Over the Horizon of Connected Vehicles: Advancing Security, Reliability, and Cost-Efficiency through Vehicular Edge Computing

Thesis submitted in partial fulfillment
of the requirements for the degree of

*Master of Science in **Computer Science and Engineering** by Research*

by

SVSLN Surya Suhas Vaddhiparthy
2021701006
suhas.vaddhipar@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled **"Over the Horizon of Connected Vehicles: Advancing Security, Reliability, and Cost-Efficiency through Vehicular Edge Computing"** by SVSLN Surya Suhas Vaddhiparthy, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Dr. Deepak Gangadharan

To my Mother, Brother, and my Late Father

Acknowledgments

I want to express my sincere gratitude to my research advisor, Dr. Deepak Gangadharan, for his constant support and guidance throughout my masters. His encouragement to pursue an idea has always been motivational and unparalleled. I am highly grateful for the exposure to diverse research domains, and his advice has helped me learn the essence of research and problem-solving practically. He has always helped me improve my writing and presentation skills, which have been invaluable during my master's journey and will also help me in the future.

I would like to also thank Dr. Sachin Chaudhari, Dr. Ramesh Loganathan, and Mrs. Anuradha Vattem for allowing me to work at Smart City Living Lab IIIT-H. I sincerely thank Dr. Pradeep [currently affiliated to SAINT GITS, India] for formally introducing me to the realm of research and his guidance about the way of life during tough times. I thank Dr. Aftab M. Hussain, Dr. Karthik Vaidhyanathan, and Dr. Harikumar Kandath for their valuable lessons and interactions, which significantly impacted my research. I am wholeheartedly grateful to Dr. Baek Gyu Kim [currently affiliated with DGIST, South Korea] for the valuable, constant support and input that helped me make a research idea a successful publication.

I thank my mentor and friend, Shubham Mante, for his guidance and suggestions throughout my masters. I thank my best friends, Ganesh CGS and Ruthwik Muppala, for making my campus life memorable. Thanks to Joseph Cherukara for all the help and discussions during my research. I thank Ashish Sharma, Usha Goparaju, and other lab mates for their support during placements.

A special thanks to Harshitha Chollangi, Mallesh Kattera, and Ramesh Yadav for constantly supporting me throughout my masters. Finally, I would like to thank my dearest family members for their encouragement and support during all the ups and downs of my life.

Abstract

The rapid technological advancements have significantly impacted numerous industries, including the automotive sector. Amid the pursuit of creating a better vehicle, Connected Vehicles (CV) emerged as a candidate technology that promises safety and enhances the overall user experience. The initial trials of realizing the CV technology involved the usage of Cloud Computing, but the idea was short-lived due to high latency and bandwidth requirement issues. This deadlock scenario was then handled by bringing the computation resources closer to the user through Mobile Edge Computing (MEC). MEC involves localized computation units that are relatively less powerful than the cloud but can handle the CV requirements. MEC technology was further enhanced by considering the CVs' dynamic network topology, which was then called Vehicular Edge Computing (VEC).

Realizing CV technology through the VEC approach involves a central cloud server with multiple edge servers handling the requirements of the CVs. A general request from a CV can involve data delivery or task offloading requirements via edge servers. Although VEC technology offers a nearly assured solution to realize CV technology, it is often faced with challenges, such as mobility, resource allocation, security, affordability, computation delay, scalability, power consumption, caching, etc.

Considering a few such critical challenges, this work aims to bridge the gap in realizing CV technology through two contributions. The first work proposes a framework that incorporates MAC protocol constraints in the data delivery optimization framework to ensure practicality and reliability in data transmission. This work presents a data-frame collision-free optimization framework by adopting a time-slot-based MAC layer strategy that uses slot assignment to ensure collision-free data delivery for multiple vehicles across various transmission channels at each edge in different test conditions. The second contribution incorporates security constraints along with the price incurred for task offloading from vehicles to edges. This work presents a price optimization framework that minimizes the overall price for realizing the network, making it affordable while considering various task-specific security requirements. Further, both works consider various CV-specific constraints, such as vehicle flow, edge resources, and overlaps, thereby ensuring practicality.

Contents

List of Figures

List of Tables

List of Acronyms

ADAS Advanced Driving Assistance System

WHO World Health Organization

AV Autonomous Vehicles

CV Connected Vehicles

SDV Self-Driving Vehicles

ITS Intelligent Transportation System

US-DoT United States Department of Transportation

DSRC Dedicated Short Range Communication

CCH Control Channel

SCH Service Channel

Caltrans California Transportation Department

V2V Vehicle to Vehicle

V2I Vehicle to Infrastructure

V2P Vehicle to Pedestrian

V2E Vehicle to Everything

VANETS Vehicular Ad-HOC Networks

RSU Road Side Unit

V2R Vehicle to Road Side Unit

AECC Automotive Edge Computing Consortium

V2C2V Cooperative-Vehicle-to-Vehicle

MCC Mobile Cloud Computing

MEC Mobile Edge Computing

VMs Virtual Machines

VEC Vehicular Edge Computing

SDN Software Defined Network

TDMA Time Division Multiple Access

MAC Medium Access Layer

HCCA Hybrid Coordination Function Controlled Channel Access

VeMAC MAC for VANETS

UAVs Unmanned Aerial Vehicles

LOS Line of Sight

NLOS Non-Line of Sight

Chapter 1

Introduction

Recent technological advancements, the implementation of sustainable policies, and shifts in consumer preferences have significantly accelerated urbanization. According to a report by the United Nations Department of Economic and Social Affairs, approximately 55% of the world's population resided in urban areas as of 2018, with projections indicating an increase to around 70% by 2050 [1]. This swift urban development has profoundly impacted numerous industries, including the automotive sector. McKinsey's *Automotive Software and Electronics 2030* report [2] forecasts that the global automotive software and electronics market will reach $462 billion by 2030. The report discusses four significant trends: autonomous driving, connected vehicles, electrifying the powertrain, and shared mobility, collectively known as ACES, as the forces disrupting the automotive sector. These mutually reinforcing forces reflect a significant shift in the future of mobility, propelled by the expansion of urban access restrictions like bans on internal combustion engines, higher adoption of nonownership models, and efficient technologies such as the Advanced Driving Assistance System (ADAS) [3] [4].

The COVID-19 pandemic and its aftermath have profoundly impacted customer preferences and regulations, leading to an increased emphasis on road safety, security, and better connectivity [5]. The growing concerns about safety requirements are consistent with the statistical report conducted by the Indian parliament [6]. Based on the road accident data from the years 1970 to 2023, the report quotes that annually, 1.5 lakh people die on Indian roads, which scales down to 18 people per hour. As per the report by the World Health Organization (WHO), the number of deaths due to road accidents drastically reaches 11.9 lakh people annually on a global scale [7]. Further, road traffic crashes are estimated to cost most countries 3% of their gross domestic product. The *Five Trends Transforming the Automotive Industry* report by PWC spotlights the potential of Autonomous Vehicles (AV) and Connected Vehicles (CV) as a key in creating safer, more efficient vehicles capable of covering extra miles with reduced maintenance costs [8]. The inclination for electric AVs is increasing due to their potential to drastically improve road safety, reduce traffic congestion, and offer significant environmental benefits. Furthermore, the report forecasts an impending surge in CV demand, pushed by consumers' growing interest in better connectivity features such as infotainment, web surfing, payments, location-based services, and music streaming [3].

As we stand on the cusp of a revolution in urban mobility, the promising capabilities of AVs and CVs beckon us toward a new era of transportation that is not only safer and more efficient but also seamlessly integrated with the digital fabric of our lives [8].

1.1 Self-Driving and Autonomous Vehicles vs Connected Vehicles

AV, Self-Driving Vehicles (SDV), and CV are often mentioned in the same breath due to their similarities in using technology and information to improve safety. Still, these technologies differ significantly due to how they interconnect and handle information.

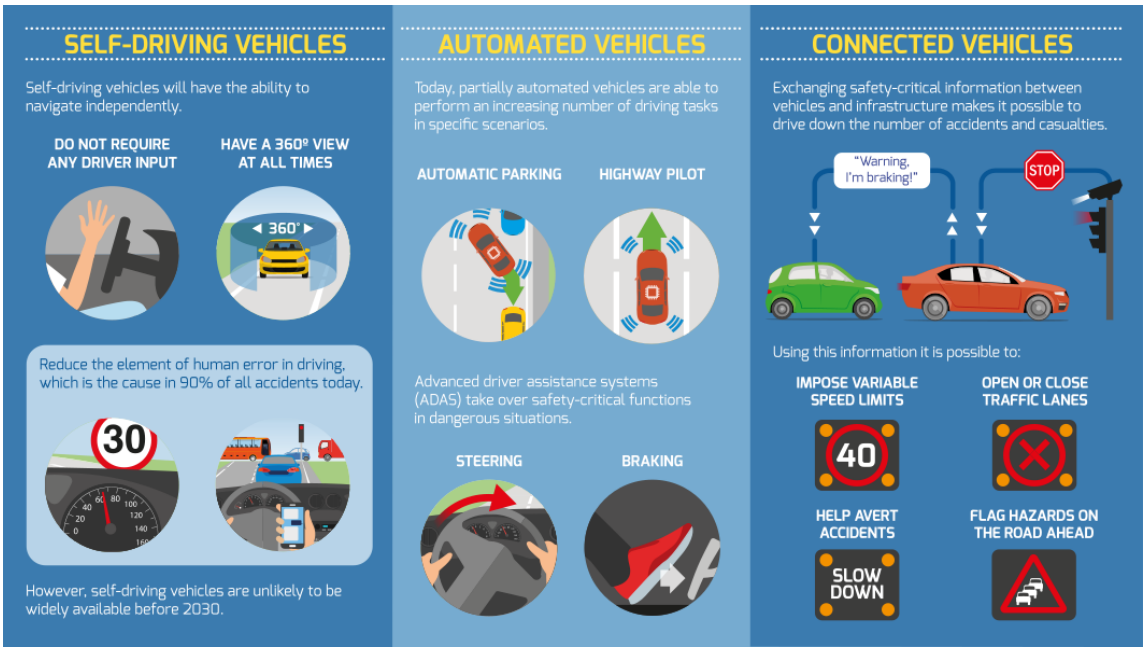


Figure 1.1: Comparision of AVs and CVs

Figure 1.1 [9] from the European Union road safety facts report compares SDVs, AVs, and CVs. SDVs and AVs use the capabilities of onboard sensors and ADAS technologies within the vehicles to independently/semi-independently make driving decisions, especially in dangerous situations, to ensure the safety of the passengers [8] [10]. Both of these technologies act as prescriptive technologies to avoid any impending hazards. On the other hand, CVs follow a preventive approach to handle potential dangers by exchanging safety-critical information with other vehicles and infrastructure. Apart from safety, the Intelligent Transportation System (ITS) white paper on CVs by the United States Department of Transportation (US-DoT) [11] anticipates that adapting CVs can mitigate over 80 % of non-impaired incidents as the CV technology warns of turnings, stopped vehicles, and other situations in advance, thereby providing sufficient reaction time. They further highlight that a widespread deployment can enable

cooperative cruise control vehicle platoons, avoiding unnecessary braking and stopping at intersections. These benefits can significantly reduce overall fuel consumption and, thereby, emissions [12] [13].

1.2 Adopting Connected Vehicles

CVs utilize a short-range protocol called Dedicated Short Range Communication (DSRC) for data exchange [14]. Owing to its capabilities over its counterpart, the C-V2X technology DSRC has been adopted as the candidate protocol for CV applications [15]. DSRC uses a multi-channel approach to facilitate seamless data exchange between network devices. Motivated by the capabilities of vehicular communication technology, the United States Federal Communication Commission has allocated a 75 MHz spectrum in a 5.9GHz band for DSRC [16].

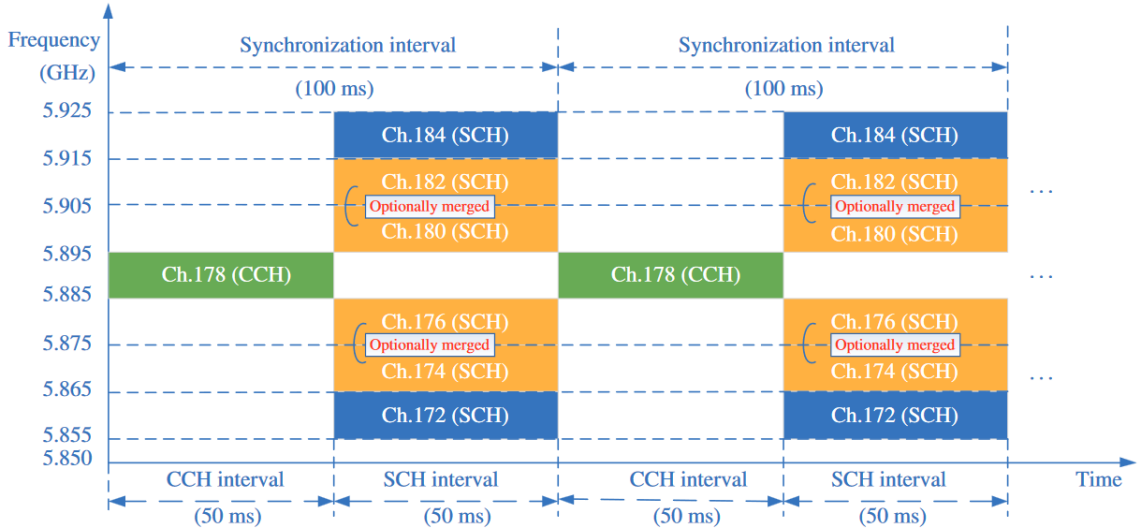


Figure 1.2: Typical Communication band in Vehicular Technology

Figure 1.2 [17] illustrates a typical channel division and bands for vehicular data communication. 1 Control Channel (CCH)and 6 Service Channel (SCH) are used for data exchange between network entities. CCH and SCH intervals are together referred to as Service Interval, which repeats for every 100 ms. CCH handles the control and public information, whereas SCH handles data dissemination depending on the application and mode of communication [17].

California Transportation Department (Caltrans) lists 4 communication modes and general use cases in a CV scenario as mentioned in Table 1.1 [18]. Among the 4 scenarios, Vehicle to Vehicle (V2V), Vehicle to Infrastructure (V2I), Vehicle to Pedestrian (V2P), and Vehicle to Everything (V2E), V2I and V2V act as representative candidates (together called as Vehicular Ad-HOC Networks (VANETS)) for vehicular technology communication [19]. V2V applications are mainly adapted for pre-crash sensing, blind spot detection, and for building cooperative forward collision warning systems. V2I (Also called Vehicle to Road Side Unit (V2R)) are adapted for communication-based applications such as curve-speed

warning, traffic violation detection, data delivery applications, and task offloading applications [18]. The current work has emphasized advancing the V2I scenario.

S.No.	Mode	Use Case
1	Vehicle-to-Vehicle (V2V)	Speed, Location, and Heading Information.
2	Vehicle-to-Infrastructure (V2I)	Signal timing, Work zones, Crashes, Congestion, and Weather conditions.
3	Vehicle-to-Pedestrian (V2P)	Information between vehicles, crosswalks, and bicyclists presence.
4	Vehicle-to-Everything (V2E)	Travel Times, incident response

Table 1.1: Different Modes of Communication in CVs from Caltrans

1.3 Need for V2I Communication

The article by Forbes on CVs highlights that a CV-enabled car would be a digital platform with many sensors that continuously generate data. The article highlights that the data traffic from connected vehicles can surpass 10 exabytes per month by 2025 [20]. Global X ETFs [21] and Automotive Edge Computing Consortium (AECC) [22] estimates around 0.383 TB per hour for a minimally connected car and up to 450 TB per day for a robo-taxi. The generated data breakdown is shown in Table 1.2. Altogether, 1 CV can produce up to 5100 TB of data annually [23].

S.No.	Sensor	Count	Data (Mbit/s/sensor)
1	RADAR	4-6	0.1-15
2	LiDAR	1-5	20-100
3	Camera	6-12	500-3,500
4	Ultrasonic	8-16	0.01
5	Others	1-each	0.1

Table 1.2: Data Generated in a CV

On the other hand, as of 2024, the global average mobile download speed is capped at 50 MB/s and upload speed at 11 MB/s, which can potentially constrain the high-velocity data of the CVs [24]. Therefore, in this bleeding edge of bandwidth requirements, the AECC has presented a distributed cloud computing approach involving localized computational and storage units closer to the vehicles for

seamless and lower latency responses. These localized units, called edge servers (Road Side Unit (RSU)), are adapted to facilitate automotive big-data requirements [22].

