to send a message on a priority basis. Thus, the authors have proposed an algorithm for message prioritization where messages are forwarded based on priority, such as emergency, low, and high priority messages. They have implemented the message prioritization inside the *OpenFlow* protocol, which can cause burden and delay. In [12], Ahmed *et al*. proposed an architecture based on SDN for infrastructure-less VANETs environments known as Unmanned Aerial Vehicle (UAV) assisted. UAVs are integrated to investigate unreachable affected zones and the management of rescue vehicles in case of emergencies. These authors examine a data processing policy that consists of computation offloading/sharing decision problems for better management. The main aim is to keep a balance between energy consumption and delay in terms of computation. A theoretical game approach is used to create offloading/sharing decision problems, and a distributed computation algorithm is designed to solve the problem.

In [13], Smita *et al*. proposed a scheduling algorithm for VANETs based on a priority-based RSA algorithm (p-RSA) using a dynamic cloud. A dynamic cloud is placed on the roadside unit's position for maintaining the quality of service to the users. This algorithm divides the services into different categories like emergency, least, urgent, and average. Hence, the highest priority is given to emergency service among all. The proposed scheduling algorithm is compared with other scheduling schemes, which consist of First Come First Serve (FCFS), new Deadline and Size of Data method (NDS), and Shortest job based on Data First (SDF), which shows better results in terms of less bandwidth consumption and less energy utilization on performing a maximum number of services. In [14], Zhang *et al*. proposed a scheduling scheme for accessing the data from the vehicle to RSU, based on both deadline and size, known as $(D * S)$ algorithm. If multiple requests ask for the same deadline among all of these requests in this algorithm, the smallest data size request will serve first. If multiple requests ask for the same data size among all of these requests, the smallest earlier deadline request should be served first. Furthermore, the author enhances the $(D * S)$ algorithm to $(D * S/N)$ schedule. Most pending data requests should be provided first, and multiple requests are served with a single wireless broadcasting mechanism. Furthermore, to provide a balance between uploading and downloading the request, a two-step scheduling scheme is introduced, showing better performance results. In [15], A. P. *et al*. proposed a scheduling strategy known as a collective scheduling algorithm. The messages' priority is achieved with three factors; the size of the message, static factor, and dynamic factor. This collective scheduling is used for clustering in VANETs. Static factors classify the safety and non-safety messages, and dynamic factors are calculated with clustering in VANETs. Based on the above three factors, the messages' priority is calculated, and these messages are rescheduled to service and control channels. The simulation result shows that this scheme is reliable. In [16], Zhu *et al*. proposed architecture for Hybrid Emergency Message Transmission (HEMT) based on SDN technology on the Internet of Vehicle (IoV) in which the emergency message is transmitted to those vehicles over the area where the coverage of RSU is not entirely accessible by proposing a mechanism known as Vehicle Multi-hop Broadcast Trigger (VMBT). Through this mechanism, real-time and coverage ratio performance is improved, and the reliable transmission of emergency messages occurs in V2V communication. The simulation result shows that the scheme is scalable, reduces the controller overhead, and improves the coverage ratio's emergency messages. There are several scheduling schemes presented in VANETs like RSU based cloud scheduling proposed by S. Singh *et al*. in [17], declared scheduling scheme for data, voice, video, and emergency based on its weight calculated by $(D * S/W)$ proposed by M. Asgari *et al*. in [18]. In VANETs, the vehicle changes its position frequently due to high mobility; for this reason, J. M. Y. Lim *et al*. [19] proposed a priority scheme where priority is given to high mobility vehicles based on prediction using the Markov model's principles. In [20], S. Mohammad Javad *et al*. have given a packet scheduling mechanism where priority is given based on the importance of the packets degrees by multi-level queuing. In [21], B. B. Dubey *et al*. proposed a scheduling policy for those in the range of RSU, and its deadline is near to expire. Moreover, it gives preference to those requests whose priority is high, but its deadline is low, due to which these messages are dropped.

Different security architectures are proposed for solving security issues in SDVN. In this connection, Harsha *et al*. [22] proposed a framework to secure the communication in software-defined VANETs by providing an identification mechanism for malicious vehicles in a dynamic environment using a trust-based concept. For the detection of malicious vehicles, they used two algorithms for providing double security checks. The first algorithm is used to identify a trusted vehicle, and the second algorithm is used to identify malicious vehicles. The system shows better results in terms of improving the throughput and reduces the delay. Maxim Kalinin *et al*. [23] suggested a Software-Defined Security (SDS) approach for VANETs based on SDN technology. It is a global security representation in which security is controlled, managed, and implemented by software. In SDS, security controls such as network segmentation, intrusion detection, and access control are automatically determined through a programmable structure that equips data control over the entire network. Here are four functional layers for SDS implementation: security software, security policy management, and orchestration, data layers, and virtualization. For SDS implementation, the author tried to achieve the best security, access control, and confidentiality in VANETs. Huijun Peng *et al*. [24] presented a method that finds the anomaly flows based on SDN to secure the SDN flows. The author gives an overview that provides the structure and the basic process flow to detect anomalies in SDN. This method classifies an optimization for anomaly detection with a proposed algorithm that improves the detection and accuracy rate of detecting anomaly and reduces the false positive rate in an SDN environment. S. M. Mousavi *et al*. [25] proposed entropy-based quick Distributed Denial of Service (DDoS) detection against SDN controllers. In this scheme, the controllers are protected by allowing the controllers' capabilities and calculating the entropy to receive grouping requests by controllers, which leads to the quick detection of identification anomalies, in [26] proposed an authentication scheme by introducing key insulation in VANETs to address security issues in different attacks on VANETs. Before signing the vehicle, it obtains its updated secret key with the help of TPD. First, the timestamp is checked whether it is valid or not, and then it matches the signature either correct or not. With this, vehicles gain forward, and backward secrecy also updates their secret keys periodically. With the in-depth study of the literature review and comprehensive analysis of these two networking trends (VANETs and SDN), we will move towards the SDN-based VANETs system. These two emerging technologies (VANETs and SDN) are still under consideration and development because of its features and real applications. To better perform these two networking trends (VANETs and SDN), we believe that security and QoS are the significant challenging concerns for moving towards the design of an efficient and secure SDVN architecture.

2.1. Issues and Vulnerabilities in SDVN

SDVN environment is at risk due to several threats and vulnerabilities. These vulnerabilities are divided into six threat vectors shown in Fig 1.

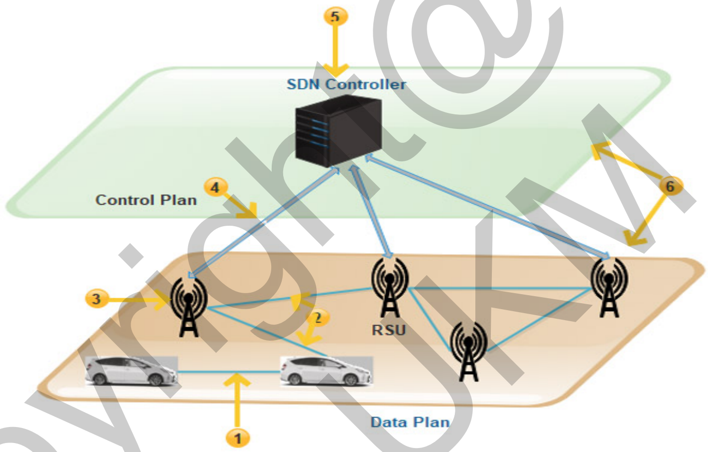


Figure 1. Issues and Threats Vectors in SDVN

These threat vectors found in SDVN may be a victim of attacks. In the first threat vector, there can be a possibility of a man-in-the-middle attack.

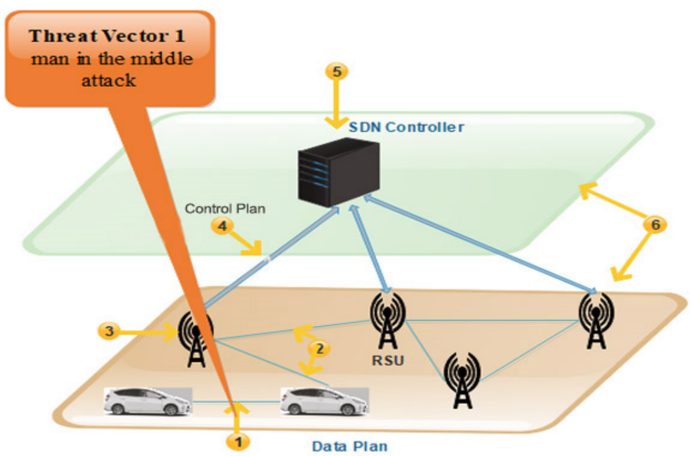


Figure 2. Threat Vector-1

The second threat vector may suffer from fake or invalid traffic flows in the data plane. The nodes can be injected with fake information that is communicated to forwarding devices [27].

