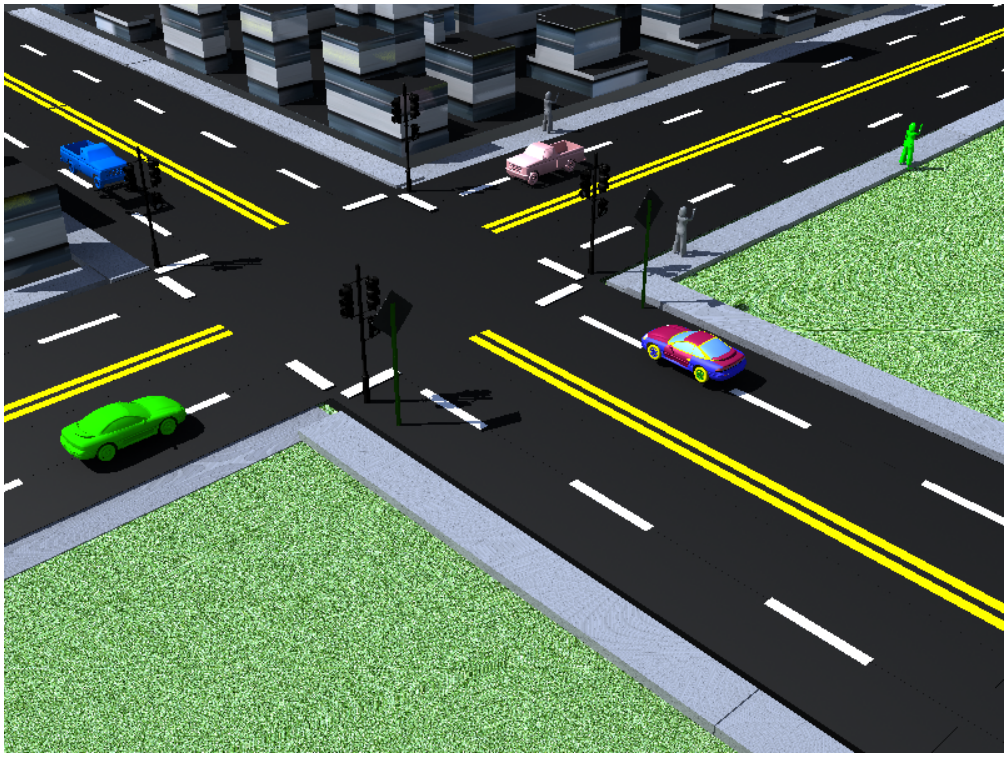

Trustworthy Intersection Management by Crowd-sourced Digital Environment Information

Master's thesis in Computer Systems and Networks

Logesh veera kumar Kalyanasundaram

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2017

Master's thesis 2017

Trustworthy Intersection Management by Crowd-sourced Digital Environment Information

Logesh veera kumar Kalyanasundaram

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2017

Trustworthy Intersection Management by Crowd-sourced Digital Environment Information
Logesh veera kumar Kalyanasundaram

Supervisor: Christian Berger, Computer Science and Engineering
Advisor: Niklas Lundin, AstaZero AB
Examiner: Riccardo Scandariato, Computer Science and Engineering

Master's Thesis 2017
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Graphical representation of Autonomous Road Intersection[6].

Typeset in LaTeX
Gothenburg, Sweden 2017

Abstract

Nowadays, the automotive industry are developing smart functionalities for vehicles, and these developments are the prior steps in the process of making autonomous vehicles. Certainly, the big future technology improvement in the automotive industry would be the autonomous cars. The time when these vehicles are on the road, they will be communicating and cooperating for faster and safer travel. To adapt to this improvement, the traffic systems should become capable of cooperating with these smart vehicles.

This project considers one of the essential elements of traffic systems, which is road intersection management. A smarter intersection management system should reduce the waiting and passing time of vehicles. Autonomous Intersection Management (AIM) is an advanced management system developed by researchers at the University of Texas for road intersections. It includes a scheduler and simulation manager to detect collisions of vehicles. However, AIM has several drawbacks like centralization(notably single point of failure) or security. In this thesis, we have worked to achieve a decentralized secure solution.

In this project, using the Design Science Research Methodology, we have designed a blockchain based Autonomous Intersection Management. Since the blockchain technology is a potential platform to implement a secure and distributed system, we have made our solution based on this technology. With the help of a popular blockchain framework called Ethereum, we implemented a proof of concept and tested with an intersection simulation tool AIM 4.0. To validate the functioning of our solution, we have conducted various experimentation based evaluation on this simulation. As a result, we have demonstrated the feasibility of blockchain based AIM thereby implementing a distributed design to address the problem. Also it is shown that the replication from the blockchain technology could improve the availability of the AIM in case of system failures.

As the autonomous cars are involving human lives, we have tried to create a safer solution. Through this, our intersection management can be robust against the vehicle's faking their positions. Assuming the autonomous cars have advanced sensors for their environmental awareness, we have tried to incorporate this sensors' information from the crowd vehicles to validate the locations of the other vehicles.