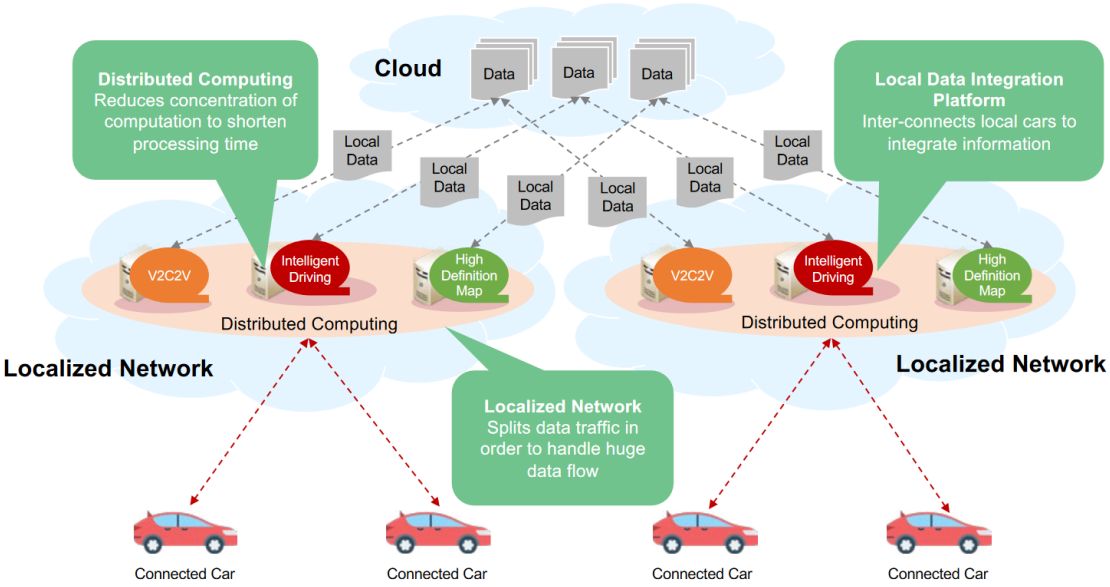


Figure 1.3: Distrubuted Cloud Computing Architecture by AECC

Figure 1.3 [22] illustrates the AECC's distributed cloud computing approach for CVs. Here, a cloud layer acts as a central unit to facilitate the requests from a CV. The cloud then allocates the necessary localized computing unit for the CV. The report also mentions various CV applications, such as Cooperative-Vehicle-to-Vehicle (V2C2V) Communication for road safety and Intelligent driving assistance systems for HD map data dissemination for awareness of location-sensitive and volatile information in maps.

1.4 Vehicular Edge Computing (VEC)

Initially, Mobile Cloud Computing (MCC) is poised as a suitable candidate to handle CVs' enormous data and computing requirements. MCC offers various advantages, such as flexible storage capacity, sophisticated services, and more extensive computation resources. However, MCC limited usage of these resources due to more significant latencies. Mobile Edge Computing (MEC) (also referred to as FOG Computing) handled this issue by bringing the computing resources closer to the users and CVs. MEC was further transformed to handle various CV-specific concerns, such as computation resource sharing through Virtual Machines (VMs), Software-Defined Networking, and Network Function Virtualization, leading to the Vehicular Edge Computing (VEC) paradigm [25] [26].

VEC emerged as an enhanced version of MEC with vehicular networks and aims to move communication, computation, and caching resources much closer to the CVs. VECs differ from MECs in

dynamic network topology due to fast-moving vehicles. Further, VECs can quickly adapt to localized environmental conditions, enabling them to analyze and disseminate real-time location-specific traffic and safety information. Figure 1.4 [27] illustrates a VEC-based approach where the RSU communicates with the CVs through the V2I communication mode for various CV applications [27].

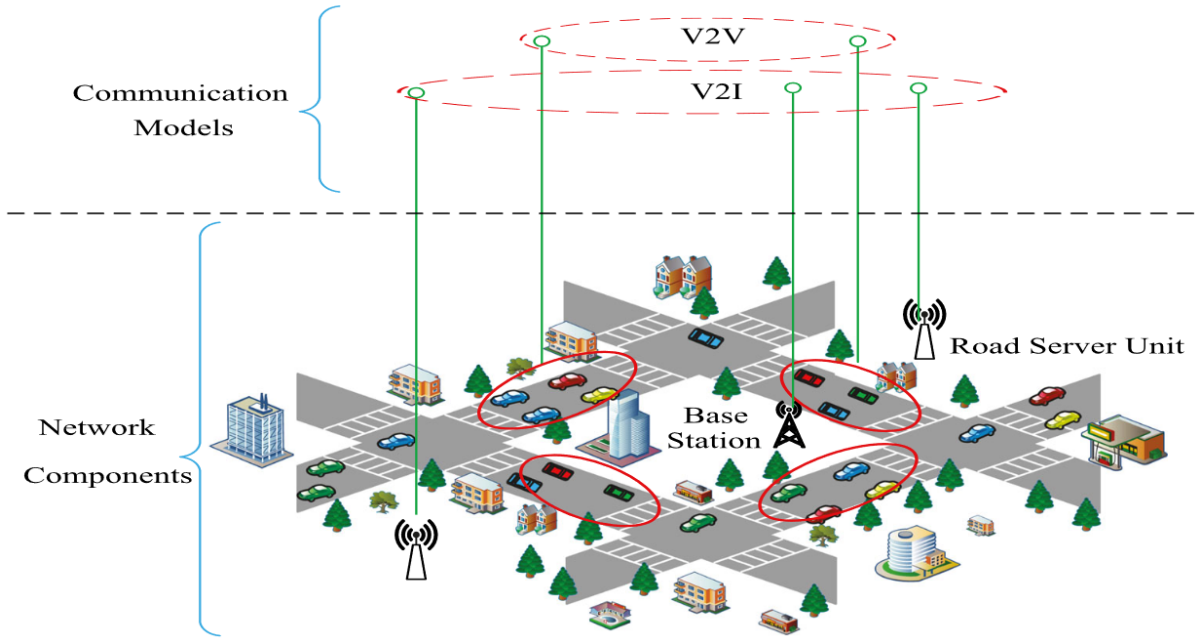


Figure 1.4: Idea of VEC for Connected Vehicles

1.4.1 VEC Architectures

Depending upon the resource management granularity, there are three main VEC architectures: two-layered, three-layered, and four-layered. The increase in layers comes with the advantage of efficient resource management but also increases the overall scheduling overhead. The description of these approaches is as follows [28] [29]

Two-Layered: Two-layered architecture has a stationary edge and vehicle layers. Here, the CV communicates directly with the stationary edges for services. This approach yields lower latency. However, the absence of a control cloud layer leads to insufficient resources during a higher demand for service [30].

Three-Layered: Unlike Two-Layered architecture, this approach has a control cloud, which helps central resource management and efficient resource allocations to CVs. In this approach, the edge layer must handle request flow from the vehicles, resource allocation, and processing, leading to higher overhead [29].

Four-Layered: The Four-Layered approach uses a Software Defined Network (SDN) layer along with the stationary edges, control cloud, and vehicle layer. SDNs increase the system's modularity and help ensure the vehicle request processing stays isolated from edge processing, thereby increasing edge efficiency. However, using SDNs induces higher network and processing overhead [31].

The current work adopts a three-layer VEC architecture. The advantages and disadvantages of each approach are summarised in Table 1.3 [28].

S.No.	Layers	Advantages	Disadvtanges
1	2	Less Commincation delay	Overhead on Cloud
2	3	Reduced Cloud Overhead	Absence of Flow-Control at Edge
3	4	Flexible applications due SDNs	High network overhead

Table 1.3: Advantages and Disadvantages of various VEC architectures

1.4.2 Applications based on VECs

Apart from the VEC applications such as V2C2V and emergency warning systems, there are two broad categories of VEC-based applications: Data Delivery and Task/Computation Offloading. Data delivery involves disseminating HD Maps, Parking Lot Information, and multimedia file-sharing data from the RSU to the vehicle. On the other hand, Task/Computation offloading involves processing data sent by the CV at the RSU for applications such as total/partial autonomous driving, sign board translation, etc. In both scenarios, the required service/data can be offered to the vehicle by the RSU as long as the CV is in the coverage region of the RSU. Here, the coverage region corresponds to the maximum distance from the RSU to the CV, where a CV can exchange information. Table 1.4 lists the various applications of a CV system.

S.No.	Application	Description
1	HD Maps [22]	Updating user on real-time information
2	Driver Health Monitoring [32]	Analyzing driver health and behavior
3	Autonomous Driving [33]	Handling vehicle maneuver and remote piloting
4	Traffic light Management [34]	Dynamically controlling traffic lights
5	Vehicle Tracking and Platooning [35]	Updating user about surroundings
6	Infotainment [36]	File Sharing, Gaming and Collaboration

Table 1.4: Various VEC Applications

1.4.3 Challenges of VEC

Adapting VEC involves various challenges, such as

- **Vehicle Mobility:** The high and variable mobility of the vehicles creates a dynamic network topology, resulting in variable network traffic and delays. CV switches between multiple RSUs based on allocation and requirements [37].
- **Resource Allocation:** VEC involves RSUs with finite computation and storage resources. Therefore, it is crucial to maintain resource management for stable functionality [30].
- **Job Completion & Computation Delay:** Owing to higher mobility, the data delivery or the task offloading needs to be completed while the vehicle is in the coverage of the RSU or within a specific deadline [25].
- **Security and Privacy:** The varying network topology increases vulnerabilities. There is a crucial requirement of security as the RSU/Edge can be accessed by various network components [28].
- **Handling Multiple Requests:** In a practical scenario, multiple CVs can arrive and request services simultaneously. Therefore, the RSU should be capable of handling multiple requests simultaneously [19].
- **Affordability:** Resource Allocation and RSU utilization incur a price depending upon the configuration and vendor. Suitable RSUs should be utilized to ensure user affordability [38].
- **Caching:** CVs belonging to a localized group can have highly correlated services. Therefore, effective caching can improve the overall resource utilization [39].
- **Power Consumption:** Like any other resource, power consumption is a crucial factor for ensuring the practicality of the system [25].

1.5 Contribution of the Work

The current work focuses on CVs that need data delivery and task offloading services from an RSU. The CV travels through various RSUs in the network from a specific point, and the current work ensures that the required services are allocated to the CV before it reaches its destination. The current work can be broadly divided into two major components as below

Data Frame Collision-Aware Data Delivery to CVs: In this work, an optimization framework is presented to minimize the slot utilization cost at all the RSUs. A slot can be defined as a finite time unit that is used for data delivery to a CV. The slot model ensures practical parallel data delivery to vehicles, which can otherwise be impossible due to network collisions. The work also ensures that RSU's resources are utilized efficiently while ensuring data delivery to all the CVs in the network.

Price Optimal Task Offloading with Security: A price optimization framework is proposed in this work, which ensures suitable RSUs are utilized to safeguard the CVs' affordability of the task offloaded to the RSU. The work also provides appropriate security classes based on the task required by the vehicle while ensuring the RSU's resources are not over-utilized.

1.6 Thesis Organization

Chapter 2 discusses the current literature on VECs, data delivery to CVs, and task offloading to RSUs. The chapter then elaborates on existing works dealing with simultaneous CV access, followed by existing security and pricing approaches.

Chapter 3 presents a slot cost minimization framework that ensures that a minimal number of slots are utilized at each RSU, inherently preventing potential data frame collision during data delivery to multiple vehicles. The work also accounts for delays due to numerous CVs at an RSU. The work presents the maximum number of CVs served with variation in total RSU count and also presents the variation in slot utilization cost at the RSU followed by the effect of traffic density for data delivery

Chapter 4 presents a price optimization framework for task offloading to RSUs. This work minimizes the overall operational and maintenance price incurred by the RSUs in the network while securely serving and processing the task data given by the CVs. The work presents the capability of the proposed framework by comparing it with two heuristics. Further, the work presents the total price incurred for various security requirements and problem sizes.

Chapter 5 discussed an additional contribution: *A Comprehensive evaluation of various GPS Spoofing scenarios in a UAV* which analyses 16 possible scenarios in indoor, outdoor, and different environmental conditions.

Chapter 6 concludes this work and provides some improvements that can be incorporated to further enhance the current work.

Chapter 2

Related Works

This chapter presents the existing literature on VECs, followed by two CV services, namely data delivery and task offloading. Further, it presents various works highlighting the adaptation of TDMA MAC protocol for simultaneous data delivery, followed by a discussion on security in VECs. The last section elaborates on computation cost in VECs.

2.1 Vehicular Edge Computing

Many works have emphasized and demonstrated the capabilities of VECs for data delivery to multiple vehicles [28] [27] [40] [41]. The survey by Liu et al. [27] elaborates on the various capabilities of a VEC system. The authors have highlighted the different aspects that can be considered for data delivery, such as energy, mobility awareness, etc. Although the work significantly gives insights into a typical VEC system, it has not evaluated the problem of data frame collisions due to simultaneous data delivery to multiple vehicles. Jie et al. [40] have presented a VEC network with a load-balancing approach. The work has demonstrated an efficient strategy to reduce processing delays across the network. However, the work has not elaborated on the effects of bandwidth utilization at an RSU.

Akbar et al. [42] have presented an optimization approach based on an exponential particle swarm algorithm to optimize energy and latency in the network. The work optimally chooses an edge for efficient resource allocation. Further, the work considers various VEC environment conditions such as mobility, bandwidth, and communication time. Although the work presents an optimal scenario for resource allocation, it doesn't account for security vulnerabilities as well as the practical problem of data frame collisions for data exchange.

2.2 Data Delivery using VEC

The work by Ryangsoo et al. [43] describes two data pre-fetching approaches for data delivery from edge to vehicles. The deterministic greedy algorithm converges to a suboptimal solution, whereas the second approach uses an online learning approach to account for network failures and converges

to an optimal solution. The experiments were conducted considering the limited edge resources and vehicle route traces. Although the work was efficient, it doesn't highlight the potential data losses due to collisions at an RSU because of simultaneous access.

In the prior work [44] [37], we have presented an optimal approach considering an overlapping window for vehicles arriving at an RSU in an equivalent time instant. Considering several constraints, the method uses an optimization framework for data allocation at multiple RSUs. However, the work presents a bandwidth division approach for data delivery to multiple vehicles, which, in real-time, can lead to data frame collisions and result in unsuccessful data delivery.

2.3 Task Offloading

Several works have considered various objectives, such as energy, time delay, and reward/incentive-based methods for building task-offloading approaches. For instance, Zhaolong et al. [45] have used finite Markov chain models for resource allocation and task offloading. This work implements a two-sided matchmaking policy using deep Q networks, considering execution delay and energy as a joint objective problem, but the work has not considered the required constraints for data delivery in their implementation. Jingyun et al. [46] have introduced the Autonomous Vehicular Edge (AVE) framework based on ant colony optimization. They have used a task scheduling approach for resource management and job caching in vehicular clouds. However, they have not considered a multi-RSU framework for task offloading.

2.4 Simultaneous CV Access using TDMA-MAC

Multiple MAC protocols have been proposed for vehicular scenarios based on the TDMA approach. The survey by Hadded et al. [47] has discussed the features and benefits of TDMA-based MAC layers. Xiaoming et al. [17] have elaborated on the usage of the Hybrid Coordination Function Controlled Channel Access (HCCA) MAC protocol, which has cycles of Service Intervals (SI), where each SI has two sub-components Control Channel (CCH) and Service Channel (SCH). All the vehicular nodes contend for channel access and medium control in the CCH phase, and the allocated vehicles exchange data in the SCH phase. Suchi et al. [48] have proposed a multi-channel TDMA MAC, which uses six SCH channels and one CCH channel. The SCH channels are allocated based on the application of the data, which considers access and merging collisions. The work by Hassan et al. [19] presents the state-of-the-art multi-channel VeMAC protocol, which addresses the access collisions and merging collisions for vehicles coming in opposite directions. All four works have stressed the need for a MAC protocol for vehicular data exchange. However, the works have not considered collision-aware data delivery to multiple vehicles from multiple RSUs.

2.5 Security in VEC

CVs utilize VEC services for either data delivery or computation offloading, which involves data exchange over an ever-changing network. The highly dynamic nature can make the VEC network prone to security vulnerabilities and therefore, security becomes crucial to ensure reliable CV services.

The survey by Nayak et al. [49] explained the need for security and privacy in a VEC scenario and then elaborated on various security attacks in a VEC scenario. The work then presents the various encryption and hashing algorithms to handle the security and privacy requirements. Additionally, the work presents a discussion on resource allocation considering security requirements. The survey by Fayi et al. [50] presents the need for security from a task-offloading point of view. The work presents a taxonomic overview of various domains within the VEC network and their security requirements, emphasizing blockchain-based security. The work also discusses various components of security requirements, such as confidentiality, integrity, availability, authenticity, and non-repudiation. Both surveys have summarised the security requirements in a VEC scenario but have not considered the need for various security class requirements regarding confidentiality, integrity, and availability. The current work considers five security classes based on the VEC application, where the higher class corresponds to the maximum security level and the lower class corresponds to the lesser security level.

The work by Chen et al. [51] has considered a VEC scenario with a trust-based security approach for task offloading. Although efficient, they have not considered the price implications of adopting various security algorithms across multiple RSUs. Several works [38] [52] [53] have also discussed the concept of security strength and security requirements in MEC scenarios. Security strength is defined as a relative measure of the security level of algorithms in terms of their computation requirement. These works have adapted a task scheduling approach to handle security requirements. However, they have not considered a multi-RSU scenario for security requirements with mobility constraints of the vehicular node.