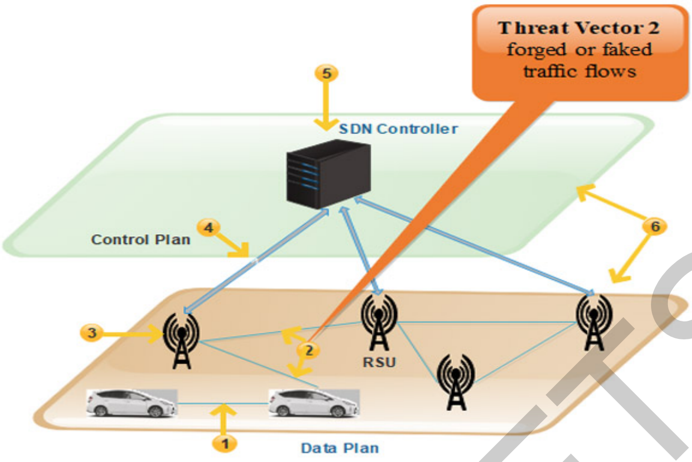


Figure 3. Threat Vector-2

The third vector may be a victim of attacks on vulnerabilities in RSUs. This weakness of the forwarding devices may allow the attacker to cause disorder in the network. The Denial of Service (DoS) attack is faced by the forwarding plane in the SDN system due to the repetitive requests in VANETs nodes. Nodes are the vehicles that have limited storage capacity. When packets are coming to nodes and nodes does not find the path for that packet, a query is sent to the RSU to ask the controller about the missing rule. When the node receives the rule, they take a decision consequently. There may be an opportunity for a DoS attack in which a large amount of data is sent from the attacker side [29].

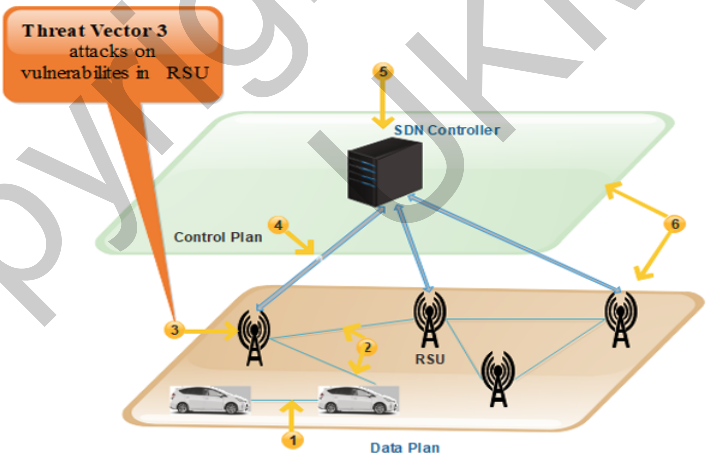


Figure 4. Threat Vector-3

Threat vectors four and five are the most critical ones due to which the network operation can be compromised. The attacker can easily control the network during handover on the control plane, and SDN controllers are also susceptible. When multiple vehicles in the network send packets simultaneously to one another, a Distributed Denial of Service (DDoS) attack can be caused in the control plane because all the rules are not available on the switch. So multiple queries are generated and sent to the controller, which causes a delay in the result of the dropping of queries [29].

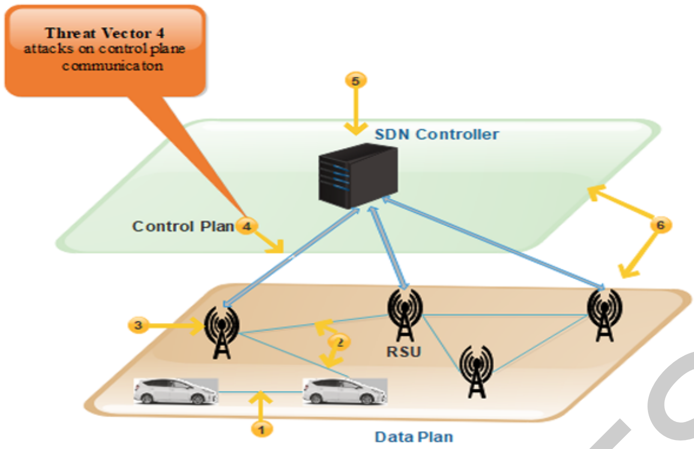


Figure 5. Threat Vector-4

The SDN controllers may be a victim of attacks due to vulnerabilities in controllers' physical error. Another one is the generation of a fake controller. The malicious user can perform the original controller's role known as identity spoofing, which sometimes forces the RSU to stop communication by dropping data [28]. In SDN, the entire network's overall functionality will be affected when a single point of failure occurs in the controller while communicating with another device in a centralized system [29].

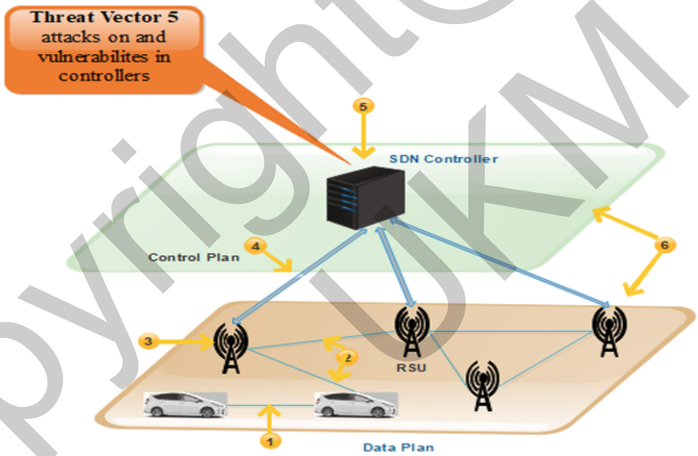


Figure 6. Threat Vector-5

The last threat vector identified between the control plane and data plane, but in this paper, we will address the security loopholes of threat vectors 1 to 5.

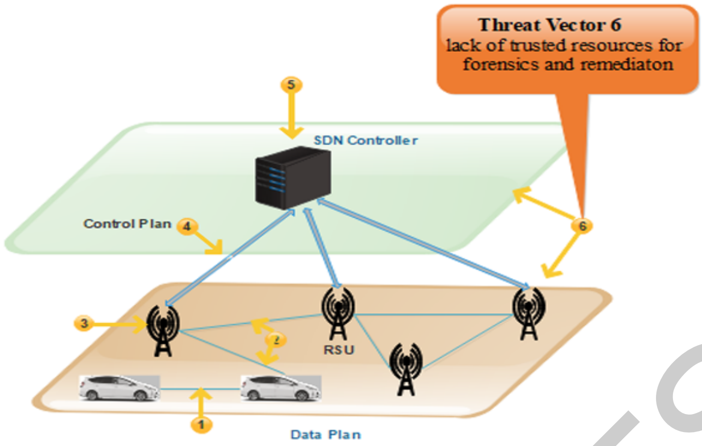


Figure 7. Threat Vector-6

3. Proposed Hierarchic Architecture for SDVN

With the deep study of literature review and comprehensive analysis of these two networking trends (VANETs and SDN), we will move towards the SDN-based VANETs architecture. These two emerging technology (VANETs and SDN) are still under consideration and development because of its feature and real applications. Therefore, it is important to design an efficient routing strategy for SDN-based VANETs architecture and security. To tackle this, we design an efficient and secure hierarchic architecture for SDVN. The network model and proposed routing strategy are discussed below.

3.1. NETWORK MODEL

In this scheme, the network model consists of the following components: the main SDN controller, sub SDN controller, BSs, RSUs, wireless switches, and vehicles. It is a hierarchic architecture, so the network's control plane consists of a central SDN controller at the top of its level. The lower level consists of sub SDN controllers, RSUs and BSs. The wireless switches and vehicles are present in the infrastructure layer. The following SDN components are needed for deploying the system:

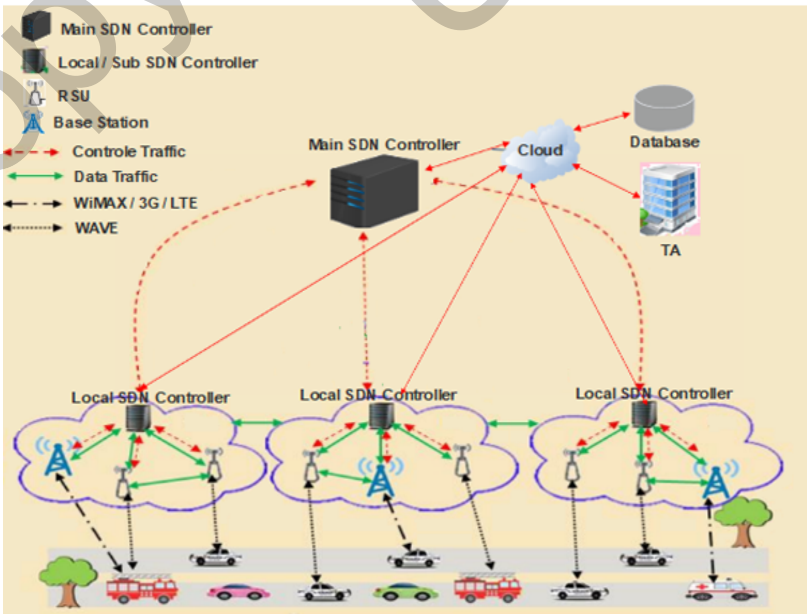


Figure 8. Proposed hierarchic architecture for SDVN

3.2. SDN Controller

The leading SDN controller builds a global view of the communication infrastructure and distributes its policy rules. Moreover, it divides the VANETs into zones of responsibility. The main SDN controller sends the global rules to each controller, which describes the network's general behavior and has a clear scope of the entire VANETs. The SDN controllers set the rules and identify the routing parameters concerning the launch of a specific protocol. The communication between the data plane and the control plane is done on *OpenFlow* protocol. In contrast, the communication between the SDN controllers and the cloud is performed through specific Application Programming Interfaces (APIs).