2.6 Computation Cost in VEC

Computation cost is a crucial parameter to ensure the system's affordability for the user. Luo et al. [54] have adapted a multi-objective optimization of delay and computation cost. They have used the Particle Swarm Optimization algorithm to solve the task offloading problem. Du et al. [55] have considered a multi-objective problem that looks at non-orthogonal access of the CVs and cost-effectiveness. Both works have emphasized the need to consider computation cost and another objective but have not considered the implications of security vulnerabilities in their approach. Huang et al. [56] have built a cost-aware computation approach with security using deep reinforcement learning. An optimal offloading policy is built using deep Q-networks in a MEC environment. The approach presents an efficient way to minimize security risks. However, it cannot be adapted to a VEC environment as CVs stay for a finite duration in the coverage regions of RSUs.

Chapter 3

Collision Aware Data Delivery Framework

3.1 Introduction

The rapid evolution in communications technologies has spurred a drastic rise in Intelligent Transportation Systems such as Connected Vehicles. The theme of connected vehicles involves various transportation elements such as RSU, commute/freight vehicles, transportation centers, etc., communicating over short-range or peer-to-peer networks. With a quick and seamless data transfer as the primary objective, these systems use DSRC protocol to exchange safety messages, road congestion information, and weather information, thus improving the users' overall safety and travel experience [12]. Apart from safety information, the AECC has mentioned one crucial application, highlighting the advantage of dynamic delivery of HD maps with real-time traffic and pedestrian information from RSU to vehicles. The report also highlights that the volume of data exchanged can reach 183.4 PB/day by 2032, making the problem challenging. Further, the problem can be even more complex when the number of vehicles requesting the dynamic maps information increases. [22].

Considering the low latency conditions, reusability, and high volume of map data, one practical approach to meet the data delivery requirements is to adopt Vehicular Edge Computing (VEC). The survey by Meneguette et al. [28] on VECs has compared several multi-tier VEC architectures based on resource availability, computation capacity, etc., highlighting the capabilities of three-tier VEC architecture, which uses a central cloud server with a global view of the vehicles with multiple RSUs for efficient data delivery.

Figure 3.1 portrays a general three-tier architecture involving a central cloud server, RSUs, and vehicles. Each vehicle initially sends the information on required data to the cloud server, after which the cloud server allocates data to RSUs based on the vehicle's route and other factors. In an ideal scenario, the vehicle starts receiving the data as soon as it enters the coverage region of the RSU [44]. However, in a real-time situation, multiple such vehicles can enter the coverage region of the RSU simultaneously, leading to data frame collisions in the network due to parallel access, resulting in unsuccessful data delivery. This access and merging collisions can worsen due to differences in vehicular mobility and densities [19] [17].

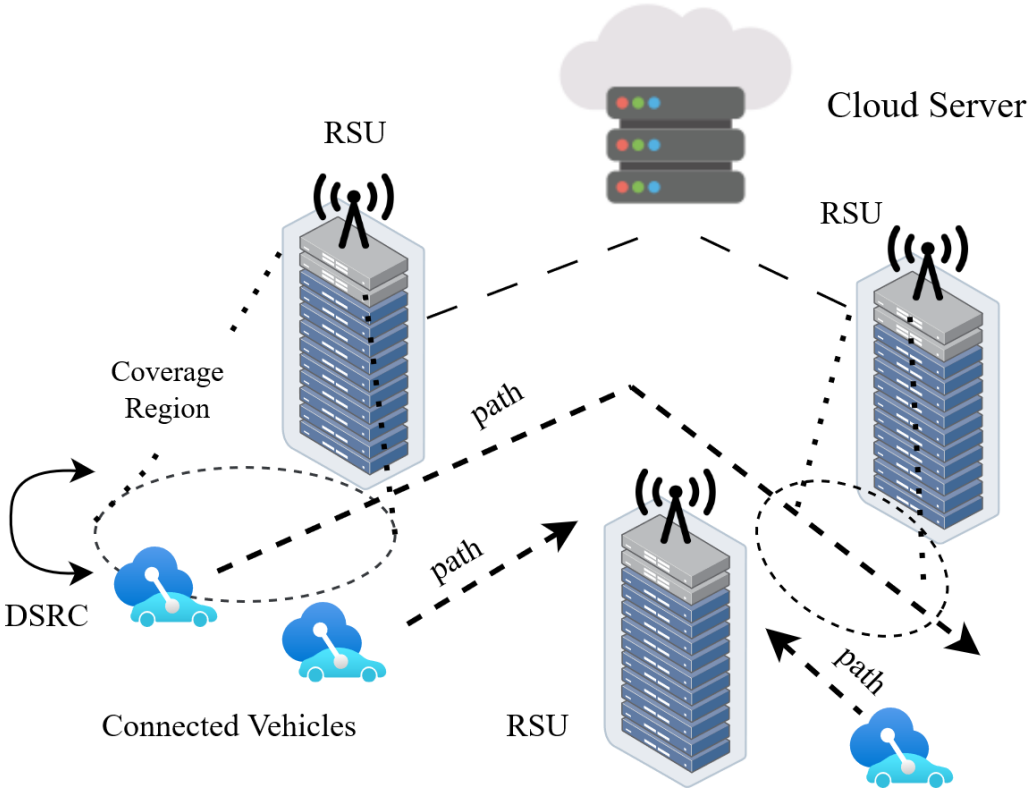


Figure 3.1: General Data Delivery Scenario

Therefore it is vital to adopt a Medium Access Layer (MAC) protocol, as the precision of data delivery is directly related to the safety and comfort of the users. The state-of-the-art MAC protocol MAC for VANETS (VeMAC) uses Time Division Multiple Access (TDMA), a time slot-based multi-channel protocol where one or more than one slot is allocated to a specific vehicle in each channel for collision-free data exchange. Given the resource availability at various edges and routes of vehicles, there are many possibilities of how the data can be allocated at each edge. However, in a real-time situation, the data frame collisions can increase with the number of parallel vehicles requesting the data. Therefore, the current work uses a slot utilization cost based on a pricing policy to reduce these collisions. The edge server with more utilized slots has more cost. Therefore, the data delivery happens from an edge with a lesser cost, minimizing data frame collisions. Based on this notion, an optimization framework is presented in the current chapter that minimizes the total slot utilization cost at all the edges considering various edge, vehicle, and timing constraints.

The contributions of this work are as follows.

1. To the best of the knowledge, this is the first work that proposes a TDMA MAC-based data collision-aware framework for data delivery from multiple edges to vehicles. This work has integrated two state-of-the-art MAC protocols considering the vehicular flow model.

2. This work presents the practical significance of the proposed framework by comparing it with an existing optimal framework for VEC that does not consider a MAC layer. We demonstrate that the existing work is optimistic with respect to the number of vehicles served, and our proposed framework shows the realistic scenario where the number of vehicles served is lesser due to collisions.

3. It also proposes an algorithm to calculate the worst-case delay for data delivery due to network contention by other overlapping vehicles that are simultaneously entering the edge's coverage region.

4. This work then presents several experiments to demonstrate the significance of the proposed framework using real-world traffic scenarios from the Luxembourg dataset.

3.2 System Model and Problem Definition

This section first presents our VEC system model consisting of a central cloud server, edges, and vehicles. This section then elaborates on the problem of minimizing slot utilization cost across all the edges.

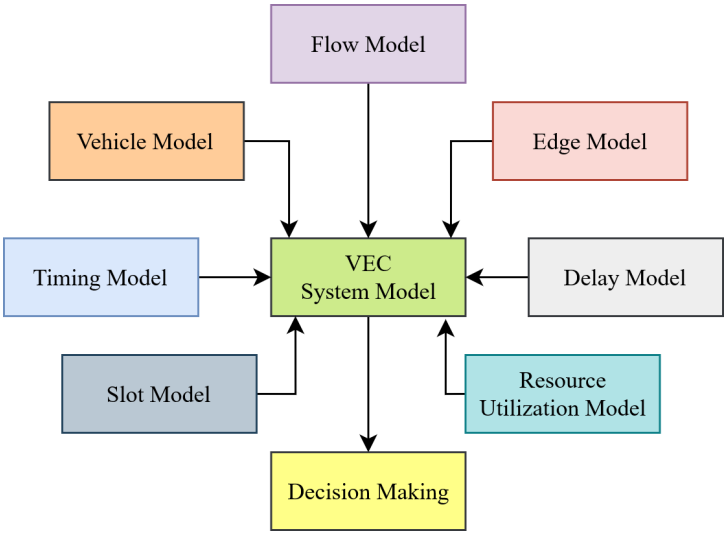


Figure 3.2: VEC System Models

3.2.1 VEC System Model

The VEC model includes a central cloud C connected to M heterogeneous edge servers $\{E_1, .., E_M\}$, each having K uniform and independent network channels with L time slots per channel and N vehicles $\{V_1, V_2, ..., V_N\}$ in the network. Each edge server has a finite coverage area for data delivery to vehicles

passing through that edge. *In the current work, we assume that the cloud maintains a route map from which it can infer and map the total data requested by the vehicles into smaller data chunks for delivery at the edges.* We model our VEC system using Vehicle, Flow, Edge, Timing, Delay, Slot, and Resource utilization models.

Vehicle Model: The vehicle model for vehicle V_i $(1 \leq i \leq N)$ is described as a tuple $V_i : \{x_{i,j}, m_i\}$, where $x_{i,j}$ contains the route information, and m_i is the total memory requested by V_i. Considering a two-lane scenario, if the vehicle V_i passes through edge E_j $(1 \leq j \leq M)$ and moves in the rightward direction, then $x_{i,j}$ is 1. If the vehicle V_i moves in the leftward direction, then the value of $x_{i,j}$ is -1. If the vehicle V_i doesn't pass through the edge, then the value of $x_{i,j}$ is 0.

Edge Model: The Edge Model for edge E_j $(1 \leq j \leq M)$ is described as $\{M_j, M_j^{occ}, L_j^{cov}, B_j, B_{j,k}\}$, where M_j is the total memory of the edge, M_j^{occ} is occupied memory for processing, L_j^{cov} is length of coverage region, B_j is the total bandwidth of edge, $B_{j,k}$ is the bandwidth of each channel of the edge. The relation between B_j and $B_{j,k}$ is given by Eq.(3.1).

$$B_j = \sum_{k=1}^{K} B_{j,k} \tag{3.1}$$

Flow Model: The velocity $v_{i,j}$ of vehicle V_i $(1 \leq i \leq N)$ at edge E_j $(1 \leq j \leq M)$ is calculated using the Greenshields equation [44]. $v_{i,j}$ depends on free flow velocity v_j^f, vehicle density k_j, and jam density of vehicles k_j^{jam} at edge E_j. These three quantities are related as shown in Eq.(3.2).

$$v_{i,j} = v_j^f * (1 - \frac{k_j}{k_j^{jam}}) \tag{3.2}$$

Timing Model: The timing model describes the various timing components of the current work. $t_{i,j}^{trav}$ is earliest travel time for vehicle V_i $(1 \leq i \leq N)$ to reach edge E_j $(1 \leq j \leq M)$. $t_{i,j}^{comm}$ is the time central cloud C takes to send data of vehicle V_i to edge E_j. t^{slot} is the period of each slot in a channel k $(1 \leq k \leq K)$. $t_{i,j}^{cov}$ is the coverage time of vehicle V_i at edge E_j. The relation between $t_{i,j}^{cov}$, l_j^{cov}, $v_{i,j}$ is given by Eq.(3.3).

$$t_{i,j}^{cov} = \frac{l_j^{cov}}{v_{i,j}} \tag{3.3}$$

Delay Model: In a TDMA MAC-based protocol system, as soon as the vehicle V_i $(1 \leq i \leq N)$ enters the coverage region of an edge E_j $(1 \leq j \leq M)$, the vehicle senses the channel for a beacon frame [46] for a time interval of t^{beacon}. After receiving the beacon frame, the vehicle starts contending for a time slot by sending requests at random intervals. $t_{i,j}^{slot\_delay}$ denotes the worst-case contention delay to transmit and receive the acknowledgment from the edge successfully. $t_{i,j}^{slot\_unav}$ is the time unavailable due to MAC protocol constraints. If $t_{i,j}^{slot\_av}$ is the available time for the vehicle to receive data after the contention phase, then the relation between $t_{i,j}^{cov}$, $t_{i,j}^{slot\_unav}$, $t_{i,j}^{slot\_delay}$, t^{beacon}, and $t_{i,j}^{slot\_av}$ is given by Eq.(3.4).

$$t_{i,j}^{slot\_av} = t_{i,j}^{cov} - t_{i,j}^{slot\_unav} - t_{i,j}^{slot\_delay} - t^{beacon} \tag{3.4}$$

Slot Model: In the current work, each channel k ($1 \leq k \leq K$) of an edge E_j is partitioned into uniform and independent time units called time slots. Each slot is of finite time unit t_{slot} and is described as a tuple $l_{j,k}$: $\{\rho, \tau, \gamma\}$ where ρ is a binary variable which indicates if a slot is available for data delivery, τ indicates the road lane to which the slot belongs, $\tau \in \{-1, 0, 1\}$. The element γ denotes the allocated vehicle id, $\gamma \in [1, N]$.

For a vehicle V_i ($1 \leq i \leq N$) at edge E_j ($1 \leq j \leq M$), $n\_slot_{i,j}^{max}$ is the total number of time slots based on the coverage time of vehicle V_i at edge E_j and can be calculated using Eq.(3.5). $n\_slot_{i,j}^{max\_alloc}$ refers to the maximum number of time slots that can be allocated considering the $t_{i,j}^{slot\_delay}$, t^{beacon}, and $t_{i,j}^{slot\_unav}$, which is calculated using $t_{i,j}^{slot\_av}$, as mentioned in Eq.(3.6). The floor Function is used to make sure there are no partial slots. The relation between $n\_slot_{i,j}^{max\_alloc}$ and $n\_slot_{i,j}^{max}$ is given by Eq.(3.7).

$$n\_slot_{i,j}^{max} = \left\lfloor \frac{t_{i,j}^{cov}}{t_{slot}} \right\rfloor \tag{3.5}$$

$$n\_slot_{i,j}^{max\_alloc} = \left\lfloor \frac{t_{i,j}^{slot\_av}}{t_{slot}} \right\rfloor \tag{3.6}$$

$$n\_slot_{i,j}^{max\_alloc} \leq n\_slot_{i,j}^{max} \tag{3.7}$$

Resource Utilization Model: The resources of an edge are utilized for storage and data delivery to vehicles. Therefore at any instant, the memory usage and the slot usage should not exceed the total memory capacity and the slots available at an edge. In the current work, we adopt the Earliest Time of Arrival (ETA) and Latest Time of Departure (LTD) from [37] for calculating the overlapping set of vehicles to determine the total slots and memory utilized. ETA refers to the time spent by a vehicle to reach an edge when it travels with minimum velocity. Similarly, LTD refers to the time spent to reach an edge when the vehicle travels with maximum velocity. For a vehicle, V_i ($1 \leq i \leq N$), the maximum and minimum velocities at edge E_j ($1 \leq j \leq M$) based on Eq.(3.2) are given by Eq.(3.8) and Eq.(3.9), where k^{min} and k^{max} refer to minimum and maximum densities at edge E_j. Using these equations, the ETA and LTD are computed using Eq.(3.10) and Eq.(3.11), as mentioned in [37], where Q is the set of edges the vehicle has passed through and $R_{m,n}$ is the road segment between the edges m and n.

$$v_{i,j}^{max} = v_j^f * (1 - \frac{k^{min}}{k_j^{jam}}) \tag{3.8}$$

$$v_{i,j}^{min} = v_j^f * (1 - \frac{k^{max}}{k_j^{jam}}) \tag{3.9}$$

$$t_{i,j}^{eta} = \frac{A_i}{v_{i,j}^{max}} + \sum_{m,n \in Q} \frac{R_{m,n}}{v_{i,j}^{max}} + \sum_{o \in \{Q - E_j\}} \frac{L_o^{cov}}{v_{i,o}} \tag{3.10}$$

$$t_{i,j}^{ltd} = \frac{A_i}{v_{i,j}^{min}} + \sum_{m,n \in Q} \frac{R_{m,n}}{v_{i,j}^{min}} + \sum_{o \in Q} \frac{L_o^{cov}}{v_{i,o}} \tag{3.11}$$

In Eq.(3.10) and Eq.(3.11), the first term signifies the time taken by the vehicle to reach the first edge from its starting point, where A_i is the distance between the starting point of the vehicle and the first edge. The second term computes the time the vehicle travels between the consecutive edges. If m,n are two consecutive edges in the set of edges Q of vehicle V_i, then $R_{m,n}$ is the distance between the edges. The last term accounts for the time spent passing through coverage regions of the edges Q of vehicle V_i; here, L_o^{cov} is the coverage region of an edge o belonging to edge set Q of vehicle V_i. The overlapping vehicle sets S_j of an edge are computed using the ETA and LTD values based on algorithm 1 in [37]. If $S_{j,ov}$ is an overlapping set of edge E_j such that $\{1 \leq ov \leq OV_j\}$, OV_j is the maximum number of overlapping sets of E_j, then the overlapping time is calculated as the maximum difference between LTD and ETA of the vehicles in the set $S_{j,ov}$, as shown in Eq.(3.12). Similar to Eq.(3.4) and Eq.(3.6), the overlapping slots $n\_slot_{j,ov}^{overlap}$ for an overlapping set $S_{j,ov}$ for data delivery can be calculated using Eq.(3.13), where $t_{j,ov}^{overlap\_unav}$ is the the unavailable time for data delivery for set $S_{j,ov}$. $n\_slot_{j,ov}^{overlap\_max}$ is the maximum overlapping slots among all the overlapping sets of edge E_j as shown in Eq.(3.14)

$$t_{j,ov}^{overlap} = t_{ov}^{ltd} - t_{ov}^{eta} \tag{3.12}$$

$$n\_slot_{j,ov}^{overlap} = \left\lfloor \frac{t_{j,ov}^{overlap} - t_{j,ov}^{overlap\_unav}}{t_{slot}} \right\rfloor \tag{3.13}$$

$$\begin{aligned} &n\_slot_j^{overlap\_max} = max(n\_slot_{j,1}^{overlap}, .., n\_slot_{j,OV_j}^{overlap}) \\ &ov \in \{1 \leq ov \leq OV_j\} \end{aligned} \tag{3.14}$$

Decision Variable $n\_slot_{i,j,k}^{alloc}$: In the current work, the number of slots allocated for a vehicle V_i $(1 \leq i \leq N)$ at edge E_j $(1 \leq j \leq M)$ in channel k $(1 \leq k \leq K)$ by the cloud server as the decision variable. $n\_slot_{i,j,k}^{alloc}$ takes a non-negative value if V_i is in the path of E_j. If V_i is not in the path of E_j, then $n\_slot_{i,j,k}^{alloc}$ = 0.