3.3. SDN Nodes

In VANETs, nodes are vehicles equipped with On-Board Units (OBUs), making the vehicles communicate with each other by sending information directly or through Road Side Units (RSUs) deployed on the road and operating on *OpenFlow* protocol.

3.4. SDN Road Side Unit

The RSU is a physical device that is permanently installed on the roadside. The RSU device is connected to the network to provide communication between vehicles and the SDN controller.

3.5. Trusted Authority (TA)

The responsibility of the TA includes the registration of vehicles. It authenticates all the users registered to the VANETs environment and manages the secret parameters like keys for all those users.

3.6. Database

A database stores information about the network, vehicles, and their owners.

3.7. SDN Cloud

The SDN controllers are connected to the cloud where different computations are performed, such as calculations of the car speed and distance, assessments of the road traffic situation, and perform services on a priority basis. The database is processed and managed through the cloud. The stored information in the database is updated continuously using a priority-based scheduling algorithm. The services are categorized on a priority basis for improving the QoS in VANETs.

4. Priority based Scheduling Algorithm

We will use a priority-based scheduling algorithm in which messages are divided into two categories, such as safety messages and non-safety messages. The safety messages consist of emergency messages, including hospital emergency, police helpline, rescue, natural disaster, etc. At the same time, the non-safety messages are related to user requirements such as the next traffic signal, nearest petrol pump, nearest airport, nearest shopping mall, and nearest restaurant, etc. The safety messages are the important messages associated with human life and usually constrained by location and time (for instance, the safety information is valuable only to measure the relative distance from its original location). In this way, we can include context information with the exact time and location. The safety messages have a smaller deadline, which indicates that the data is valuable or outdated. It will be discarded if the information is outdated; otherwise, it is forwarded through the application layer for immediate response. We use an NDS method, where the message with the smallest deadline and size will be assigned first in the scheduling queue. In contrast, non-safety messages are given to the output queue on an FCFS basis.

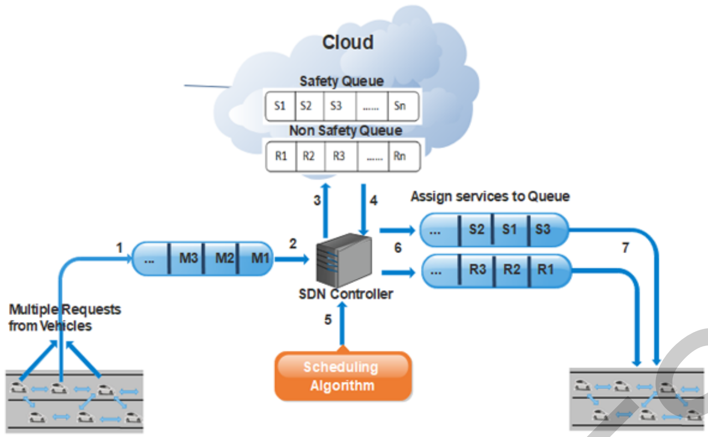


Figure 9. Services on priority based scheduling

Following are the steps for categorizing the services on priority based scheduling;

1. Multiple vehicles are sending requests for different services; these requests are stored in the queue.
2. Every request is forwarded one by one to the SDN controller.
3. The SDN controller is connected to the cloud where various computations are performed, such as services are categorized into the safety and non-safety messages, and then these messages are sent back to the SDN controller.
4. Scheduling algorithm assigns the priority to emergency messages based on deadline and size. The message having the least deadline and smallest length will be considered for higher priority among all services.
5. The services are forwarded to the output queue to the given priority, as shown in Fig.9.
6. The vehicles efficiently receive their services.
7. For non-safety messages, the requests are categorized based on FCFS.

In the following algorithm 1, the vehicles send a request for different services. These requests are placed in a queue. In this case, we say List (L1) is sending to SDN controller for further processing.

Algorithm 1: For Vehicles/Nodes request to cloud

Input: Request type
Output: List of request L1.

1. for $(i = 1; i <= n; i++)$
   // vehicle request $i = \{1, 2, 3, \cdots n\}$
2. $S = \{j1,\ j2,\ j3, \cdots jn\}$
   // S =Request Type (vehicle can send multiple requests such as nearest ATM, nearest petrol pump, natural disaster, police helpline, rescue, etc.
   // $i = 1 \cdots, n$ are vehicles.
3. L1= Add request of vehicle (i) // vehicle $(i) = S = \{S1, S2, S3 \cdots Sn\}$
4. Return L1
5. End of for

In the following algorithm 2, these services are categorized into safety, and non-safety messages and the two lists are prepared, i.e., List (L2) and (L3). The safety messages are placed in (L2), and the non-safety messages are placed in (L3).

In the following algorithm 3, the (L2) and (L3) are the lists of safety, and non-safety messages take as an input. Furthermore, for safety messages, the weight is calculated for each message based

Algorithm 2: Data categorization by cloud

Input: List of the request of vehicle L1
Output: L2 and L3 safety and non-safety list of requests.

1. for $(i = 1; i <=$ length of $L1; i++)$
2. if $(L1_i =$ ("ambulance", "hospital emergency", "police helpline", "rescue"))
3. Assign $L1_i = L_2$
4. else assign $L1_i = L_3$
   End if
5. Return L2 and L3
   End of for

on deadline and size. Get the length and deadline of a message and then find the average length and deadline of each message, sorted in ascending order. The average difference is calculated for each message based on deadline and size. The messages that have the smallest deadline and size will be assigned first in the scheduling queue. Moreover, for non-safety messages, the priority is given based on the FCFS scheduling algorithm.

Algorithm 3: Prioritization of Safety and Non-Safety List (i.e. $L2$ & $L3$)

Input: $L2$ and $L3$
Output: $L4$ and $L5$ lists i.e., prioritize the list of safety and non-safety messages are sent to vehicles

1. for $(i = 1; i <=$ length of $L2; i++)$
   $L4 = PS_i = D_i * S_i$
   $Q1 = PS_i$ // Q1 is the random list of L3.
2. for $(i = 1; i <= Q1.\text{length}; i++)$
   Find min $Q1_i$
   $L4 = \min Q1_i$ // Build list $L4$ from minimum to maximum
   End for
3. Non-Safety for (j=1; j<=length of L3; j++)
   $PNS_J = FCFS$
   $L5 = PNS_J$

In the above algorithm, the (PS_i) stands for the priority of safety messages, and (PNS_J) stands for the priority of non-safety messages.

5. Proposed Security Mechanism

To protect the critical information from adversaries attacks during transmission, we have proposed a novel security mechanism among V2V and V2I communications. Additionally, our proposed security mechanism consists of secure the communication between vehicles to vehicles, secure the communication between vehicles and RSUs, and secure the communication between the sub and main SDN controller.

5.1. Secure Communication between Vehicles to Vehicles

We use Public Key Infrastructure (PKI) based digital signature scheme to secure the communication between vehicles. Before starting this concept, an overview of PKI and digital signature are presented;

5.2. Secure Digital Signature

The digital signature is a mathematical process of protecting the document from unauthorized users. It ensures that the digitally transferred data is authentic and validates that the document sent has no changes.

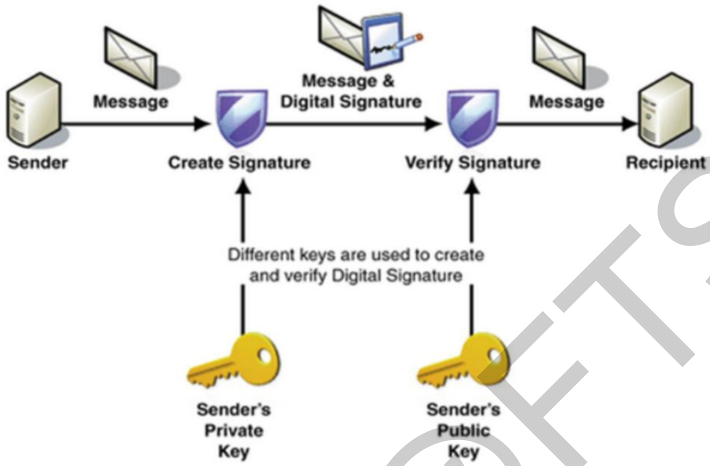


Figure 10. Working flow of digital signature [30]

Moreover, a digital certificate is signed and provided by the Certification Authority (CA) to guarantee trust in the signed data.

5.3. Signing and Verification process of Digital signature

The following are the process of signing a digital signature.

1. First, the generation of hash value using hash function and algorithm.
2. The encryption is done by the sender's private key on the generated hash value. This encrypted hash value is known as a digital signature.
3. The original data and signature are then sent to the receiver.

The following are the steps to process digital signature verification.

The decryption is done by the sender's public key to get the hash value.

(a) Take the hash value for the original data.
(b) Then these hash values are compared;
(c) If these hash values are matched with each other, we say that the received data is not changed but has its original form.

If these hash values do not match each other, we say that the received data is changed and does not remain in its original form. After that, the data is sent by the sender, as shown in Fig.11.