3.2.2 Problem Formulation

The current work builds a data collision-aware optimization framework for data delivery from edges to vehicles in a scenario where multiple vehicles enter the coverage region of the edge simultaneously while considering various data delivery and resource availability constraints. The frameworks run in the cloud layer to optimize the number of slots a vehicle utilizes at an edge in an overlapping time with multiple simultaneous vehicles.

3.3 Optimization Framework

This section first discusses worst-case timing delay values for various timing components involved in calculating the $n\_slot_{i,j}^{max\_alloc}$, the maximum slots available for data delivery. We then elaborate on various constraints necessary for data delivery from edges to vehicles. Finally, we present the objective function for optimizing the number of slots utilized for each overlapping set at an edge.

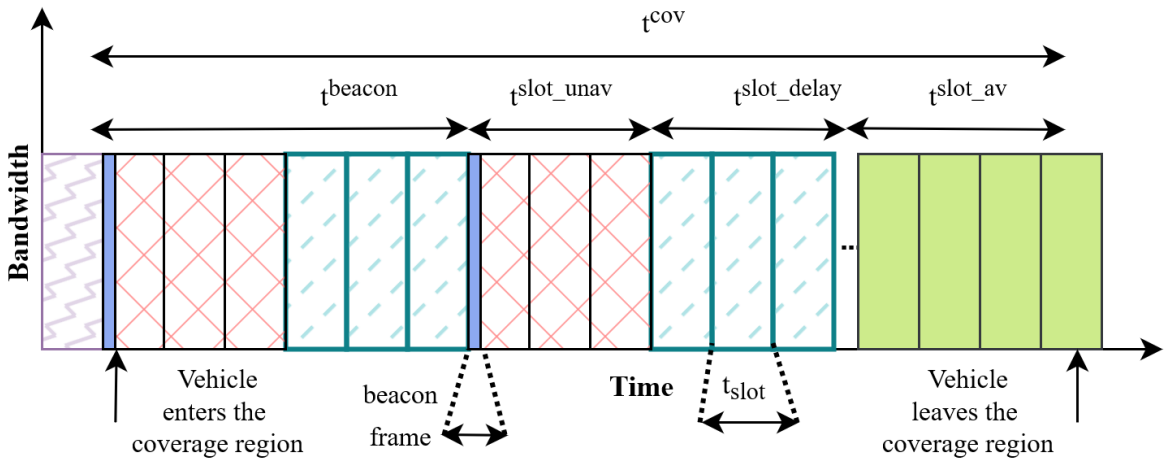


Figure 3.3: Various timing delays in a TDMA MAC Channel

3.3.1 Worst Case Slot Calculation

Figure 3.3 illustrates the various timing components in a general TDMA MAC-based channel when one vehicle enters and leaves the coverage region. The calculations for $t_{i,j}^{slot\_unav}$, $t_{i,j}^{slot\_delay}$, t^{beacon} are as follows.

Delay due to unavailable and beacon time: t^{beacon} is the maximum time delay to receive a beacon frame. The beacon signal is broadcasted at the end of the service interval cycle, where the service interval t_{SI} is a cycle of the control and service phases with a defined number of slots. In the worst-case scenario, the vehicle will wait for the entire service interval cycle to receive the beacon frame. $t_{i,j}^{slot\_unav}$ is unavailable time windows which are MAC protocol dependent. In the case of the VeMAC protocol, each service interval cycle is partitioned equally for vehicles moving in leftward and rightward directions. Similarly, the service channel slots are unavailable during the control channel phase in Hybrid Coordination Function Controlled Channel Access (HCCA). Therefore depending upon the arrival time and departure time of a vehicle at an edge, the relationship between $t_{i,j}^{cov}$ and $t_{i,j}^{slot\_unav}$ can be given by Eq.(3.15). f_{unav} is a positive real number between 0 and 1.

$$
\begin{aligned}
t^{beacon} &= t_{SI} \\
t^{slot\_unav}_{i,j} &= f_{unav} * t^{cov}_{i,j}, f_{unav} \in [0,1]
\end{aligned}
\tag{3.15}
$$

Delay due to contention: $t^{slot\_delay}_{i,j}$ is the worst-case time for a vehicle to send a control message and receive an acknowledgment from the edge. If $n^{ovc}_{i,j}$ is the total number of contenting vehicles in the overlapping vehicle set, then in the worst-case scenario, a vehicle would receive the acknowledgment after all the other contending vehicles. Algorithm 1 illustrates the method to calculate the worst-case overlapping set for a vehicle based on LTD and ETA at an edge. In line 2, the loop iterates over all the vehicles passing through the edge E_j leaving the current vehicle. Now in lines 3 to 6, the $n^{ovc}_{i,j}$ is incremented if the current vehicle V_k has the ETA between the ETA and LTD of the vehicle V_i and returns the total count in line 7. Therefore, if each contenting vehicle takes a service interval to send and receive an acknowledgment from the edge, then the total delay due to contention can be given by Eq.(3.16), where t_{SI} is the time interval of the service interval.

$$
t^{slot\_delay}_{i,j} = n^{ovc}_{i,j} * t_{SI} \tag{3.16}
$$

Algorithm 1 Algorithm to calculate overlapping vehicle count for a vehicle V_i passing through an edge E_j

Require: Current vehicle V_i, ETA, and LTD of the vehicles passing through the edge E_j.

Ensure: $n^{ovc}_{i,j}$, the number of overlapping vehicles.

Process:

1: $n^{ovc}_{i,j}$ = 0
2: **for** $V_k \in E_j$ **do**
3: **if** (ETA[i] $\leq$ ETA[k]) **and** (ETA[k] $\leq$ LTD[i]) **then**
4: $n^{ovc}_{i,j}$ += 1
5: **end if**
6: **end for**
7: **return** $n^{ovc}_{i,j}$

3.3.2 Optimization Constraints

Figure 3.4 illustrates the optimization constraints for collision-aware data delivery from edges to multiple vehicles. The constraints are detailed below.

Range Constraints: This constraint sets the lower and upper bound for memory $m_{i,j,k}$ allocated in a channel k of edge E_j for a vehicle V_i and the product of $n\_slot^{alloc}_{i,j,k}$ and $B_{j,k}$ is equal to memory allocated for vehicle V_i. The constraints are as follows.

$$\begin{aligned} & m_{i,j,k} = 0, x_{i,j} = 0 \\ & m_{i,j,k} \geq 0, |x_{i,j}| = 1 \\ & m_{i,j,k} \leq m_i, |x_{i,j}| = 1 \\ & m_{i,j,k} = n\_slot^{alloc}_{i,j,k} * B_{j,k} \\ & i = 1..N, j = 1..M, k = 1..K \end{aligned} \tag{3.17}$$

Accumulation Constraints: This constraint ensures that the summation of $m_{i,j,k}$ over all the channels of all the edges equals m_i, the memory requested by the vehicle V_i. The constraint is as follows.

$$\begin{aligned} & \sum_{j=1}^{M} \sum_{k=1}^{K} n\_slot^{alloc}_{i,j,k} * B_{j,k} = m_i, \\ & i = 1..N, j = 1..M, k = 1..K \end{aligned} \tag{3.18}$$

Time to Edge Constraints: The timing constraints ensure that the cloud has already transferred the data to the edge before the vehicle arrives at an edge. The constraint is as follows.

$$\begin{aligned} & m_{i,j,k} * t^{comm}_{i,j} \leq m_{i,j,k} * t^{trav}_{i,j} \\ & i = 1..N, j = 1..M, k = 1..K \end{aligned} \tag{3.19}$$

Edge resource Constraints: The Edge resource constraints ensure that the memory utilized at an edge at any instant is less than the total memory of the edge. In the current work, we consider the overlapping sets of vehicles to calculate the resources utilized at an edge by different vehicles. Using this constraint, we can ensure that the memory used for data delivery plus the memory occupied for internal processing is always less than the total memory of the edge at any given instant. The constraint is given below.

$$\begin{aligned} & \max(\sum_{i_1 \in S_{j,1}} \sum_{k=1}^{K} m_{i_1,j,k}, .., \sum_{i_{OV_j} \in S_{j,OV_j}} \sum_{k=1}^{K} m_{i_{Ov_j},j,k}) + M^{occ}_j \\ & \leq M_j, j = 1..M, i_1, \ldots, i_{OV_j} \end{aligned} \tag{3.20}$$

Slot Schedulability Constraints: This constraint ensures that the total slots utilized by a vehicle at an edge in a channel are less than the total available slots for data delivery for that vehicle, considering the various time delays. This constraint further ensures that the total slots utilized by all the vehicles in an overlapping set are less than the max overlapping slots $n\_slot^{overlap\_max}_j$ of edge E_j.

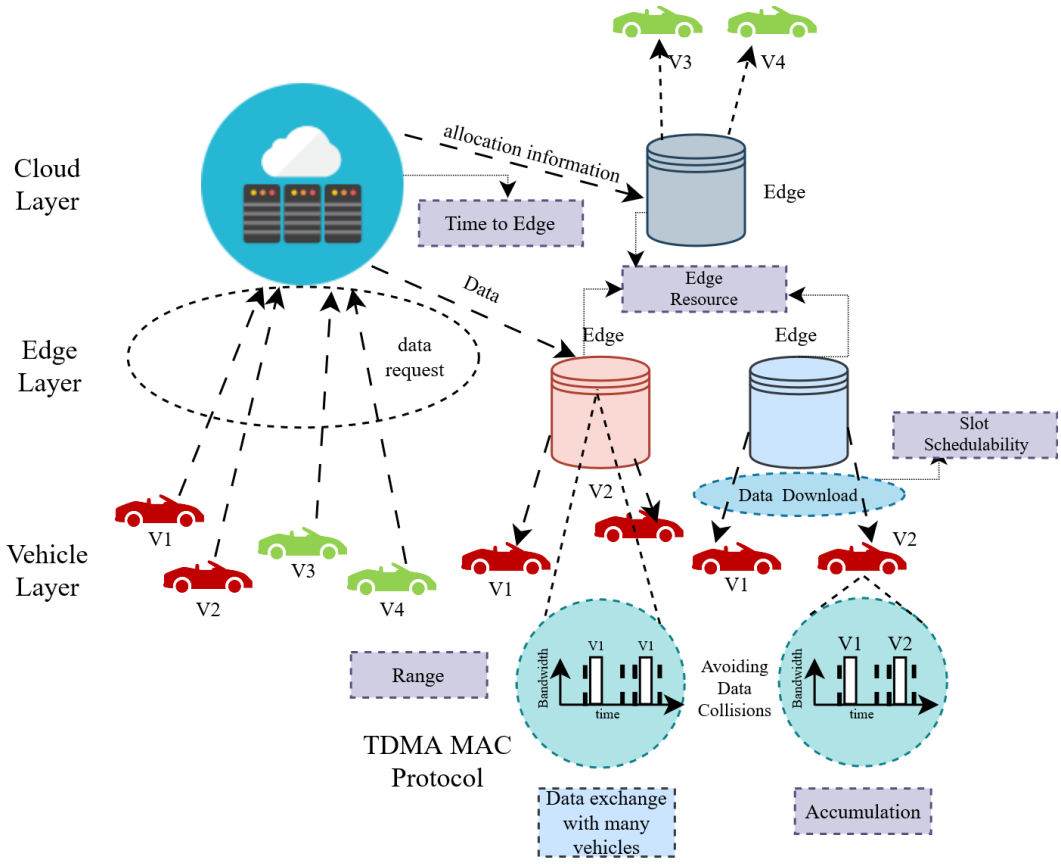


Figure 3.4: Three Tier Architecture with MAC Protocol and Data Delivery Constraints

$$\sum_{k=1}^{K} n\_slot_{i,j,k}^{alloc} \leq n\_slot_{i,j}^{max\_alloc}$$
$$\sum_{i \in S_{j,ov}} \sum_{k=1}^{K} n\_slot_{i,j,k}^{alloc} \leq n\_slot_{j}^{overlap\_max} \tag{3.21}$$
$$i = 1..N, j = 1..M, k = 1..K$$

3.3.3 Objective Function

Eq.(3.22) portrays the slots utilized for data delivery by vehicles in one overlapping set at an edge across all the channels for unique vehicles which belong to set $S_{j,ov}$ and S_j^{ov} is a set such that it doesn't contain $S_{j,ov}$. Eq.(3.23) calculates the slots used at an edge considering data delivery and those utilized for MAC protocol-specific metadata such as CF-Poll-frame and Acknowledgment frame for all the overlapping sets. In the current work, the objective of the optimization is to minimize the global slot utilization cost, which is computed based on a pricing mechanism used in [44]. This mechanism helps

balance the slots utilized across all the edges for data delivery. The global slot utilization cost uses a non-linear pricing policy, as mentioned in Eq.(3.24), where β is the cost factor.

$$slot_{i,ov}^{util\_data} = \sum_{i \notin \{i \in S_{j,ov} \cap S_j^{ov}\}} \sum_{k=1}^{K} \frac{n\_slot_{i,j,k}^{alloc}}{n\_slot_j^{overlap\_max}} \tag{3.22}$$

$$slot_j^{util} = \sum_{ov=1}^{OV_j} (slot_{i,j_{ov}}^{util\_data} + slot_{i,j_{ov}}^{util\_meta}) \tag{3.23}$$

$$slot_j^{util\_cost} = \beta * (1 + slot_j^{util})^2 \tag{3.24}$$

Finally, the objective function for the current work is given by Eq.(3.25)

$$minimize \sum_{j=1}^{M} slot_j^{util\_cost} \tag{3.25}$$

3.4 Experiments and Results

This section presents the practicality of the proposed collision-aware data delivery framework using two MAC protocols, VeMAC and HCCA, and compares it with a bandwidth optimal (OPT-NoMAC) case from [37] which does not consider a MAC layer. This section compares the three frameworks in terms of the maximum number of vehicles serviced and bandwidth cost. This work then presents the variations in global slot utilization cost by differing the traffic densities, edge counts, and vehicle counts by using a real-world traffic scenario from the Luxembourg dataset [57].

In the current work, K_j is set to 15, and values of $K_j^{jam}, m_i, L_j^{cov}, M_j, B_j, v_j^f$ are obtained from [44], the values for MAC-Header, Acknowledgment frame, and CF-Poll frame are obtained from [17]. The current experiments have been conducted on Intel(R) Xeon(R) CPU E5-2640-based server with 20 cores, each running at 2.6GHz. The results are as follows.

3.4.1 Maximum Vehicles Served

Figure 3.5 illustrates the maximum number of vehicles served for OPT-NoMAC, HCCA, and VeMAC approaches for various edge counts. The increased availability of extra computation and memory resources of the newer edges can explain the trend of serving more vehicles with an increasing edge count. It can be observed that HCCA has fewer vehicles served than the other two approaches, as the available time for data delivery is less due to the absence of the service channel slots during the control channel phase [58]. Due to this, the total available slots for data delivery are less numerous than those for VeMAC. At a lower edge count, VeMAC approaches OPT-NoMAC as the number of data frame collisions is lesser due to smaller overlapping vehicle sets. With the increasing edge count and vehicles, data frame collisions increase due to the larger number of overlapping vehicles contenting for data simultaneously.

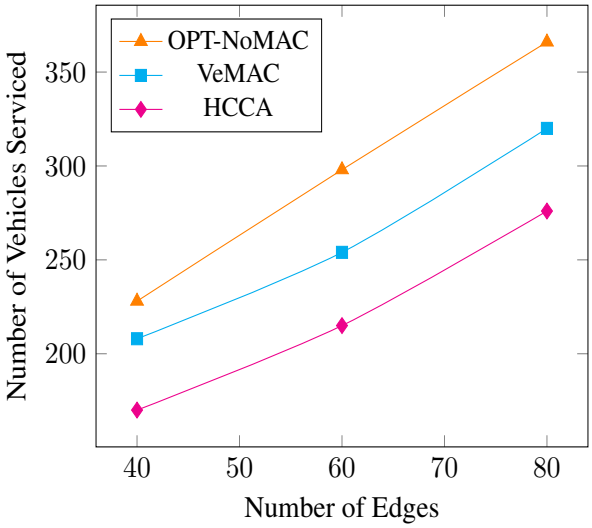


Figure 3.5: Maximum Vehicles Served

This can explain the decrease in the number of vehicles served by VeMAC compared to OPT-NoMAC at a higher edge count scenario.

3.4.2 Analysis of Slot Utilization Cost for various Densities

Figure 3.6a and 3.6b illustrate the variations in slot utilization cost for different vehicle densities, number of vehicles, and edge counts for HCCA and VeMAC protocols. Each plot corresponds to variations in global slot utilization cost for 70 and 80 edges scenarios. It can be observed that with increasing vehicle density values, there is decreasing trend in slot utilization cost. This can be explained by the increased coverage time $t_{i,j}^{cov}$ due to a decrease in the velocity of the vehicle at an edge, as mentioned in Eq(3.3). This increased coverage time increases the number of time slots for data delivery, allowing the vehicle to choose edges with lesser slot utilization costs.

It can be observed that HCCA has more slot utilization costs than VeMAC for various vehicle sizes. This is due to the differences in slot utilization cost for metadata which is MAC protocol dependent. HCCA data frame is associated with a CF-Poll frame and an Acknowledgment frame for each data slot [58], whereas the VeMAC data frame has an Acknowledgment frame only [19], thereby increasing the slot utilization cost for HCCA.

3.4.3 Analysis of Variation in Bandwidth Cost

Figure 3.7a and 3.7b illustrate the variation in bandwidth cost for OPT-NoMAC, VeMAC, and HCCA for 120 vehicles. The bandwidth cost equation is obtained from [37]. The MAC approaches have higher bandwidth costs due to the presence of metadata for collision avoidance, which involves transmitting extra information along with the vehicle's data. HCCA uses a CF-Poll frame and an Acknowledgment frame

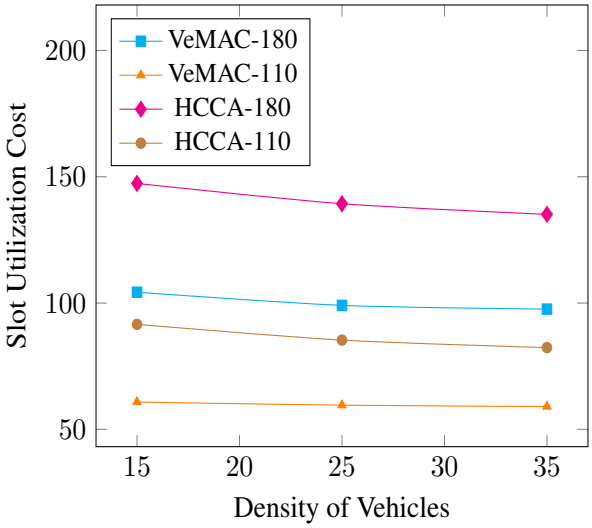
for collision avoidance and thus incurs more bandwidth cost than VeMAC as it uses an Acknowledgment frame, which explains the higher bandwidth cost for HCCA than VeMAC. With the increasing edge count, the average bandwidth cost per edge decreases due to data distribution across more edges. The total bandwidth cost for all three approaches increases with the increase in edge count, as a minimum bandwidth cost is incurred due to the presence of an edge and also due to data distribution across edges.

3.4.4 Analysis of Variation in Slot Utilization Cost

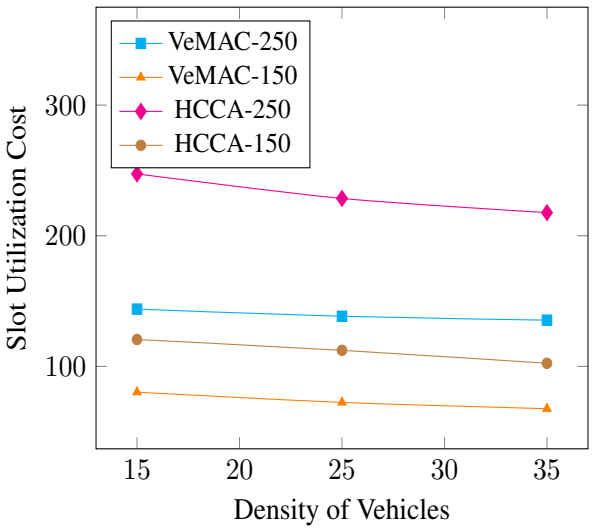
Figures 3.8a and 3.8b illustrate the variation in global slot utilization cost using VeMAC and HCCA for 120 vehicles. For VeMAC, the distribution in total slots utilized changed from a mean of $1.67e4$ and peak value of $1.34e5$ for 50 edges to a mean of $1.22e4$ and a peak of $1.24e5$ slots per edge for 80 edges case, which explains the decrease in mean slot utilization cost per edge. The increase in total slot utilization cost with edges can be explained due to the minimum cost incurred due to the addition of newer edges. Similarly, for the HCCA protocol, the distribution in slots utilized changed from a mean of $2.95e4$ and peak value of $2.35e5$ slots for 50 edges to a mean of $1.84e4$ and a peak of $1.89e5$ slots per edge for 80 edge cases. This considerable variation in slot utilization explains the decrease in average and total slot utilization costs for the HCCA protocol. VeMAC has a lower slot utilization cost than HCCA due to the presence of CF-Poll in metadata associated with the vehicle's data when compared to VeMAC.

3.5 Conclusion and Future Scope

The current chapter has proposed a collision-aware data delivery framework for connected vehicles via edges. This considers the practical scenario of multiple vehicles requesting data simultaneously from the edges which can result in data frame collisions due to parallel access. To prevent these collisions, this work has proposed an optimization framework that considers a TDMA MAC-based approach to handle data delivery to multiple vehicles simultaneously. The work has presented the practicality of the proposed work through several experiments by using two state-of-the-art MAC protocols VeMAC and HCCA against an optimal approach that doesn't consider the MAC layer. This work can be further extended by considering data delivery time, security, etc. in the objective function.

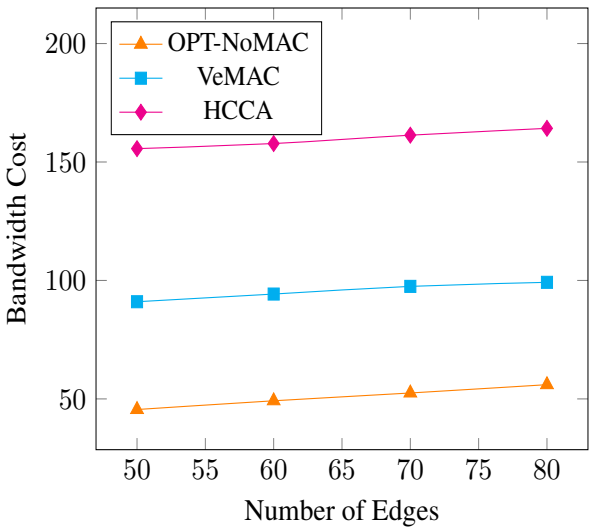


(a) M = 70

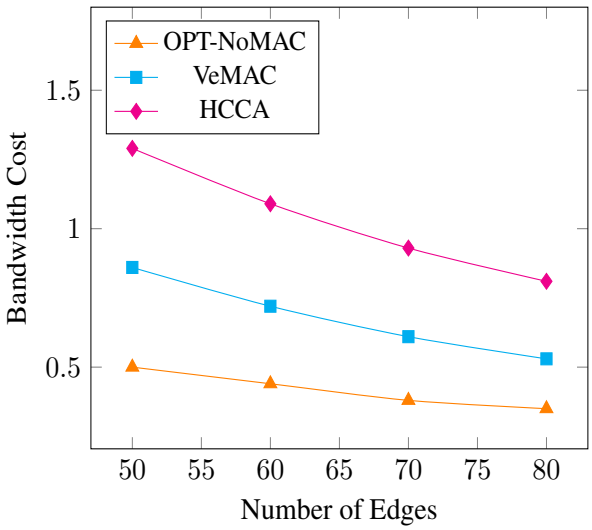


(b) M = 80

Figure 3.6: Variation in slot utilization costs for 70 and 80 edges

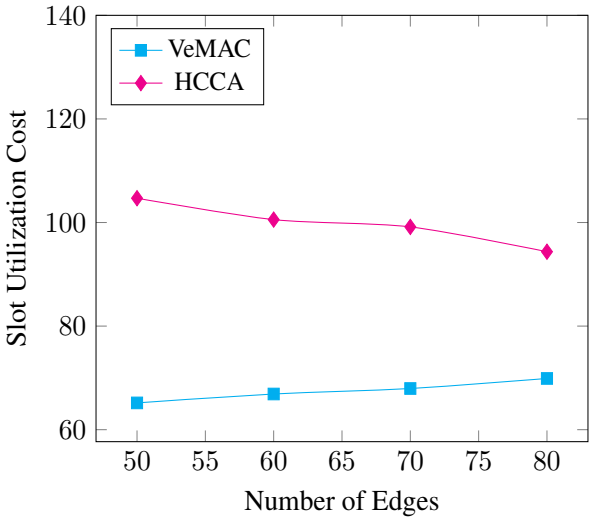


(a) Total BW Cost

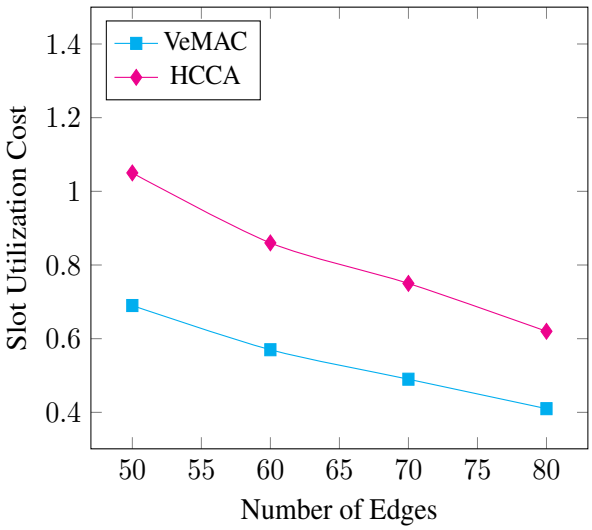


(b) Average BW Cost per Edge

Figure 3.7: Variation in Bandwidth (BW) cost



(a) Total Slot Utilization Cost



(b) Average Slot Utilization Cost

Figure 3.8: Variation in Slot Utilization Cost

Chapter 4

Price Optimal Secure Task Offloading Framework

4.1 Introduction

The advent of Connected Vehicles (CV) has revolutionized the concept of safety, travel, and user experience in a vehicle. The idea of CV involves intelligent vehicles exchanging data and services with network components such as Roadside units (RSUs), transportation hubs, and other vehicles over a short communication range. These systems typically use DSRC protocol in a Vehicular Edge Computing (VEC) environment for seamless data and service exchange [16]. A general VEC approach involves a Cloud, an Edge, and a Vehicular layer, where the vehicles receive the data and services from the edge layer to ensure low latency and bandwidth requirements. Initially, the concept of CV was proposed to exchange safety messages, road congestion information, and weather information to build an efficient and safer vehicle [12]. However, the promising capabilities of low latency and the capacity to handle bandwidth requirements have attracted various stakeholders to innovate CV-specific applications that involve task offloading from vehicles to suitable RSUs. Applications such as Driver Health Monitoring, Autonomous Driving, Vehicle Platooning, and infotainment services are a few such scenarios that can be potentially adopted through task offloading from vehicles to edges [32] [36].

Task offloading involves sending raw data to nearly RSUs, which process this data and return the results to the vehicles. Task offloading helps reduce the computational load on individual vehicles [45]. However, this data transfer can become vulnerable in a network-compromised scenario, compromising sensitive information to unauthorized users. Apart from the security concerns, task offloading also requires RSU resource management, as incorrect allocations can result in non-optimal utilization or increase the computation price, rendering the network financially infeasible [38].

Therefore, minimizing the total price incurred while considering the security requirements in the network is vital to ensure the affordability and safety of task offloading from vehicles to edges. In a realistic scenario, there can be many ways in which tasks can be offloaded to increase the total price and also over-utilize RSUs. Therefore, the current work minimizes the total price by using a pricing policy to reduce these non-optimal allocations in the network. The edge server, which handles more tasks, contributes more to the total price; based on this notion, a price-optimization framework is proposed,

which minimizes the total price incurred in the network while ensuring security, edge, and vehicle constraints vehicles.

The contributions of this work are as follows.

1. To the best of the knowledge, this is the first work that proposes a price-optimal framework for task offloading considering multiple vehicles, multiple edges, vehicular flow, resource requirements, and security.

2. This work presents the practical significance of the proposed framework by comparing it with two greedy approaches in terms of the total price factor.

3. This work considers five security classes that differ in confidentiality, integrity, and availability. This work presents the effect of price factor variations due to the security requirements for task offloading for various problem sizes.

4. This work presents the experiments considering real-world traffic scenarios from the Luxembourg dataset and evaluates the effects of security classes on a real-time edge device.

4.2 System Model and Problem Definition

This section presents the VEC system model, which consists of a central cloud server, edges, and vehicles, followed by the price optimization problem with security across all the edges.

4.2.1 VEC System Model

The VEC model includes a central cloud C connected to M heterogeneous edge servers $\{E_1, .., E_M\}$ and N vehicles $\{V_1, V_2, ..., V_N\}$ in the network. Each edge server has a finite coverage area for task offloading to vehicles passing through that edge. *The current work assumes that the CV offloads the task to one edge server only*. This work models the VEC system using Vehicle, Edge, Flow, Timing, Security, Cloud, and Resource Utilization models.

Vehicle Model: The vehicle model for vehicle V_i $(1 \leq i \leq N)$ is described as a tuple V_i : $\{x_{i,j}, d_i, ts_i\}$, where $x_{i,j}$ contains the route information, and d_i is the input memory of the task ts_i requested by V_i. Considering a single-lane scenario, if the vehicle V_i passes through edge E_j $(1 \leq j \leq M)$, then $x_{i,j}$ is 1. If the vehicle V_i doesn't pass through the edge, then the value of $x_{i,j}$ is 0.

Edge Model: The Edge Model for edge E_j is described as $\{M_j, M_j^{occ}, L_j^{cov}, P_j, B_j, P_j^{occ}, C_j, C_j^{main}\}$, where $(1 \leq j \leq M)$. Here M_j is the total memory of the edge, M_j^{occ} is occupied memory for processing, L_j^{cov} is the length of coverage region, B_j is the total bandwidth of the edge, P_j is the total number of Virtual Machines (VMs), P_j^{occ} is the total number of VMs used for maintenance and C_j is the price incurred in cents per VM per second for processing a task and C_j^{main} is the price incurred cents per VM per second for maintaining the RSU.

Flow Model: The velocity $v_{i,j}$ of vehicle V_i $(1 \leq i \leq N)$ at edge E_j $(1 \leq j \leq M)$ is calculated using the Greenshields Eq.([44]). $v_{i,j}$ depends on free flow velocity v_j^f, vehicle density k_j, and jam density of vehicles k_j^{jam} at edge E_j. These three quantities are related as shown in Eq.(4.1).

$$v_{i,j} = v_j^f * (1 - \frac{k_j}{k_j^{jam}}) \tag{4.1}$$

Timing Model: The timing model describes the various timing components of the current work. $t_{i,j}^{trav}$ is earliest travel time for vehicle V_i $(1 \leq i \leq N)$ to reach edge E_j $(1 \leq j \leq M)$. $t_{i,j}^{comm}$ is the time central cloud C takes to send vehicle data V_i to edge E_j. t^{slot} is the period of each slot in a channel k $(1 \leq k \leq K)$. $t_{i,j}^{cov}$ is the coverage time of vehicle V_i at edge E_j. The relation between $t_{i,j}^{cov}$, l_j^{cov}, $v_{i,j}$ is given by Eq.(4.2). $t_i^{service}$ is the time taken to process the task ts_i of V_i, and $t_i^{security}$ is the time taken to process the security requirements of task ts_i.

$$t_{i,j}^{cov} = \frac{l_j^{cov}}{v_{i,j}} \tag{4.2}$$

Security Model: In the current work, a security class s_i is assigned to a vehicle V_i depending upon the nature of the task. Here, security class refers to a specific level of the security triads: confidentiality, integrity, and availability. A higher security class is highly resilient to attacks but can incur more computation costs. The current work adopts the analysis by Architecture Reference for Cooperative and Intelligent Transportation (ARC-IT) [59] for security classes. Each security triad has three levels: low, medium, and high. Thereby, there are potentially 27 different device security classes, of which 5 security classes, as mentioned in Table 4.1, are chosen as candidates for assigning to the CV tasks as presented in [60].

Security Class	Confidentiality	Integrity	Availability
1	Low	Moderate	Moderate
2	Moderate	Moderate	Moderate
3	Moderate	High	Moderate
4	High	High	Moderate
5	High	High	High

Table 4.1: Security Classes from ARC-IT

The data d_i of a vehicle V_i is initially secured with a class 1 security before sending it to the RSU for processing. The received data is first decrypted and verified before processing. The processed memory r_i is secured with the security class depending on task ts_i before returning it to the V_i. The variable q_i refers to the memory required for processing the security requirements of the task ts_i.