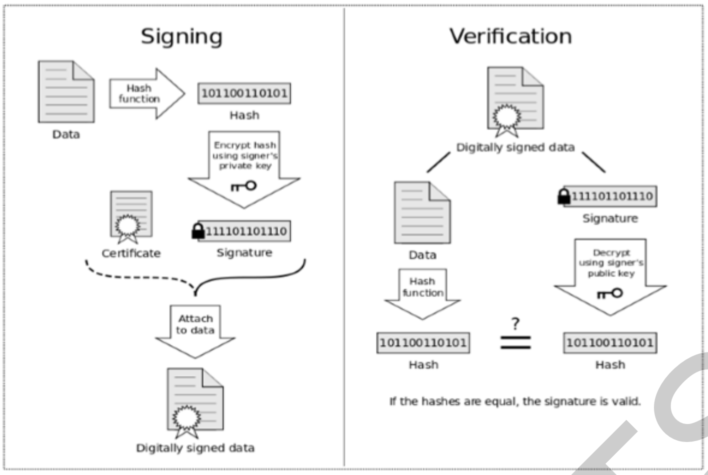


Figure 11. Signing and Verification process of Digital signature [30]

In a digital signature, we achieved the authentication and integrity of sensitive data. Initially, we have defined global public key components for the generation of user private key. In the user private key, we select a random number x belongs to (q). Moreover, we calculate the user public key, where (g) is a generator and (x) is the selected random number belongs to mod (p). We kept secret the security number (t) preserved the data's privacy, while we use a signature algorithm for verification of sender data on the receiver side. Furthermore, in the verification step, we authenticate the identity of the received data and claimed sender. The below algorithm explained the proposed digital signature process in detail.

Algorithm 4: Digital Signature Algorithm

Input:- Signing
Output:- Verification
Steps:-

1. **Global Public Key Components**

   P: Prime number $2^{L-1} < P < 2^L$
2. **User Private Key**

   x: Random number

   Where $0 < x < q$
3. **User Public Key**

   y : Random number

   $Where\ \ g^x \text{mod}\ p$
4. **Secret Number**

   k: any integer number

   Where $0 < k < q$
5. **Signature**

   $r = (g^k \bmod p) \bmod q$

   $s = [k^{-1}(H(M) + x \cdot r)] \bmod q$
6. **Verifying**

   $v = [(g^{u1}\ y^{u2}) \bmod p] \bmod q$

   $u1 = [H(M')w] \bmod q$

   $w = (s')^{-1} \bmod q$

   $u2 = [(r')w] \bmod q$

   $V = r'$

5.4. PKI Based Digital Signature Scheme

Whenever a vehicle wants to communicate with another vehicle, the following steps are required, as shown in Fig.12.

1. The sender sends a request to the Registration Authority (RA) with his public key for issuing the certificate.
2. The RA verifies the sender's request and forwards it to the Certificate Authority (CA).
3. The CA issues the certificate with his public key, stores this certificate to the repository, and sends a copy to Validation Authority (VA).
4. Then, this certificate is back sent to the sender.
5. After that, the sender sends this certificate along with a digital signature to the receiver.
6. When a recipient receives this certificate, it is further sent to the VA to check the certificate's validity. The VA checks three things, first, checks that the certificate is valid; if the certificate is valid, then it sends a message to a receiver that the certificate is valid; second, in case of the invalid certificate, the receiver will not regard the message; third, if the sender has no certificate validity at all the receiver considers that this is the malicious user.
7. After checking the validity, the VA sends it back to the receiver.

After the above process, secure communication will be established between V2V.

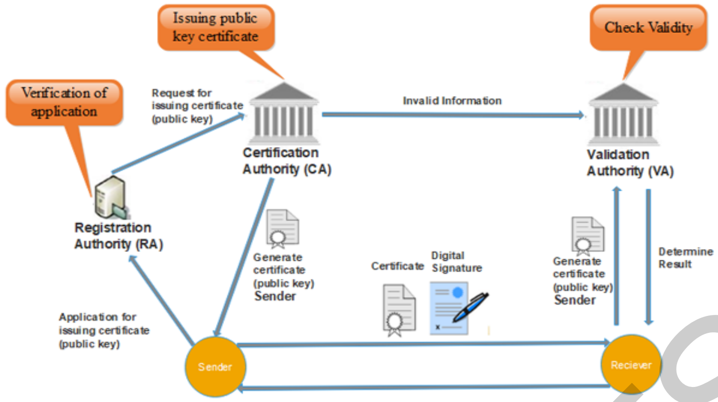


Figure 12. PKI based digital signature scheme for secure V2V communication

5.5. Secure Communication between Vehicles and RSU

The public key authority provides the essential security for public key distribution that maintains an active directory of the public key for all members. The following process occurs, as shown in Fig.13.

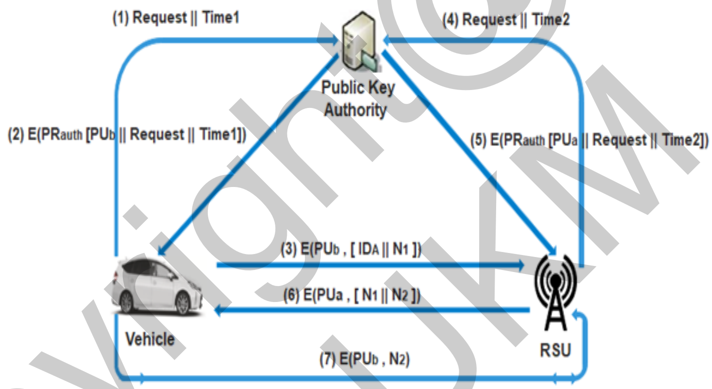


Figure 13. Public key authority infrastructure for secure communication b/w vehicle and RSU

1. The vehicle sends a message to a public directory that contains a request and timestamp for the current public key of RSU.
2. The public key authority responds to a vehicle message that is encrypted with the private key of the authority (PR-auth). The decryption of the message is done using the public key of the authority by the vehicle.
3. The message includes the public key of RSU, the original request, and the original timestamp.
4. The vehicle stores the public key of RSU. For encrypting the message, an identifier of the vehicle (IDA) and a nonce (N1) are used for unique identification.
5. The RSU sends a message to a public directory containing a request and timestamp for its current public key.
6. As usual, the public key authority responds to the RSU message and retrieves the vehicle's public key. In this way, the public keys have been securely delivered to the vehicle and RSU to protect an intruder's communication.
7. When the RSU sending a message to the vehicle using the public key of the vehicle (PUa) with a nonce (N1), and RSU generates a new nonce (N2), to assure that this vehicle and RSU are correspondents to each other.
8. With the help of the public key of RSU, the vehicle encrypts the message and returns nonce (N2) to RSU to ensure the exact correspondent.

So, in this case, seven messages are required for secure communication between the vehicle and RSU.

5.6. Secure Communication between Main SDN Controller and Sub SDN Controller

Whenever controllers are required to communicate with each other, the following steps are needed before starting the secure communication;

1. Any controller has its master keys like a master public key (M_{PUK}) and master private key (M_{PRK}).
2. Master public keys of both are exchanged publically.
3. The sub SDN controller sends a message to the main SDN controller that contains ID_{Sub}, a nonce (N), and a timestamp that is encrypted with the public key of the main SDN controller.
4. The main SDN controller decrypts the message with his private key, gaining the original message, and responding sub SDN controller message that includes ID_{Main}, timestamp, and adds one nonce $(N+1)$ and is encrypted using the public key of sub SDN controller.
5. The sub SDN controller decrypts the message using his private key to gain the original message that contains ID_{Main}, timestamp, and nonce plus one $(N+1)$.
6. So the main and sub SDN controllers have one nonce (N) and nonce plus one $(N+1)$. They perform XOR operation on nonce values to produce a secret session key after establishing a secure connection.

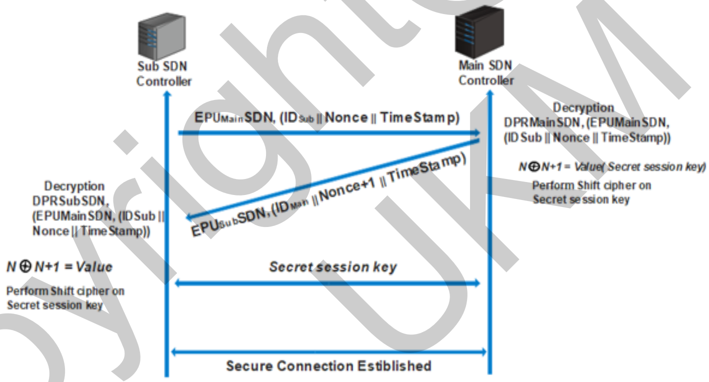


Figure 14. Three way hands shake mechanism for secure communication between sub and main SDN controller

6. Formal Proof

Theorem 1. *Using theorem (1), we proved the confidentiality of our proposed scheme against adversary attacks, i.e., IND-CCA2.*

Proof. We used the Polynomial probabilistic algorithm against (IND-CCA2) in the random oracle model to satisfy our proposed scheme's confidentiality. Using the DDHP assumption, we showed how Challenger ($\mathbb{C}$) attacks a secure channel to tamper the sensitive information transfer from the vehicle to RSU.