Cloud Model: The Central Cloud C processes the information about the task ts_i and d_i of vehicle V_i and assigns the necessary security class s_i class. Further, the cloud calculates the required VMs p_i, the time to process the task $t_i^{service}$, and the time to process the security requirements $t_i^{security}$.

Resource Utilization Model: The resources of an edge are utilized for storage and computation of tasks offloaded by the vehicles. Therefore, at any instant, the memory usage and the total VM usage should not exceed the total memory capacity and the VMs P_j available at an edge. In the current work, we adopt the Earliest Time of Arrival (ETA) and Latest Time of Departure (LTD) from [37] for calculating the overlapping set of vehicles to determine the total VMs and memory utilized. ETA refers to the time spent by a vehicle to reach an edge when it travels with minimum velocity. Similarly, LTD refers to the time spent to reach an edge when the vehicle travels with maximum velocity. For a vehicle, V_i $(1 \leq i \leq N)$, the maximum and minimum velocities at edge E_j $(1 \leq j \leq M)$ based on Eq.(3.2) are given by Eq.(4.3) and Eq.(4.4), where k^{min} and k^{max} refer to minimum and maximum densities at edge E_j. Using these equations, the ETA and LTD are computed using equations 4.5 and 4.6, as mentioned in [37], where Q is the set of edges the vehicle has passed through and $R_{m,n}$ is the road segment between the edges m and n.

$$v_{i,j}^{max} = v_j^f * (1 - \frac{k^{min}}{k_j^{jam}}) \tag{4.3}$$

$$v_{i,j}^{min} = v_j^f * (1 - \frac{k^{max}}{k_j^{jam}}) \tag{4.4}$$

$$t_{i,j}^{eta} = \frac{A_i}{v_{i,j}^{max}} + \sum_{m,n \in Q} \frac{R_{m,n}}{v_{i,j}^{max}} + \sum_{o \in \{Q - E_j\}} \frac{L_o^{cov}}{v_{i,o}} \tag{4.5}$$

$$t_{i,j}^{ltd} = \frac{A_i}{v_{i,j}^{min}} + \sum_{m,n \in Q} \frac{R_{m,n}}{v_{i,j}^{min}} + \sum_{o \in Q} \frac{L_o^{cov}}{v_{i,o}} \tag{4.6}$$

In equations 4.5 and 4.6, the first term signifies the time the vehicle takes to reach the first edge from its starting point, where A_i is the distance between the vehicle's starting point and the first edge. The second term computes the time the vehicle travels between the edges, and the last term accounts for the time spent passing through coverage regions of the edges. The overlapping vehicle sets S_j of an edge are computed using the ETA and LTD values based on algorithm 1 in [37].

4.2.2 Decision Variable

In the current work, a binary decision variable $serv_{i,j}$ indicates if a vehicle V_i offloads a task to edge E_j. If V_i is not in the path of E_j then $serv_{i,j} = 0$

4.2.3 Problem Formulation

The current work builds a price optimization framework for task offloading from vehicles to edges, considering security and resource availability constraints. The frameworks run in the cloud layer to allocate a suitable RSU to the vehicle to minimize the total price incurred while ensuring security.

4.3 Optimization Framework

This section first discusses the constraints necessary for task offloading from vehicles to edges, followed by the objective function for optimizing the total price incurred in the network.

4.3.1 Optimization Constraints

The various constraints for the price optimal framework are detailed below.

Service Accumulation Constraint: This constraint ensures that the summation of $serv_{i,j}$ over all the edges equals 1. This ensures the vehicle offloads the task to one edge. The constraint is as mentioned in Eq.(4.7).

$$\sum_{j=1}^{M} serv_{i,j} * x_{i,j} = 1, i = 1..N, j = 1..M \tag{4.7}$$

Edge resource Constraints: The Edge resource constraints ensure that the memory and total VMs utilized at an edge at any instant are less than the total available resources of the edge. In the current work, we consider the overlapping sets of vehicles to calculate the resources utilized at an edge by different vehicles. Using this constraint, we can ensure that the memory and the VMs used for task-offloading plus the resources used for maintenance are always less than the total resources of the edge at any given instant. if $\{S_{j,1}, ..., S_{j,ov_j}\}$ are overlapping sets of edge E_j, then the edge resource constraint can be given as Eq.(4.8) for memory and Eq (4.9) and for VMs.

$$\max(\sum_{i \in S_{j,1}} (serv_{i,j} * (d_i + q_i + r_i)), ..., \sum_{i \in S_{j,ov_j}} (serv_{i,j} * (d_i + q_i + r_i))) + M_j^{occ} \leq M_j, \quad i = 1..N, j = 1..M \tag{4.8}$$

$$\max(\sum_{i \in S_{j,1}} (serv_{i,j} * p_i), ..., \sum_{i \in S_{j,ov_j}} (serv_{i,j} * p_i)) + P_j^{occ} \leq P_j, i = 1..N, j = 1..M \tag{4.9}$$

Bandwidth Schedulability Constraints: The bandwidth schedulability constraint from [44] is modified to handle the security computation time for ensuring the security of the task offloaded by vehicle

V_i to edge E_j. The term $D_{i,j}^{min,serv}$ is the minimum number of bytes that can transferred during the transit to a vehicle and is given by Eq.(4.10). Eq.(4.11) gives the bandwidth schedulability constraint.

$$D_{i,j}^{min,serv} = \frac{B_j \times \left(\frac{L_j}{v_{u,j}} - t_i^{service} - t_i^{security}\right)}{k_j^{jam} \times L_j} \tag{4.10}$$

$$serv_{i,j} * (d_i + r_i) \leq D_{i,j}^{min,serv} \tag{4.11}$$

4.3.2 Objective Function

Eq.(4.12) portrays the total price incurred at edge E_j due to task offloading. Eq.(4.13) is the total price incurred over a time window t^{window}. The summation of $price_j^{task}$ and $price_j^{maintenance}$ gives the total price incurred $price_j^{incurred}$ at E_j as given in Eq.(4.14).

In the current work, the objective of the optimization is to minimize the total price factor, which is computed based on a computation mechanism used in [44]. This approach helps balance the price utilized across all the edges for task offloading. The Price Factor (PF) calculation uses a non-linear pricing policy, as mentioned in Eq.(4.15), where β is the cost factor.

$$price_j^{task} = \sum_{i=1}^{N} (serv_{i,j} * P_i * C_j * (t_i^{service} + t_i^{security})) \tag{4.12}$$

$$price_j^{maintenance} = P_j^{occ} * C_j^{main} * t^{window} \tag{4.13}$$

$$price_j^{incurred} = price_j^{task} + price_j^{maintenance} \tag{4.14}$$

$$pf_j = \beta * (1 + price_j^{incurred})^2 \tag{4.15}$$

Finally, the objective function for the current work is given by Eq.(4.16)

$$minimize \sum_{j=1}^{M} pf_j \tag{4.16}$$

Eq.(4.16) ensures that all the vehicles can offload the task with suitable security requirements to the RSUs before they reach the destination.

4.4 Experimental Setup

This section presents the hardware and software components used in the current work. The description is as follows

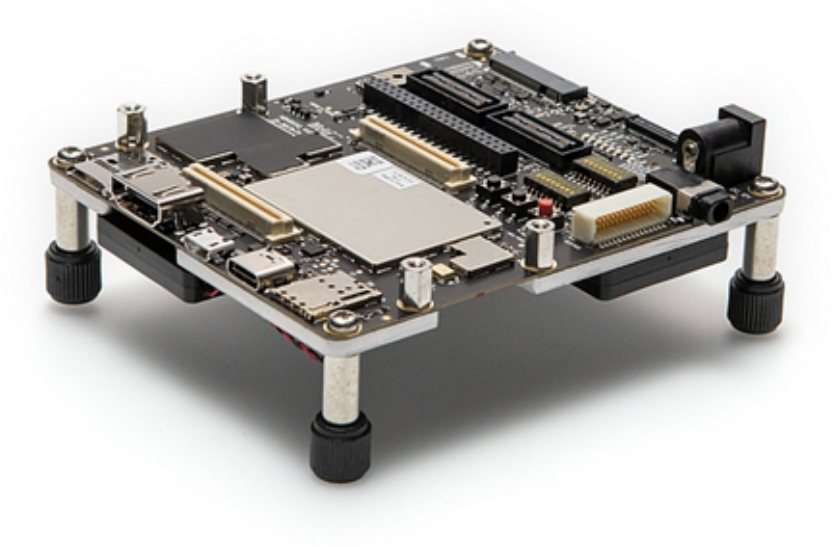


Figure 4.1: Qualcomm Snapdragon 8 Gen 2 Mobile Hardware Development Kit

4.4.1 Hardware Components

In the current work, the Qualcomm Snapdragon 8 Gen 2 mobile hardware development kit as shown in Figure 4.1 is used as an edge device to simulate the computation time needed for processing security algorithms. The device runs on the Android 13 operating system with 6 GB RAM, a 2.8GHz processor, and with dedicated GPU and DSP processor [61]

4.4.2 Software Components

Optimizer: In the current work, CVX programming in Matlab for convex programming. CVX allows constraints and objectives to be specified using standard Matlab expression syntax.

Solver: In the current work, Gurobi optimization is used to solve the binary optimization problem of minimizing the total price for task offloading with security [62].

Cryptography: The cryptography python module is used as a baseline reference to simulate the necessary encryption and hashing algorithm in the current work. Python cryptography module includes general recipes such as symmetric ciphers, message digests, and derivation functions [63].

4.4.3 Translating Security Classes to Algorithms

The United States National Institute of Standard and Technology (NIST) [64] presents 13 hashing algorithms and 4 encryption algorithms that can ensure security. The current works adopt the best 3 encryption algorithms and hashing each as a suitable candidate to address the security requirements for

task offloading. Table 4.2 presents the mapping for low, moderate, and high levels of confidentiality, integrity, and availability to their algorithmic equivalents. In the current work, high availability refers to a scenario where the entire bandwidth is available for data exchange, and a moderate level refers to a scenario with 90 % of bandwidth and a low level refers to 50 % of bandwidth being available for data exchange. Further, the table also presents the data processing rate of the algorithms in KB/ms on the Qualcomm Snapdragon 8 Gen 2 Mobile Hardware Development Kit.

S.No	**Security Triad**	**Security Level**	**Algorithm**	**Run Time (KB/ms)**
1	Confidentiality	Low	AES-128	1306.471
		Moderate	AES-192	1030.301
		High	AES-256	808.574
2	Integrity	Low	SHA3-256	606.32
		Moderate	SHA3-384	530.411
		High	SHA3-512	347.119
3	Availability	Low	0.5*BW	N/A
		Moderate	0.9*BW	N/A
		High	1.0*BW	N/A

Table 4.2: Algorithmic Equivalents of Security Triad levels

Considering the algorithms from Table 4.2, Table 4.1 translates to Table 4.3

Security Class	**Confidentiality**	**Integrity**	**Availability**
1	AES-128	SHA3-384	0.9*BW
2	AES-192	SHA3-384	0.9*BW
3	AES-192	SHA3-512	0.9*BW
4	AES-256	SHA3-512	0.9*BW
5	AES-256	SHA3-512	1.0*BW

Table 4.3: Security Classes and their Algorithms

4.5 Experiments and Results

This section evaluates the proposed price optimization framework for various security classes. The first section presents the capability of the proposed framework to minimize the total price factor in comparison with two greedy approaches followed by an evaluation of the variation in total price factor for variation in security classes and problem sizes by using a real-world traffic scenario from the Luxembourg dataset [57].

In the current work, β is set to 1000, The values of P_j and C_j are as shown in Table 4.4, P_j^{occ} takes a value between 1 and 3 VMs, p_i takes a value between 3 and 4. VMs, d_i takes a value between 4 and 9, $t_i^{service}$ takes a value between 1 and 5. C_j^{main} is set to 0.001 times C_j , the values of $K_j, K_j^{jam}, L_j^{cov}, M_j, M_j^{occ}, B_j, v_j^f$ are obtained from [44].

S.No	**Total Number of VMs** (P_j)	**Price/VM/sec** (C_j)
1	8	0.014
2	16	0.029
3	32	0.058

Table 4.4: VM configurations and the price in cents

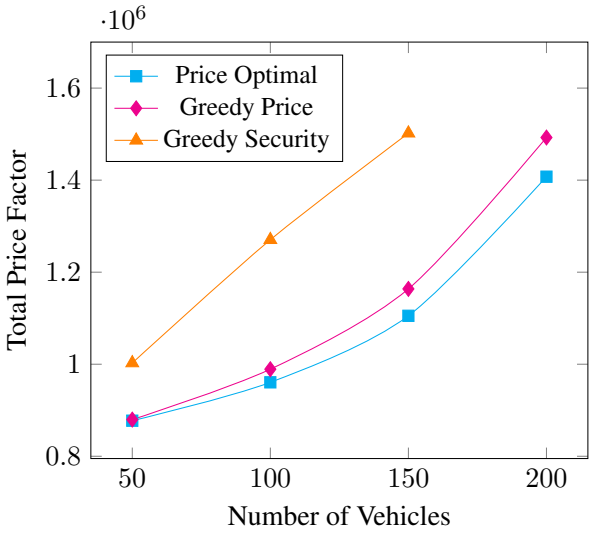


Figure 4.2: Capability of Proposed Optimization framework

4.5.1 Capability of Proposed Framework

Figure 4.2 presents the variations of the total price factor for an increasing number of vehicles in the network for 80 edges. The figure presents the efficiency of the proposed framework in comparison to two greedy algorithms namely greedy price and greedy security. It can be observed that the proposed framework has a lower total price factor in comparison to the two algorithms.

Greedy price algorithms allocate resources to vehicles with the highest resource requirements first. The algorithm always chooses an edge that has the least increase in total price factor, Whereas the greedy security algorithm allocates to vehicles with the highest security requirement first while choosing a random edge for allocation. It can be observed that the greedy price algorithm follows a closer total price for a lower vehicle count which can explained by similarity in the objective of allocation, which later deviates due to its heuristic nature resulting in sub-optimal allocations. On the other hand, the greedy security algorithm has an overall higher total security price due to random allocations. Further, it is unable to serve 200 vehicles due to its non-optimal nature in terms of the total price factor.

S.No	Security Class	Additional Vehicles	Total Price Factor
1	Class 1	50	960642.91
		100	1106319.36
		150	1405257.09
2	Class 2	50	960662.09
		100	1106452.66
		150	1405524.96
3	Class 3	50	960832.52
		100	1107068.30
		150	1407035.76
4	Class 4	50	960890.87
		100	1107230.16
		150	1407446.65

Table 4.5: Variation in Price for various class and problem sizes

4.5.2 Analysis of Price Factor vs Security Class and Vehicle Count

Table 4.5 presents the variation in the total price for 4 security classes (as class 5 uses the same security algorithms as class 4) and problem sizes for 80 edges. In this experiment, 10 vehicles from each

class were chosen as a base scenario (In total 50 vehicles for all five security classes). Keeping the base scenario constant; the total price factor is tabulated for various security classes and problem sizes.

It can be observed that there is a non-linear increase in the total price for every addition of 50 new vehicles, which can be explained by the load-balancing effect of the quadratic policy on the total price factor. Across the security classes, the total price factor increases for the corresponding problem sizes due to the usage of better security algorithms, which incurs more computation resources.

4.6 Conclusion and Future Scope

In the current work, a price-optimal task offloading framework with security is proposed for affordable and safer task offloading. This scenario considers 5 security classes in terms of the security triads: confidentiality, integrity, and availability, which can be applied to various applications to ensure security in task offloading. Apart from security, the framework minimizes the total price incurred in the network to ensure affordability for task offloading. The work presents the capability of the proposed framework by comparing it with two greedy algorithms in terms of the total price factor. Further, the work evaluates the variation in the total price factor for various security classes and problem sizes. This work can be further extended by considering power implications and task deadlines as a multi-objective function in the future.

Chapter 5

Other Research Contribution

This chapter describes an additional research contribution conducted in parallel with mainstream research. The work *A Comprehensive Evaluation on the Impact of Various Spoofing Scenarios on GPS Sensors in a Low-Cost UAV* identifies 16 GPS spoofing scenarios on a low-cost UAV. This work considers various settings where a UAV can be deceived with an incorrect GPS location during normal operation. The work is elaborated further.

5.1 Introduction

Unmanned Aerial Vehicles (UAVs), particularly low-cost UAVs, have become increasingly important due to their wide range of applications and ease of use. However, with the rapid growth of the UAV market, the rising security concerns pose a greater risk. One such primary concern is GPS location spoofing attacks, which can compromise a UAV's navigation system, making it crucial to analyze these attacks [65]. GPS signals are vulnerable to various forms of interference and attacks, which can compromise the accuracy and reliability of the signals. Two of the most effective cyber security attacks are spoofing and jamming [66]. Spoofing attacks involve the transmission of artificial GPS signals by unauthorized personnel that can deceive the UAV's GPS receiver, blocking factual information on its location and movement. In contrast, jamming attacks involve the transmission of noise signals that can interfere with authentic GPS signals, rendering them unavailable or unreliable. A Software Defined Radio (SDR) can generate various artificial GPS signals, potentially compromising the UAV system [67].