Initial:- Challenger ($\mathbb{C}$) runs the setup algorithm using PKI based digital signature to get the system parameters and compute the secure key for decryption.

1. $R = g^k \bmod q$
2. S = $[K^{-1}(H(M) + (x.r))] \bmod q$

   Where $k0 < k < q$

3. $X =$ Random number where $0 < x < q$

Phase.1:- Challenger ($\mathbb{C}$) keeps secret the key 'k' and assume the key parameters to find the prime divisor of $(P - 1)$

where $g = h^{(p-1)/q}$ mod p

where h is any integer lies $1 < h < P - 1$

Now public key of the vehicle and other system parameters are transferred to the RSU to secure communication using the secret key.

Attacker: Initially, the attacker ($\mathbb{A}$) performs the DDHP queries to get the random users x.

Where $0 < x < q$

If an attacker gets a valid random number, it will compute the private key; otherwise, the attacker cannot temper the secure communication between the vehicle and RSU. It is a computationally hard problem for adversaries to get the valid random number x.

Phase.2: Attacker ($\mathbb{A}$) used the queries of Phase.1 as input and computed the session key using DDHP assumptions.

$S = \sum_{i=1}^{n} n \bigoplus n + 1$

Now perform shift cipher on compute session key (S). Furthermore, we define events, i.e., $e_1,\ e_2, e_3, e_4$.

e_1: Attacker does not execute the session key query using random number x.

e_2 : $Challenger(\mathbb{C})$ does not abort the PKI based digital signature queries.

e_3: Attacker ($\mathbb{A}$) Choose the RSU identity during the challenge phase.

e_4 : $Attacker(\mathbb{A})$ can guess the PU_a and PU_a using system parameter from public key authority.

Now Session key $(S) = (1 - T)^{qk}$, $S[e_2||e_1\ ] = (1 - T)^{qk}$,

$S[e_3||e_1||e_2\ ] \geq T$, and $S[e_4||e_1||e_2||e_3\ ] \geq \varepsilon$

So $S[e_1 \Lambda\ e_2 \Lambda\ e_3 \Lambda\ e_4\ ] \geq\ T\ (1 - T)^{qk+qu}\varepsilon$

Now solving DDHP instance $T \leq t + O\ (q_u)T_n + O(2q_{H1} + 2q_k)tm$

□

Theorem 2. *In our proposed scheme using theorem (2) we proved Unforgeability i.e., (EUF-CMA)*

Proof. We used a polynomial-time probabilistic algorithm against (EUF-CMA) in the random oracle model to satisfy our proposed scheme's unforgeability property.

Using CDHP assumptions, we proved that Forger ($\mathbb{F}$) used the non-negligible feature ε to forge the PKI based digital signature between vehicle and RSU for secure distribution of public-key certificate.

$\varepsilon^{'} \leq \varepsilon T(1 - T)^{qk+n-1}$

$T \leq t + O(2q_{h1} + q_k + 3q_s + n + 1)T_m\ + O(q_s)tp$

Where $h_i(i = 1, 2, 3, \ldots n + 1)$

Initial:- Challenger ($\mathbb{C}$) run the setup algorithm using PKI based digital signature in time (T).

Challenger ($\mathbb{C}$) applies the CDHP (P, aP, bP) queries to proved unforgeability.

Phase.1:- Challenger ($\mathbb{C}$) keeps the private key of the signer to protect the vehicle's data using the digital signature algorithm.

Challenger ($\mathbb{C}$) performs the setup algorithm along with other system parameters.

$PU_a = g^x$ mode P

Where x is a random number chosen by vehicle during the key generation process

$K =$ integer number

Where $0 < k < q$

1. $r = (\ g^k \text{ mod } p) \text{ mod } q$
2. $s = [\ k^{-1}(h(M) + x.r)]\ mod\ q$
3. $\ y^{u2})$ mod $p]$mod q
4. $u1 = [h(M^{'})w]$ mod q
5. $w = (s^{'})^{-1}$ mod q

6. $u2 = [(r^{'})w] \bmod q$
7. $V == r^{'}$

Attacker ($\mathbb{A}$) randomly select $x \ \varepsilon \ Z^{*}p$ and compute $Pr_a = g^{-x}d \bmod P$ and returns session key $(S) = (1-T)^{qk}$

Forgery ($\mathbb{F}$) used the CDHP assumptions to execute the private key for the tempering of the digitally signed document of the vehicle.

If $x^{'} = x$ accepted otherwise rejected ($\perp$)

For all $1 \leq i \leq m$, and $\mathbb{C}$ wants to get the system tuples $\{x,\ PU_a,\ PU_b, Pr_a\}$ from list and generates the following equations.

$e(\ h_1, PU_a, S) = e(S^{*}, P)\ e(\sum_{i=1}^{n}\ h^{*}i, PU_a - P_{Pub}\ )$

$e(\ h_1^{*}, Pr_a,\ S) = e(S^{*}, P)\ e(\sum_{i=1}^{n}\ h^{*}i,\ x^{*}i,\ PU_a - P_{Pub})$

Now Challenger ($\mathbb{C}$) execute

$S = (\ \ h_1^{*})^{-1}(Pr_a - \ \sum_{i=1}^{n}\ h^{*}i,\ x^{*}i,\ PU_b\ )$

Furthermore, we will calculate the probability of ($\mathbb{C}$) success using the following events.

$e_1 :$ $\mathbb{C}$ does not execute the CDHP queries for session key generation.

$e_2 :$ ($\mathbb{F}$) execute a correct and non-trivial encoded text of vehicle.

$e_3 :$ e_2 happens, and $x_i = 0 < x < q$

If the above events happened, so ($\mathbb{C}$) successful otherwise fails.

Session key $(S) = (1-T)^{qk}\ \geq (1-T)^{qk}$

$S[e_3||e_1\ ] \geq \varepsilon$

$S[e_3||e_1 \land e_2\ ] \geq T(1-T)^{n-1}$

So that $S[e_1 \land e_2 \land e_3\ ] \geq\ (1-T)^{qk}$

$\varepsilon T\ (1-T)^{n-1} = \varepsilon T\ (1-T)^{qk+n-1}$

Hence we proved that our proposed scheme satisfied both the security properties of confidentiality and unforgeability using theorem 1 and 2. □

7. Evaluation and Experiments

In this section, we present the proposed model simulation setup and evaluation of the model. We use the CloudSim toolkit to simulate the proposed priority-based scheduling algorithms and AVISPA to check our proposed security model's security mechanism.

7.1. Simulation Setup

The CloudSim [31] toolkit has been used to simulate the proposed priority-based scheduling algorithm. This framework is used for modeling and simulation of cloud computing services. There are two types of scheduling queues, such as safety and non-safety. In the safety queue, every message is scheduled based on length and deadline. The message that has the smallest deadline and size will be assigned first in the scheduling queue. For the non-safety queue, the messages are processed based on the FCFS method.

7.2. Experimental Evaluation

We created a data center; having a processing rate is 1000 Million Instructions Per Second (MIPS) and memory is 512 MB. In the first step, we got the length and deadline of a cloudlet and then found the average length and deadline of each cloudlet, which are sorted in ascending order in the lists. The average difference is calculated for each cloudlet based on deadline and size, and the cloudlets that have the smallest deadline and size assigned first in the scheduling queue. For non-safety messages, the priority is given based on the FCFS scheduling algorithm.

Table 1. Configuration of Simulated Cloud.

Cloud	Number
No. of Datacenter	1
No. of Cloudlet	40
No. of Broker	1
No. of Virtual Machines	1

Table 2. Configuration of Data center.

Data center	Configuration
Architecture	*x86*
RAM (MB)	512
Hypervisor	*Xen*
Storage (MB)	10000
MIPS	1000
Bandwidth (MBps)	1000

7.3. Simulation Result

In this section, each task's total execution time is calculated in the cloud by adopting the scheduling policy based on deadline and size. Table 3 shows the expected calculated execution time for safety messages based on the sum of the start and running time. Figure 18 and Table 4 show the expected calculated execution time for non-safety messages. Figure 19 shows the comparison of safety and non-safety messages in terms of the computed execution time, which shows better results than non-safety messages.

In Table 3, we calculate the result of 10 cloudlets based on the sum of start and running time and the average result is calculated for 10 cloudlets and then 20, 30, and 40 cloudlets as well for safety messages.

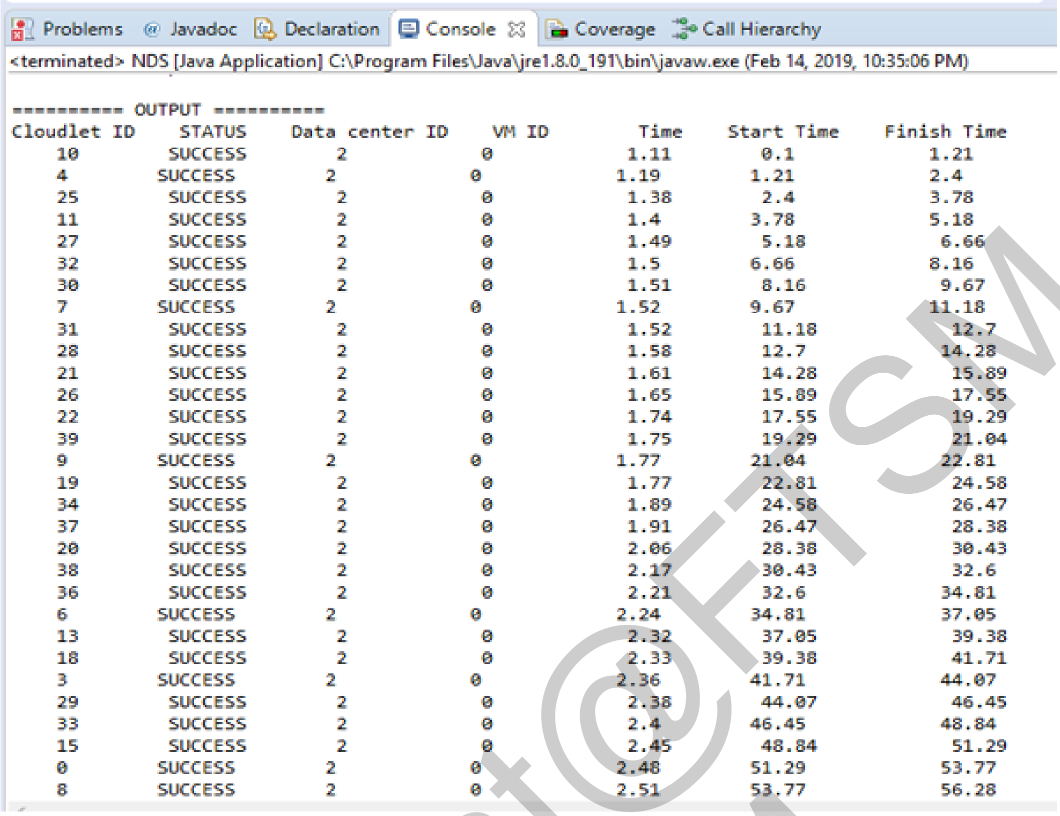
Table 3. Total calculated execution time for safety messages.

No of Cloudlets	Run time	Start time	Finish time
10	960.22	3613.79	4574.01
20	2641.34	16444.33	18085.74
30	2283.45	35747.62	38031.07
40	2925.7	110361.11	112962.04

In Table.4 we calculate the result of 10 cloudlets based on the sum of start and running time and the average result is calculated for 10 cloudlets and then 20, 30, and 40 cloudlets as well for non safety messages.

Table 4. Total calculated execution time for non safety messages.

No of Cloudlets	Run-time	Start time	Finish time
10	13.47	58.06	71.45
20	17.729	194.38	230.79
30	22.9	424.49	448.248
40	28.53	687.26	771.99s



Problems @ Javadoc Declaration Console Coverage Call Hierarchy

<terminated> NDS [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (Feb 14, 2019, 10:35:06 PM)

========== OUTPUT ==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
10	SUCCESS	2	0	1.11	0.1	1.21
4	SUCCESS	2	0	1.19	1.21	2.4
25	SUCCESS	2	0	1.38	2.4	3.78
11	SUCCESS	2	0	1.4	3.78	5.18
27	SUCCESS	2	0	1.49	5.18	6.66
32	SUCCESS	2	0	1.5	6.66	8.16
30	SUCCESS	2	0	1.51	8.16	9.67
7	SUCCESS	2	0	1.52	9.67	11.18
31	SUCCESS	2	0	1.52	11.18	12.7
28	SUCCESS	2	0	1.58	12.7	14.28
21	SUCCESS	2	0	1.61	14.28	15.89
26	SUCCESS	2	0	1.65	15.89	17.55
22	SUCCESS	2	0	1.74	17.55	19.29
39	SUCCESS	2	0	1.75	19.29	21.04
9	SUCCESS	2	0	1.77	21.04	22.81
19	SUCCESS	2	0	1.77	22.81	24.58
34	SUCCESS	2	0	1.89	24.58	26.47
37	SUCCESS	2	0	1.91	26.47	28.38
20	SUCCESS	2	0	2.06	28.38	30.43
38	SUCCESS	2	0	2.17	30.43	32.6
36	SUCCESS	2	0	2.21	32.6	34.81
6	SUCCESS	2	0	2.24	34.81	37.05
13	SUCCESS	2	0	2.32	37.05	39.38
18	SUCCESS	2	0	2.33	39.38	41.71
3	SUCCESS	2	0	2.36	41.71	44.07
29	SUCCESS	2	0	2.38	44.07	46.45
33	SUCCESS	2	0	2.4	46.45	48.84
15	SUCCESS	2	0	2.45	48.84	51.29
0	SUCCESS	2	0	2.48	51.29	53.77
8	SUCCESS	2	0	2.51	53.77	56.28

Figure 15. Experimental results in term of execution time for scheduling safety messages

Figure 16 shows the expected calculated execution time for safety messages based on the sum of start and running time for 10, 20, 30, and 40 cloudlets.

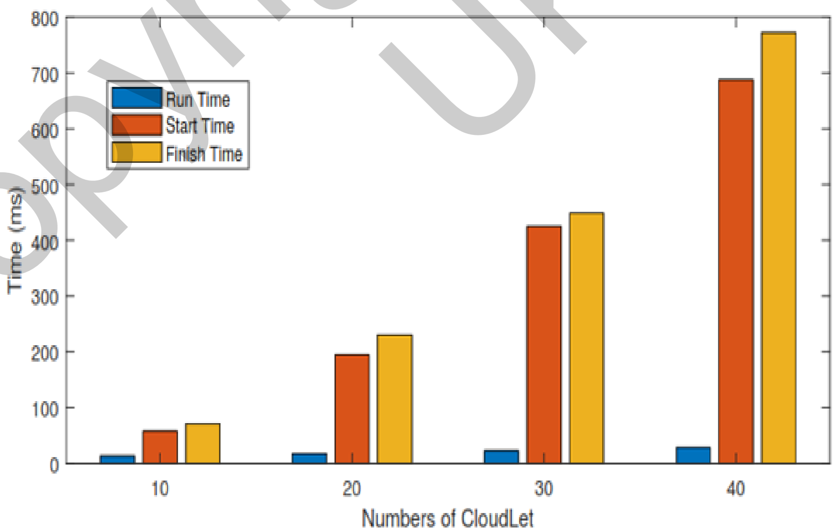
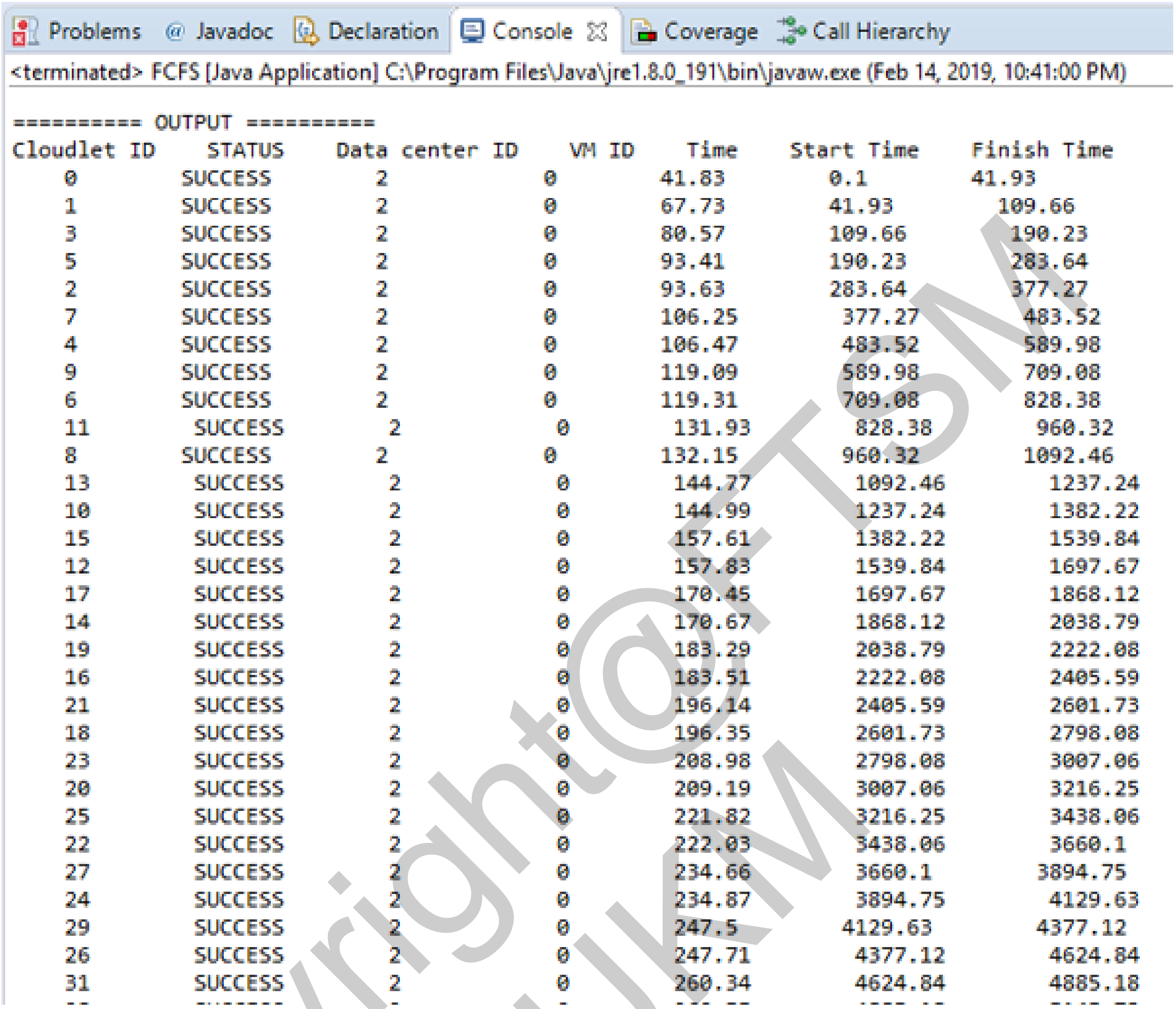


Figure 16. The life cycle of safety messages

In this section, the experimental result is carried out for 40 messages, and the execution time of each task is calculated in the cloud by adopting an FCFS basis. Figure 17 shows the experimental result in term of execution time for scheduling non-safety messages based on FCFS.