Considering a UAV's dynamic and varying operational conditions, it is crucial to identify and study the spoof signal characteristics and other GPS parameters, such as the number of satellites in view and the precision of measurements in various scenarios [68]. This exploration can significantly contribute to understanding the effects of GPS spoofing on a low-cost UAV in diverse operational settings, which can help in designing efficient anti-spoofing algorithms [69].

The research contributions of this work are as follows.

- Identified and evaluated 16 potential spoofing scenarios considering various environmental conditions such as indoor/outdoor, spoofing methods such as static/dynamic, and the spoof signal's

propagation paths such as Line of Sight (LOS) and Non-Line of Sight (NLOS) for low-cost UAVs using the multi-SDR-based spoofing technique.

- Conducted trials to capture and visualize the variations in several GPS-related parameters such as Horizontal Dilution Of Precision (HDOP), Vertical Dilution Of Precision (VDOP), and the changes in GPS satellites in view under normal conditions, jamming conditions, and jamming with spoofing conditions.
- Presented the real-time characteristics of the signal-to-noise power density of the spoof signal over distance for each operational condition.
- Analyzed the relation between maximum spoofable distance (deviation that can be induced in a UAV's location) to satellite count

5.2 Spoofing Approach and Scenarios

In the current work, multi-SDR-based GPS spoofing combines spoofing and jamming techniques to examine various spoofing scenarios. The overall system architecture and the different spoofing scenarios are elaborated below.

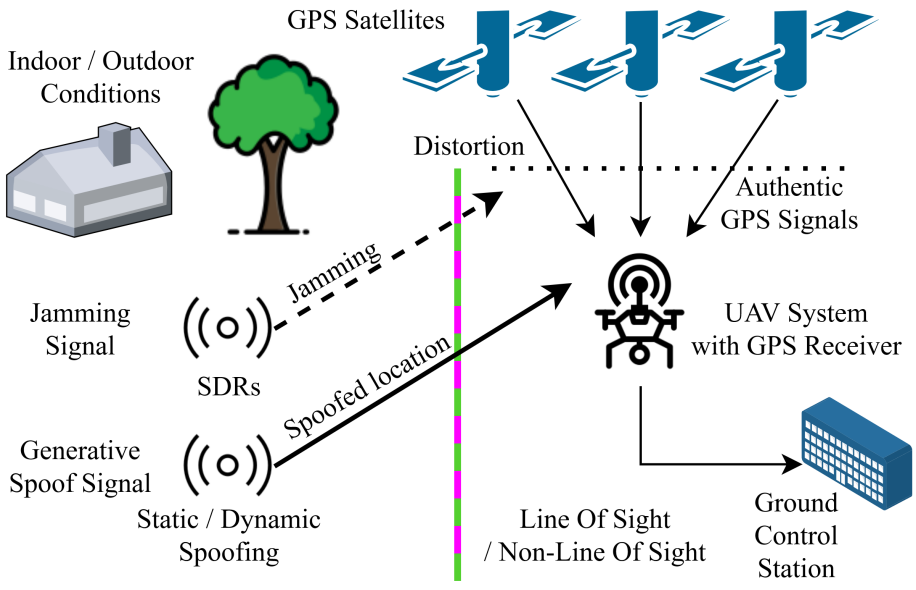


Figure 5.1: System Architecture for GPS Spoofing

Figure 5.1 illustrates the overall system architecture with various UAV operational conditions and spoof signal scenarios considered in the current work. The location spoofing process involves a multi-SDR-based approach to simultaneously distort the authentic GPS signals through jamming and generate

spoof signals for location manipulation through generative spoofing. Generative spoofing involves spoof signal generation with the same structure as authentic GPS signals [70].

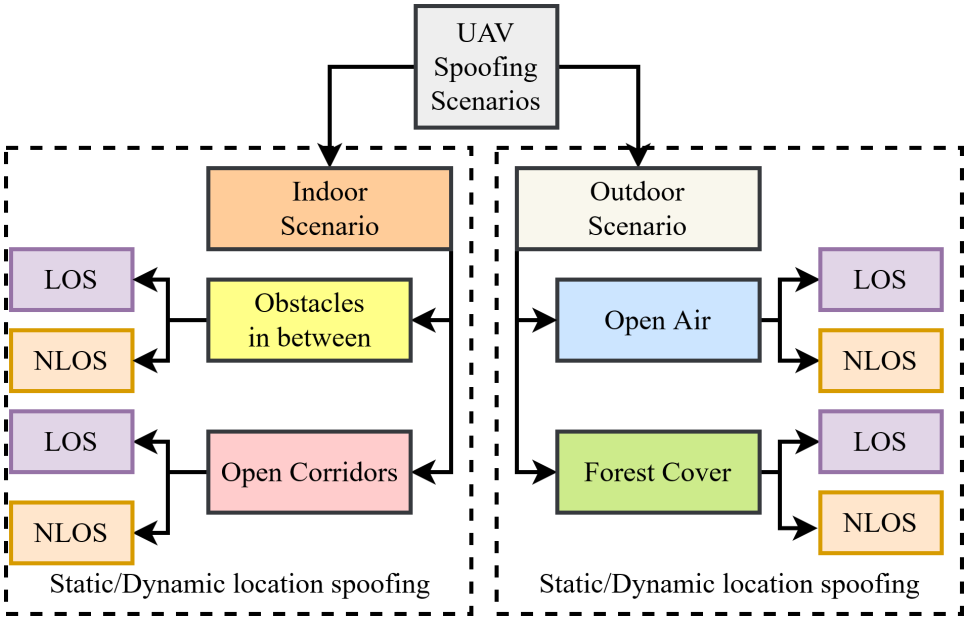
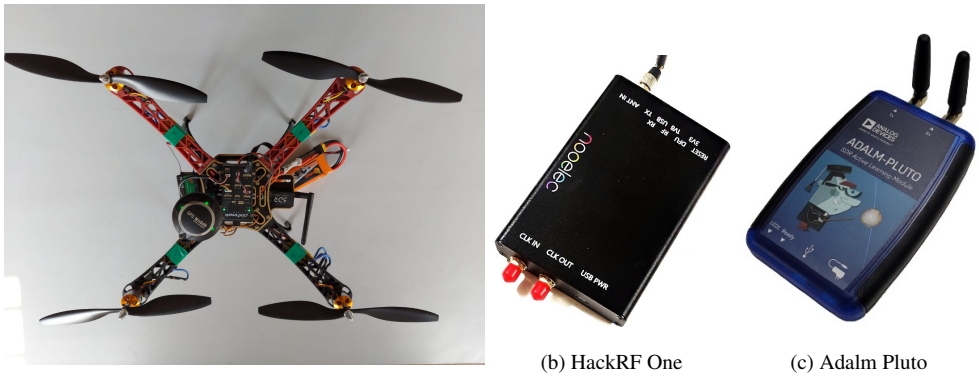


Figure 5.2: Location spoofing scenarios of a UAV

Figure 5.2 illustrates the various spoofing scenarios identified and analyzed in the current work. These scenarios can be categorized based on spoof location information, environmental conditions, and propagation path.

5.3 Experimental Setup



(a) Custom Built UAV System with Neo M8N GPS

(b) HackRF One SDR

(c) Adalm Pluto SDR

Figure 5.3: Hardware Components in the Experimental Setup

The UAV platform considered in this work is a custom-built quadcopter and can perform autonomous and other operation modes. Figure 5.3a displays the different onboard components of the UAV system. PixHawk 2.4.8 [71] is used as the flight controller. QGround-Control, an open-source ground control system, is adapted for analyzing the UAV state and for control. The onboard components include a Radio Telemetry module that uses the 433MHz band for control information exchange with the ground station. PixHawk provides various sensors capable of sensing positional information through the integrated IMU and magnetometer. The additional onboard sensors include a UBlox Neo M8N GPS Receiver [72] [73] for location reliability. Neo M8N has a navigation update rate of 18Hz with a position accuracy of 2 meters. This sensor uses the L1 GPS band for the location with a center frequency of 1575.42 MHz.

The current spoofing architecture involves the generation of spoof signals and jamming using an SDR. An accurate internal crystal oscillator needs to be used to generate a spoof signal. On the other hand, the jamming signal's primary purpose is to corrupt the entire L1 band with a high-energy noise. As shown in Figure 5.3b, and 5.3c, HACKRF One SDR with a nearly accurate temperature-controlled oscillator (TCXO) with a crystal error of less than 0.5 ppm is used as the source for the spoof signal and Adalm Pluto, a general-purpose SDR without any TCXO is being used. GPS-SDR-SIM [74], an open-source signal generation tool, is used for spoofing and jamming signal generation.

5.4 Experiments and Results

5.4.1 Indoor and Outdoor Static Spoofing Experiments

Figures 5.4a, 5.4b, 5.4c, and 5.4d portray the variation of GPS parameters for four static indoor spoofing scenarios, namely open corridor LOS (OC-LOS), NLOS (OC-NLOS), obstacles in between LOS (OB-LOS), and NLOS (OB-NLOS). These scenarios are analyzed under normal, jamming, and jamming with spoofing conditions. Similarly, Figures 5.4e, 5.4f, 5.4g, and 5.4h portray four static outdoor conditions, namely open air LOS (OA-LOS), NLOS (OA-NLOS), and forest cover LOS (FC-LOS), NLOS (FC-NLOS), under different scenarios.

In the figures 5.4a, 5.4b, 5.4e, and 5.4f, the decrease in precision can be explained by increased HDOP and VDOP values due to jamming which later improves in jamming with spoofing case due to GPS fix on spoofing signals. The considerable increase in GPS count and CN0 values in figures 5.4c, 5.4g, 5.4d, and 5.4h can be explained by the replacement of weaker GPS signal by jamming and spoofing signals in indoor scenarios, unlike outdoor scenarios.

Figures 5.4c and 5.4g illustrate the nature of LOS and NLOS scenarios. The decreasing trend is due to the blocking of weak energy signalsSpoofing signals are replaced by spoofing signals, thereby increasing the GPS count. However, in the outdoor scenario, The decrement in GPS count can be explained by the non-availability of weaker signals in NLOS conditions.

Figures 5.4k and 5.4l illustrate normalized deviations spread in GPS parameters of jamming with the spoofing setting concerning the normal setting. The deviation in each parameter can be calculated

using (5.1), where $param_{js}$ is a parameter observed in jamming with spoofing setting and $param_n$ is the corresponding parameter value observed in the normal setting. The difference between these values is divided by the maximum observed value to get the normalized parameter deviation. The high deviation in GPS count and average CN0 can explain the observed deviations in the indoor scenario. The deviation in the outdoor scenario can be explained by the considerable deviation in HDOP and VDOP values. It can be observed that each scenario has unique variations in deviations, thereby indicating that each scenario differs considerably.

$$param_{deviation} = \frac{param_{js} - param_n}{\max(param)} \tag{5.1}$$

5.4.2 Analysis of CN0 Variation

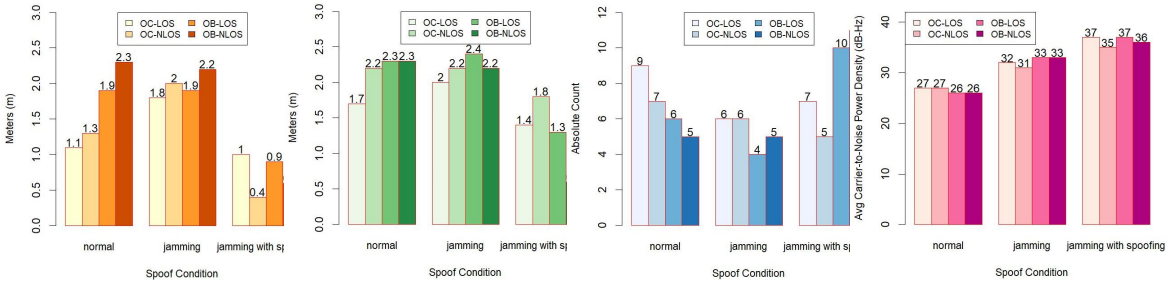
Figure 5.4i illustrates the variation in CN0 with the distance for indoor conditions. It has been observed that indoor OB-LOS has a lesser drop in CN0 with distance followed by OC-LOS, OB-NLOS, and OC-NLOS. This can happen due to signal propagation and reflections. The increase in CN0 in scenarios with obstacles can be possible due to multi-path reflections of the signals.

5.4.3 Analysis of Effective Spoofable Distance

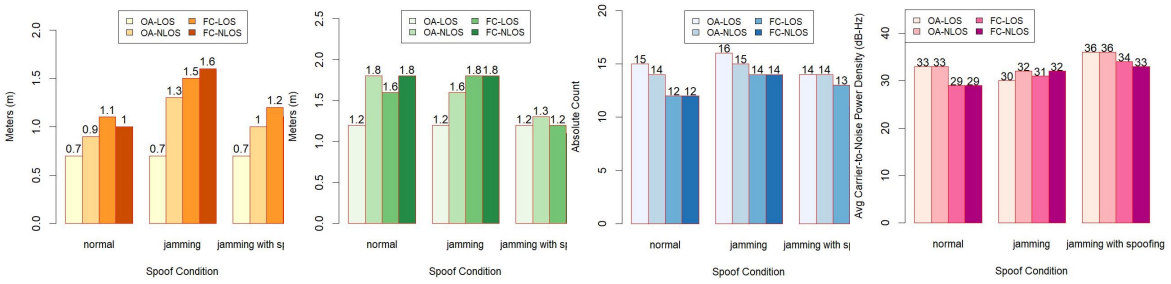
Figure 5.4j shows the relation between the effective spoofable distance and absolute satellite count observed across all experiments. The deviation in location is observed to be resilient as long as the satellite count is below 10. This behavior can be explained by the number of satellites needed for a stable GPS fix. The effective spoofable distance decreases with an increase in GPS count due to the high possibility of attaining a lock on original signals.

5.5 Conclusion and Future Scope

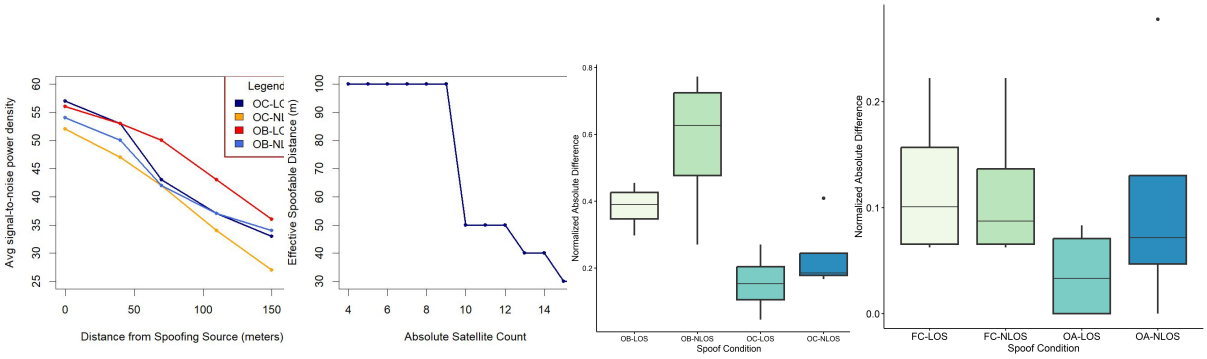
Low-cost UAVs have opened up a wide range of possibilities for civilian and commercial applications due to their affordability and ease of use. However, as low-cost UAVs become more prevalent, the need for security becomes vital due to the prevalence of cyber security attacks. GPS spoofing is a type of attack in which a malicious actor transmits false GPS signals to a UAV, causing it to believe it is in a different location than it is, thereby potentially crashing or even being commandeered by the attacker. Therefore, it is crucial to analyze various spoofing scenarios to understand the nature of these attacks. In the current work, we have identified 16 such scenarios and analyzed them in terms of GPS parameters such as Horizontal Dilution Of Precision (HDOP), Vertical Dilution Of Precision (VDOP), GPS count in view, and average signal-to-noise power density (CN0). Further, we have also analyzed the variation of spoofing distance with satellite count. and the variation of CN0 with distance. This work can be further extended to identify the critical zones to build an efficient anti-gps spoofing algorithm.



(a) Variations in HDOP (b) Variations in VDOP (c) Variation in GPS count (d) Variation in CNO



(e) Variations in HDOP (f) Variations in VDOP (g) Variation in GPS count (h) Variation in CNO



(i) Variations in CN0 (j) Variation in Spoof Distance (k) Indoor Conditions (l) Outdoor Conditions

Figure 5.4: Analysis of various indoor and outdoor scenarios

Chapter 6

Conclusion and Future Work

With rapid technological advancement, the Connected Vehicles (CV) paradigm is drastically transforming the automotive industry, enabling efficient data and service delivery to vehicles via edges. Realizing CV technology involves Vehicular Edge Computing (VEC), where the computation resources are brought much closer to the users to tackle latency and bandwidth requirements. However, VEC technology faces challenges, such as mobility, resource allocation, security, affordability, computation delay, scalability, power consumption, caching, etc. Considering a few such critical challenges, this work bridges the gap in realizing CV technology through two contributions.

The first contribution presents a data-frame collision-free optimization framework by adopting a time-slot-based MAC layer strategy that uses slot assignment to ensure collision-free data delivery for multiple vehicles across various transmission channels at each edge in different test conditions. The second contribution presents a price optimization framework for task offloading from CVs. This framework reduces the overall price for realizing the network, making it affordable while considering various task-specific security requirements.