========== OUTPUT ==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	41.83	0.1	41.93
1	SUCCESS	2	0	67.73	41.93	109.66
3	SUCCESS	2	0	80.57	109.66	190.23
5	SUCCESS	2	0	93.41	190.23	283.64
2	SUCCESS	2	0	93.63	283.64	377.27
7	SUCCESS	2	0	106.25	377.27	483.52
4	SUCCESS	2	0	106.47	483.52	589.98
9	SUCCESS	2	0	119.09	589.98	709.08
6	SUCCESS	2	0	119.31	709.08	828.38
11	SUCCESS	2	0	131.93	828.38	960.32
8	SUCCESS	2	0	132.15	960.32	1092.46
13	SUCCESS	2	0	144.77	1092.46	1237.24
10	SUCCESS	2	0	144.99	1237.24	1382.22
15	SUCCESS	2	0	157.61	1382.22	1539.84
12	SUCCESS	2	0	157.83	1539.84	1697.67
17	SUCCESS	2	0	170.45	1697.67	1868.12
14	SUCCESS	2	0	170.67	1868.12	2038.79
19	SUCCESS	2	0	183.29	2038.79	2222.08
16	SUCCESS	2	0	183.51	2222.08	2405.59
21	SUCCESS	2	0	196.14	2405.59	2601.73
18	SUCCESS	2	0	196.35	2601.73	2798.08
23	SUCCESS	2	0	208.98	2798.08	3007.06
20	SUCCESS	2	0	209.19	3007.06	3216.25
25	SUCCESS	2	0	221.82	3216.25	3438.06
22	SUCCESS	2	0	222.03	3438.06	3660.1
27	SUCCESS	2	0	234.66	3660.1	3894.75
24	SUCCESS	2	0	234.87	3894.75	4129.63
29	SUCCESS	2	0	247.5	4129.63	4377.12
26	SUCCESS	2	0	247.71	4377.12	4624.84
31	SUCCESS	2	0	260.34	4624.84	4885.18

Figure 17. Experimental result in term of execution time for scheduling non safety messages

Figure 18 shows the expected calculated execution time for non-safety messages based on the sum of start and running time for 10, 20, 30, and 40 cloudlets.

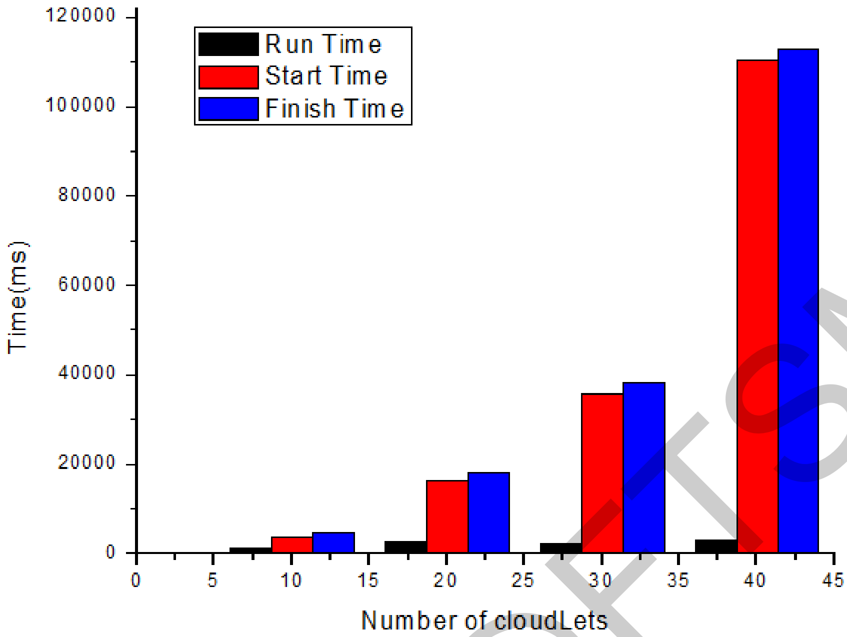


Figure 18. Life Cycle of non-safety messages

The above Figure 19 shows the comparison of calculated execution time for safety messages and non-safety messages. The calculated execution time of 40 messages is carried out for safety and non-safety messages and we see that the safety messages are executed in less time as compared to non-safety messages.

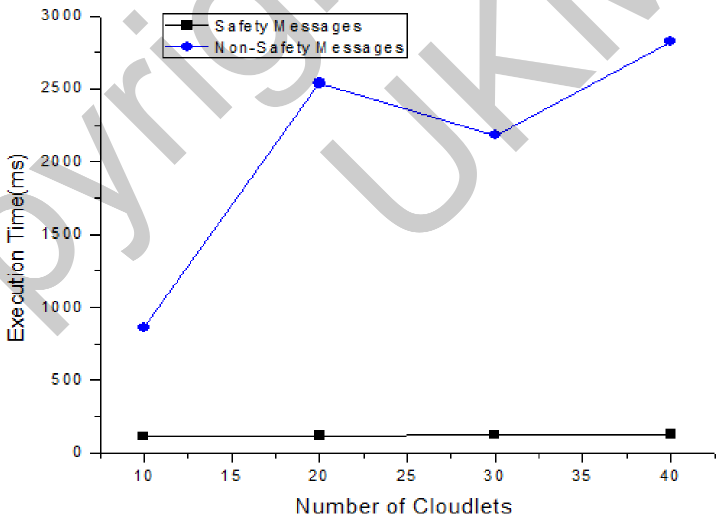


Figure 19. Comparison of calculated execution time for safety messages and non safety messages

7.4. Security Analysis

The second section of this paper contributes to secure the identified threat vectors and their vulnerabilities. We validate our proposed security scheme by using a familiar simulation tool called AVISPA [32,33]. In AVISPA, the user can interact with the help of a tool to identify the security problems to validate/verify and check the internet's sensitive security module and different cryptography techniques. This makes sure that the proposed security module or protocol is SAFE or UNSAFE by coding it into the High-Level Protocol Specification Language (HLPSL), which is then converted into machine language with the help of intermediate format (IF). There are four back ends modules, such as On-the-Fly Model-Checker (OFMC), Constraints Logic-based Attack Searcher (CL-AtSe), TA4SP protocol analyzer, and SAT-based Model-Checker (SATMC) to calculate and identify the results.

To secure the proposed SDVN architecture, we proposed a PKI-based digital signature scheme for the secure communication between V2V, public-key authority infrastructure used for V2I, and a three-way handshake mechanism to secure communication between main and sub SDN controllers. The proposed security scheme Secure Session Communication between V2V (SSCV2V) is validated with AVISPA, and Figure 21 ensures that V2V and V2I are SAFE as well as achieve confidentiality, integrity, and non-repudiation property. For the secure communication between the sub and main SDN controllers (SCSMC) scheme, Figure 22 shows the simulation results, which is SAFE.

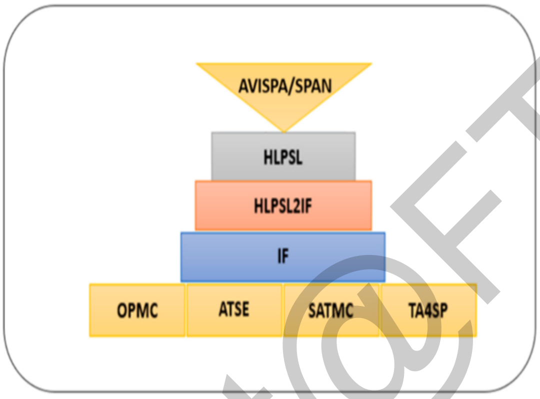


Figure 20. AVISPA Tool Architecture [31]

Figure 21. SSCV2V Simulation Result-1

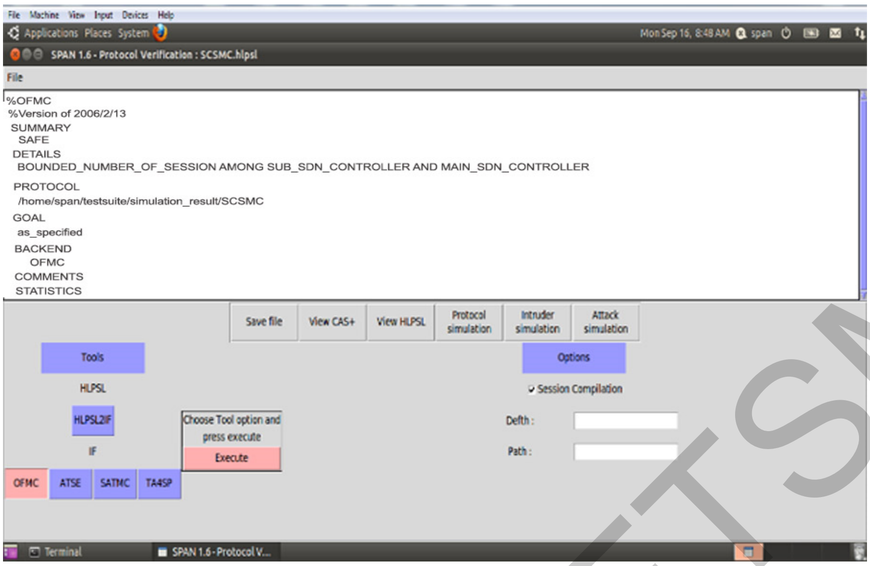


Figure 22. SCSMC Simulation Result-2

8. Conclusion

Quality of Service and security are the main research concerns in designing our proposed SDVN architecture. QoS in traffic management is achieved by priority based Scheduling Algorithm (PSA), where messages are categorized into two queues, i.e., safety queue and non-safety queue. In the safety queue, the messages are prioritized based on deadline and size using NDS as the safety messages are human life critical and constrained by location and deadline. In contrast, the non-safety queue is prioritized based on the FCFS method. We have used the CloudSim toolkit to simulate the proposed PSA. The simulation result of PSA shows better results than non-safety messages in terms of execution time. Moreover, we have focused on the vulnerabilities of the proposed SDVN architecture by addressing the identified threat vectors. We have used a PKI-based digital signature scheme to secure communication between V2V, public-key authority infrastructure for V2I, and a three-way handshake mechanism for the secure communication between main and sub SDN controllers. We have validated our proposed security model using the AVISPA simulation tool that ensures our architecture is secure and provides basic security properties such as confidentiality, non-repudiation, integrity, and unforgeability. Similarly, we have provided formal security proof to show that our scheme is secure.

Future Work

It is possible to provide an appropriate mechanism for the last threat vector that can cause due to the requirement of trusted resources for forensics and remediation, which can agree for investigations and exclude quick and secure recovery modes for carrying the network back into a safe operating condition.

Author Contributions: Formal analysis, Muhammad Adnan and Asif Umer; Funding acquisition, Mahdi Zareei and Shidrokh Goudarzi; Investigation, Jawaid Iqbal and Mahdi Zareei; Methodology, Muhammad Adnan; Project administration, Abdul Waheed and Shidrokh Goudarzi; Resources, Abdul Waheed and Shidrokh Goudarzi; Software, Muhammad Adnan, Jawaid Iqbal and Asif Umer; Supervision, Noor Ul Amin and Jawaid Iqbal; Validation, Noor Ul Amin; Writing – original draft, Muhammad Adnan; Writing – review & editing, Abdul Waheed.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kreutz, Diego, *et al*. "Software-defined networking: A comprehensive survey." *Proceedings of the IEEE* 103.1 (2015): 14-76.
2. Singh, Surmukh, and Sunil Agrawal. "VANET routing protocols: Issues and challenges." *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*. IEEE (2014): 1-5.

3. Pankaj Kumar, Chakshu Goel, Inderjeet Singh Gill, "Performance Evaluation of Network Aggregation Techniques in VANET," International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering, 5.1, January(2017).
4. Gheorghe, Gabriela, *et al*. "SDN-RADAR: Network troubleshooting combining user experience and SDN capabilities." *Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft)*. IEEE (2015): 1-5.
5. Yaqoob, Ibrar, *et al*. "Overcoming the key challenges to establishing vehicular communication: Is SDN the answer?." *IEEE Communications Magazine* 55.7 (2017): 128-134.
6. Kalinin, Maxim O., V. M. Krundyshev, and P. V. Semianov. "Architectures for building secure vehicular networks based on SDN technology." *Automatic Control and Computer Sciences* 51.8 (2017): 907-914.
7. Toufga, Soufian, *et al*. "Towards Dynamic Controller Placement in Software Defined Vehicular Networks." Sensors 20.6 (2020): 1701.
8. Nkenyereye, Lionel, *et al*. "Software defined network-based multi-access edge framework for vehicular networks." IEEE Access 8 (2019): 4220-4234.
9. Sadio, Ousmane, Ibrahima Ngom, and Claude Lishou. "Design and Prototyping of a Software Defined Vehicular Networking." IEEE Transactions on Vehicular Technology 69.1 (2019): 842-850.
10. Yaqoob, Ibrar, *et al*. "Overcoming the key challenges to establishing vehicular communication: Is SDN the answer" *IEEE Communications Magazine* 55.7 (2017): 128-134.
11. B. V, "An Intelligent Framework for Vehicular Ad-hoc Networks using SDN Architecture," IJCSN -International J. Comput. Sci. Netw,, vol. 3, no. 6, pp. 2277–5420, 2014.
12. Alioua, Ahmed, *et al*. "Efficient data processing in software-defined UAV-assisted vehicular networks: A sequential game approach." *Wireless Personal Communications* 101.4 (2018): 2255-2286.
13. Singh, Smita, Sarita Negi, and Shashi Kant Verma. "VANET based p-RSA scheduling algorithm using dynamic cloud storage." *Wireless Personal Communications* 98.4 (2018): 3527-3547.
14. Zhang, Yang, Jing Zhao, and Guohong Cao. "On scheduling vehicle-roadside data access." *Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*.(2007): 9-18.
15. Kumar, Rohit, *et al*. "A Collective Scheduling Algorithm for Vehicular Ad Hoc Network." *Recent Trends in Communication, Computing, and Electronics*. Springer, Singapore, 2019. 165-180.
16. Zhu, Wanting, *et al*. "SDN-enabled hybrid emergency message transmission architecture in internet-of-vehicles." *Enterprise Information Systems* 12.4 (2018): 471-491.
17. Singh, Smita, *et al*. "Comparative study of existing data scheduling approaches and role of cloud in vanet environment." *Procedia Computer Science* 125 (2018): 925-934.
18. M. Asgari, M. Shahverdy, and M. Fathy, "A qos-based scheduling algorithm in vanets" J. Theor. Appl. Inf. Technol., 77. 3 (2015): 429–437
19. Lim, Joanne Mun-Yee, *et al*. "Performance modelling of adaptive VANET with enhanced priority scheme." *KSII Transactions on Internet and Information Systems* 9.4 (2015): 1337-1358.
20. Javad, Sayadi Mohammad, and Fathy Mahmood. "A new approach in packet scheduling in the VANET." *arXiv preprint arXiv:1010.0430* (2010).
21. Dubey, Brij Bihari, *et al*. "Priority based efficient data scheduling technique for VANETs." *Wireless Networks* 22.5 (2016): 1641-1657.
22. Vasudev, Harsha, and Debasis Das. "A trust based secure communication for software defined VANETs." *2018 International Conference on Information Networking (ICOIN)*. IEEE, (2018): 316-321
23. Kalinin, Maxim, *et al*. "Software defined security for vehicular ad hoc networks." *2016 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, (2016): 533-537
24. Peng, Huijun, *et al*. "A detection method for anomaly flow in software defined network." *IEEE Access* 6 (2018): 27809-27817.
25. Mousavi, Seyed Mohammad, and Marc St-Hilaire. "Early detection of DDoS attacks against SDN controllers." *2015 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, (2015): 77-81
26. Zhou, Yousheng, *et al*. "An efficient V2I authentication scheme for VANETs." *Mobile Information Systems* 2018 (2018).
27. Liyanage, Madhusanka, *et al*. "Opportunities and challenges of software-defined mobile networks in network security." *IEEE Security & Privacy* 14.4 (2016): 34-44.

28. Akhunzada, Adnan, *et al*. "Securing software defined networks: taxonomy, requirements, and open issues." *IEEE Communications Magazine* 53.4 (2015): 36-44.
29. Shafiq, Hammad, Rana Asif Rehman, and Byung-Seo Kim. "Services and security threats in sdn based vanets: A survey." *Wireless Communications and Mobile Computing* 2018 (2018).
30. Sadio, Ousmane, Ibrahima Ngom, and Claude Lishou. "SDN architecture for intelligent vehicular sensors networks." *2018 UKSim-AMSS 20th International Conference on Computer Modelling and Simulation (UKSim)*. IEEE, (2018): 139-144.
31. Wadhonkar, Arnav, and Deepti Theng. "An Analysis of Priority Length and Deadline Based Task Scheduling Algorithms in Cloud Computing." *IJCSN International Journal of Computer Science and Network* 5.2 (2016): 360-364.
32. Ning, Wang, *et al*. "A task scheduling algorithm based on qos and complexity-aware optimization in cloud computing." (2013): 5-5.
33. Panwar, Neelam, *et al*. "An enhanced scheduling approach with cloudlet migrations for resource intensive applications." *Journal of Engineering Science and Technology* 13.8 (2018): 2299-2317.
34. Ullah, Insaf, *et al*. "A novel provable secured signcryption scheme: A hyper-elliptic curve-based approach." *Mathematics* 7.8 (2019): 686.