As the integration of CV evolves, it becomes crucial to continuously enhance the VEC-based approaches for CV services. The current work can be extended in a multitude of directions considering various VEC challenges for data delivery and task offloading. One such problem can improve resource utilization through optimized caching. For, instance vehicles moving in a platoon or groups can potentially have similar data or task requirements. These requirements can be potentially analyzed and grouped in suitable RSUs to reduce data redundancy. Apart from caching one critical aspect that can also be considered is the power efficiency of RSUs as energy is the critical driving force in a battery/solar-powered RSU environment. Apart from these practical challenges, the near practical efficiency of the current work can be evaluated by simulation through digital twins. This approach can involve building a virtual VEC and CV environment to understand and identify the potential deployment issues thereby minimizing the chances of in-field failures of the systems.

Related Publications

- **SVSLN Surya Suhas Vaddhiparthy**, Joseph John Cherukara, Deepak Gangadharan, and BaekGyu Kim. "Collision-Aware Data Delivery Framework for Connected Vehicles via Edges" In 2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall), pp. 1-7. IEEE, 2023.
- **SVSLN Surya Suhas Vaddhiparthy**, Joseph John Cherukara, Deepak Gangadharan, and BaekGyu Kim. "Security Aware Price Optimal Task Offloading for Connected Vehicles via Edges" In 2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall) (In Submission).

Other Publications

- Ruthik P., **SVSLN Surya Suhas Vaddhiparthy**, Pranav Kannan, and Deepak Gangadharan. "S^2P: Two-Stage Super-Pixel Algorithm for Enhanced Lane Detection on Edge" In 2024 IEEE International Conference on Edge Computing and Communications (EDGE) (In Submission).

- Ashish Sharma, **SVSLN Surya Suhas Vaddhiparthy**, Sai Usha Goparaju, Deepak Gangadharan, and Harikumar Kandath. "Attention Meets UAVs: A Comprehensive Evaluation of DDoS Detection in Low-Cost UAVs" In 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE) (In Submission).

- Pranavasri, V. J. S., Leo Francis, Ushasri Mogadali, Gaurav Pal, **SVSLN Surya Suhas Vaddhiparthy**, Anuradha Vattem, Karthik Vaidhyanathan, and Deepak Gangadharan. "Scalable and Interoperable Distributed Architecture for IoT in Smart Cities." in 2023 IEEE Virtual World Forum on Internet of Things (WF-IoT) (Accepted).

- Joseph John Cherukara, **SVSLN Surya Suhas Vaddhiparthy**, Deepak Gangadharan, and BaekGyu Kim. "Dynamic Data Delivery Framework for Connected Vehicles via Edge Nodes with Variable Routes." In 2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall), pp. 1-7. IEEE, 2023.

- **SVSLN Surya Suhas Vaddhiparthy**, Garapati Sreya, Prudhvi Raj Turlapati, Deepak Gangadharan, and Harikumar Kandath. "A Comprehensive Evaluation on the Impact of Various Spoofing Scenarios on GPS Sensors in a Low-Cost UAV." In 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), pp. 1-6. IEEE, 2023.

- S. Mante, **SVSLN Surya Suhas Vaddhiparthy**, R. Muppala, A. M. Hussain, D. Gangadharan and A. Vattem, "A Multi Layer Data Platform Architecture for Smart Cities using oneM2M and IUDX", in 2022 IEEE Virtual World Forum on Internet of Things (WF-IoT), pp. 1-6. IEEE, 2022.

- Sai Usha Nagasri Goparaju, **SVSLN Surya Suhas Vaddhiparthy**, C Pradeep, Anuradha Vattem, Deepak Gangadharan, "Design of an IoT system for machine learning calibrated TDS measurement in smart campus", in 2021 IEEE World Forum on Internet of Things (WF-IoT), pp. 877-882. IEEE, 2021.

Bibliography

[1] "undesa: World urban population." [Online]. Available: https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html

[2] "Mckinsey automotive software and electronics 2030 report." [Online]. Available: https://www.mckinsey.com/~/media/mckinsey/industries/automotive%20and%20assembly/our%20insights/mapping%20the%20automotive%20software%20and%20electronics%20landscape%20through%202030/outlook%20on%20the%20automotive%20software%20and%20electronics%20market%20through%202030/automotive-software-and-electronics-2030-full-report.pdf

[3] "Mckinsey automotive market 2030." [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/mapping-the-automotive-software-and-electronics-landscape-through-2030

[4] "Advanced driver assistance systems." [Online]. Available: https://www.ti.com/applications/automotive/adas/overview.html

[5] Y. Gong, P. Lu, and X. T. Yang, "Impact of covid-19 on traffic safety from the "lockdown" to the "new normal": A case study of utah," *Accident Analysis & Prevention*, vol. 184, p. 106995, 2023.

[6] "Lok-sabha road accidents in india." [Online]. Available: https://loksabhadocs.nic.in/Refinput/New_Reference_Notes/English/24042023_162909_102120526.pdf

[7] "Who-road accidents across world." [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[8] "Pwc - five trends five trends transforming the automotive industry." [Online]. Available: https://www.pwc.com/gx/en/industries/automotive/assets/pwc-five-trends-transforming-the-automotive-industry.pdf

[9] "Comparision between avs, cvs." [Online]. Available: https://www.roadsafetyfacts.eu/uploads/2019/03/Main_Questions-05-1.svg

[10] "Europe-road safety facts." [Online]. Available: https://www.roadsafetyfacts.eu/how-can-automated-and-connected-vehicles-improve-road-safety/

[11] “us-its.” [Online]. Available: https://www.its.dot.gov/

[12] “us-dot: connected vehicles white paper.” [Online]. Available: https://www.its.dot.gov/research_areas/pdf/WhitePaper_connected_vehicle.pdf

[13] “Times of india on connected vehicles.” [Online]. Available: @misc{cvwhite,url={https://www.its.dot.gov/research_areas/pdf/WhitePaper_connected_vehicle.pdf},title={uS-DOT:connectedVehiclesWhitePaper}}

[14] “Us-fcc dedicated short range communications.” [Online]. Available: https://www.fcc.gov/wireless/bureau-divisions/mobility-division/dedicated-short-range-communications-dsrc-service

[15] V. Mannoni, V. Berg, S. Sesia, and E. Perraud, “A comparison of the v2x communication systems: Its-g5 and c-v2x,” in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, 2019, pp. 1–5.

[16] “Us-fr dsrc bands.” [Online]. Available: https://www.federalregister.gov/documents/2021/05/03/2021-08802/use-of-the-5850-5925-ghz-band

[17] X. Jiang, H. Wu, H. Jiang, X. Du, and J. Fang, “Co-hcca: Bandwidth allocation strategy in internet of vehicles with dynamically segmented congestion control,” *Journal of Communications and Information Networks*, vol. 6, no. 2, 2021.

[18] “California trasnportation.” [Online]. Available: https://dot.ca.gov/programs/traffic-operations/cav

[19] H. A. Omar, W. Zhuang, and L. Li, “Vemac: A tdma-based mac protocol for reliable broadcast in vanets,” *IEEE Transactions on Mobile Computing*, vol. 12, no. 9, 2013.

[20] “Forbes connected vehicles.” [Online]. Available: https://www.forbes.com/sites/tmobile/2021/08/28/meeting-the-growing-demand-for-connected-car-data/

[21] “Report by global x.” [Online]. Available: https://assets.globalxetfs.com/files/ChartingDisruption2023.pdf

[22] “aecc: operational behavior of a high definition map application white paper.” [Online]. Available: https://aecc.org/wp-content/uploads/2020/07/Operational_Behavior_of_a_High_Definition_Map_Application.pdf

[23] “Visual capitalist - connected vehicles.” [Online]. Available: https://www.visualcapitalist.com/network-overload/

[24] “Statista - network speeds.” [Online]. Available: https://www.statista.com/statistics/896779/average-mobile-fixed-broadband-download-upload-speeds/

[25] X. Gu, G. Zhang, and Y. Cao, "Cooperative mobile edge computing-cloud computing in internet of vehicle: Architecture and energy-efficient workload allocation," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 8, p. e4095, 2021.

[26] S. Raza, S. Wang, M. Ahmed, M. R. Anwar, and V. Shakhov, "A survey on vehicular edge computing: Architecture, applications, technical issues, and future directions," *Wirel. Commun. Mob. Comput.*, vol. 2019, jan 2019. [Online]. Available: https://doi.org/10.1155/2019/3159762

[27] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mob. Netw. Appl.*, vol. 26, no. 3, 2021.

[28] R. Meneguette, R. De Grande, J. Ueyama, G. P. R. Filho, and E. Madeira, "Vehicular edge computing: Architecture, resource management, security, and challenges," *ACM Comput. Surv.*, vol. 55, no. 1, 2021.

[29] S. Raza, S. Wang, M. Ahmed, M. R. Anwar, *et al.*, "A survey on vehicular edge computing: architecture, applications, technical issues, and future directions," *Wireless Communications and Mobile Computing*, vol. 2019, 2019.

[30] Z. Zhou, P. Liu, J. Feng, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Computation resource allocation and task assignment optimization in vehicular fog computing: A contract-matching approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3113–3125, 2019.

[31] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management," *Ieee Network*, vol. 27, no. 5, pp. 48–55, 2013.

[32] M. Aazam and X. Fernando, "Fog assisted driver behavior monitoring for intelligent transportation system," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2017, pp. 1–5.

[33] A. Tesei, M. Luise, P. Pagano, and J. Ferreira, "Secure multi-access edge computing assisted maneuver control for autonomous vehicles," in *2021 IEEE 93rd vehicular technology conference (VTC2021-Spring)*. IEEE, 2021, pp. 1–6.

[34] L. Cruz-Piris, D. Rivera, S. Fernandez, and I. Marsa-Maestre, "Optimized sensor network and multi-agent decision support for smart traffic light management," *Sensors*, vol. 18, no. 2, p. 435, 2018.

[35] B. Gaudet, "Review of cooperative truck platooning systems," *National Research Council Canada*, vol. 10, pp. 1–79, 2014.

[36] H. Hartenstein, B. Bochow, A. Ebner, M. Lott, M. Radimirsch, and D. Vollmer, "Position-aware ad hoc wireless networks for inter-vehicle communications: the fleetnet project," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, 2001, pp. 259–262.

[37] A. Gupta, J. J. Cherukara, D. Gangadharan, B. Kim, O. Sokolsky, and I. Lee, "Global edge bandwidth cost gradient-based heuristic for fast data delivery to connected vehicles under vehicle overlaps," in *2022 IEEE 95th Vehicular Technology Conference*, 2022.

[38] Z. Li, H. Yu, G. Fan, Q. Tang, J. Zhang, and L. Chen, "Cost-efficient security-aware scheduling for dependent tasks with endpoint contention in edge computing," *Computer Communications*, vol. 211, pp. 119–133, 2023.

[39] F. Zeng, K. Zhang, L. Wu, and J. Wu, "Efficient caching in vehicular edge computing based on edge-cloud collaboration," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 2, pp. 2468–2481, 2022.

[40] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, 2020.

[41] Y. Zhao and B. Kim, "Optimizing allocation and scheduling of connected vehicle service requests in cloud/edge computing," in *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, 2020.

[42] A. Akbar and S. B. Belhaouarie, "Save: Self-aware vehicular edge computing with efficient resource allocation," in *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*. IEEE, 2023, pp. 157–162.

[43] R. Kim, H. Lim, and B. Krishnamachari, "Prefetching-based data dissemination in vehicular cloud systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 1, 2016.

[44] D. Gangadharan, O. Sokolsky, I. Lee, B. Kim, C.-W. Lin, and S. Shiraishi, "Bandwidth optimal data/service delivery for connected vehicles via edges," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, 2018.

[45] Z. Ning, P. Dong, X. Wang, J. J. P. C. Rodrigues, and F. Xia, "Deep reinforcement learning for vehicular edge computing: An intelligent offloading system," *Association for Computing Machinery*, vol. 10, no. 6, 2019.

[46] J. Feng, Z. Liu, C. Wu, and Y. Ji, "Ave: Autonomous vehicular edge computing framework with aco-based scheduling," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 12, 2017.

[47] M. Hadded, P. Muhlethaler, A. Laouiti, R. Zagrouba, and L. A. Saidane, "Tdma-based mac protocols for vehicular ad hoc networks: A survey, qualitative analysis, and open research issues," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 4, 2015.

[48] S. Johari and M. B. Krishna, "Time-slot reservation and channel switching using markovian model for multichannel tdma mac in vanets," *IEEE Access*, vol. 10, 2022.

[49] B. P. Nayak, L. Hota, A. Kumar, A. K. Turuk, and P. H. J. Chong, "Autonomous vehicles: Resource allocation, security, and data privacy," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 117–131, 2022.

[50] S. Y. Fayi and Z. Sheng, "A survey of security, privacy and trust issues in vehicular computation offloading and their solutions using blockchain." *Open Research Europe*, vol. 3, 2023.

[51] X. Chen, H. Guo, and J. Liu, "Efficient and trusted task offloading in vehicular edge computing networks," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, 2022, pp. 5201–5206.

[52] H. Chen, X. Zhu, D. Qiu, L. Liu, and Z. Du, "Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds," *IEEE Transactions on Parallel and distributed systems*, vol. 28, no. 9, pp. 2674–2688, 2017.

[53] W. Liu, S. Peng, W. Du, W. Wang, and G. S. Zeng, "Security-aware intermediate data placement strategy in scientific cloud workflows," *Knowledge and information systems*, vol. 41, pp. 423–447, 2014.

[54] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Minimizing the delay and cost of computation offloading for vehicular edge computing," *IEEE Transactions on Services Computing*, vol. 15, no. 5, pp. 2897–2909, 2021.

[55] J. Du, Y. Sun, N. Zhang, Z. Xiong, A. Sun, and Z. Ding, "Cost-effective task offloading in noma-enabled vehicular mobile edge computing," *IEEE Systems Journal*, vol. 17, no. 1, pp. 928–939, 2022.

[56] B. Huang, Y. Li, Z. Li, L. Pan, S. Wang, Y. Xu, and H. Hu, "Security and cost-aware computation offloading via deep reinforcement learning in mobile edge computing," *Wireless Communications and Mobile Computing*, vol. 2019, pp. 1–20, 2019.

[57] L. Codeca, R. Frank, and T. Engel, "Luxembourg sumo traffic (lust) scenario: 24 hours of mobility for vehicular networking research," in *2015 IEEE Vehicular Networking Conference (VNC)*, 2015.

[58] Y. Feng, C. Jayasundara, A. Nirmalathas, and E. Wong, "A feasibility study of ieee 802.11 hcca for low-latency applications," *IEEE Transactions on Communications*, vol. 67, no. 7, 2019.

[59] "Arc-it - us-dot." [Online]. Available: https://www.arc-it.net/index.html

[60] "Arc-it - us-dot." [Online]. Available: https://www.arc-it.net/html/security/deviceclasses.html

[61] "Qcom-edge device." [Online]. Available: https://developer.qualcomm.com/hardware/snapdragon-8-gen-2-hdk

[62] "Gurobi-optimiser." [Online]. Available: https://www.gurobi.com/

[63] “Python- cryptography.” [Online]. Available: https://cryptography.io/en/latest/

[64] “Nist-national institute of standards and technology.” [Online]. Available: https://www.nist.gov/

[65] L. Meng, L. Yang, W. Yang, and L. Zhang, “A survey of gnss spoofing and anti-spoofing technology,” *Remote Sensing*, vol. 14, no. 19, 2022. [Online]. Available: https://www.mdpi.com/2072-4292/14/19/4826

[66] I. G. Ferrão, S. A. da Silva, D. F. Pigatto, and K. R. L. J. C. Branco, “Gps spoofing: Detecting gps fraud in unmanned aerial vehicles,” in *2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE)*, Nov 2020, pp. 1–6.

[67] L. Junzhi, L. Wanqing, F. Qixiang, and L. Beidian, “Research progress of gnss spoofing and spoofing detection technology,” in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, Oct 2019, pp. 1360–1369.

[68] E. Basan, O. Makarevich, M. Lapina, and M. Mecella, “Analysis of the impact of a gps spoofing attack on a uav,” in *CEUR Workshop Proceedings*, vol. 3094, 2022, pp. 6–16.

[69] W. Chen, Y. Dong, and Z. Duan, “Compromising flight paths of autopiloted drones,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2019, pp. 1316–1325.

[70] Y. Gao and G. Li, “A slowly varying spoofing algorithm avoiding tightly-coupled gnss/imu with multiple anti-spoofing techniques,” *IEEE Transactions on Vehicular Technology*, vol. 71, no. 8, pp. 8864–8876, Aug 2022.

[71] “Pixhawk flight controller https://ardupilot.org/copter/docs/common-pixhawk-overview.html.”

[72] “Ublox neo 8mn https://www.u-blox.com/en/product/neo-m8-series.”

[73] S. M. Wong, H. W. Ho, and M. Z. Abdullah, “Design and fabrication of a dual rotor-embedded wing vertical take-off and landing unmanned aerial vehicle,” *Unmanned Systems*, vol. 9, no. 01, pp. 45–63, 2021.

[74] “Gps-sdr-sim: Software-defined gps signal simulator https://github.com/osqzss/gps-sdr-sim.”