Keywords: Autonomous Intersection Management, Blockchain technology, Ethereum, Crowd-sourced traffic systems, Road intersection management.

Acknowledgements

I would like to thank my supervisor Christian Berger, Chalmers University of Technology and my industrial supervisor Niklas Lundin, AstaZero AB for providing me the opportunity to work on this project and also for the support given through out the project.

Finally, I would like to thank my examiner Riccardo Scandariato, Chalmers University of Technology for the valuable feedback and suggestions regarding the project.

Logesh veera kumar Kalyanasundaram, Gothenburg, August 2017

Contents

List of Figures

1
Introduction

1.1 Background

Recent advancements in the technology of autonomous driving have brought self-driving vehicles into reality. Along with vehicles, Cooperative Intelligent Transport System (C-ITS) is also becoming smarter. In this project, we consider the road intersection as an interesting research area under C-ITS.

Continuous research on Autonomous Intersection Management (AIM) is being done at the University of Texas [1]. This research literature has helped in understanding the essential components for intersection management and its functionality. The vital part of AIM is an infrastructure called intersection manager which manages the road crossing safely. A brief introduction to AIM is provided in this section and a more elaborate explanation is available in Section 2.1.

The researchers designed AIM with an intention to make the road intersection safer and faster to drive through. Specifically for the autonomous cars which drive by themselves without human intervention and by gathering/sensing route and environment information. The road intersection with the AIM deployed will manage the vehicle passings through a bi-directional communication between them. The endpoints of this communication are Driver Agent and Intersection Manager which are vital components of the AIM. The driver agent is a software module installed on the autonomous cars, and the Intersection Manager is the coordinator for the road intersection.

When a vehicle drives towards this road intersection, a request for passage is sent to the Intersection Manager. The Intersection Manager collects such requests from several vehicles and simulates the passage of the vehicles in the junction. Based on the simulation results, responses are sent to the respective vehicles.

1.2 Problem Domain and Motivation

This section explains the problem in the state of art intersection management system and provides motivation for this project.

An explanation on how the AIM works is required to understand the problems arising from the current state of the art system. In the current system, Intersection Manager is the service provider, and the autonomous cars are the service users.

These Intersection Managers replace the traffic lights to reduce the waiting time for vehicles at the junctions. These autonomous cars are assumed to have the capabilities of Vehicle to Infrastructure (V2I) communication, which can be used to interact with the Intersection Manager. The communication between the cars and the Intersection Manager are performed by request and response messages. Every road intersection is assumed to have a dedicated Intersection Manager that processes these vehicles' requests for pass through. The request messages contain information like speed, position of the vehicle and the response messages contain the approval or wait message.

In the design of AIM, a central coordinator called Intersection manager is solely responsible for handling the traffic. Thus, it leads to a centralised architecture [7]. There are several well-known drawbacks in centralisation, some of them being the single point of failure, security risk and bottleneck. The security risk of a central coordinator could cause lack of trust among entities involved. The key problem addressed in this thesis is the single point of failure.

The "single point of failure" drawback in the AIM is due to the dependency on the intersection manager. This central coordinator's up-time is directly equivalent to the entire system's up-time where the up-time means the time system/service is available. The reason for the failure of intersection manager could be many. Some of them are hardware failures, bugs and errors in the software. By default, the AIM does not have backup units to function as a replacement for a failed intersection manager. Also, these infrastructures could be quite expensive to include backup units.

The security attacks will be high on this intersection manager because it is a single unit that controls the system and hacking of such device would let the hacker control the traffic. For instance, if the hacker managed to hack the intersection manager, he/she could mislead the vehicle to collide with another vehicle. Also, the hacker could make a Denial of Service (DOS) attack on Intersection Manager so that the vehicles would end up waiting at the road until the intersection manager recovers. These kind of threats might cost human lives and reputation of the vehicle company.

A cyberattack on the intersection manager could let the hacker send manipulated responses to the cars, which could raise doubts in trusting the AIM. A compromised intersection manager could also lead to an accident. For example, consider a situation where two cars send requests to this Intersection Manager for passing through the same area at the same time. In that case, the compromised IM will send approval response to both cars, which could ultimately lead these cars to a collision.

The bottleneck issue might not be a critical one, but it questions the main necessity of the AIM. The AIM was developed with an intention to reduce the travel time of the vehicles crossing the road junction. When the number of vehicles on the road increases, the load on the intersection manager will be high and could lead to a slower response.

We have seen the possible problems if the intersection manager is compromised due

to a cyberattack. To add up to this, the vehicles are also highly exposed. A recent attack on an electric Tesla car [19] let the hackers apply the brake remotely and any of such zero-day hacks could create a chance for a hacked car to be on the road. These vehicles could cause accidents by sending a fraudulent request to the intersection manager and confuse it to make mistakes in allocating space and time to the other vehicles on the crossing.

1.3 Requirements for the Solution

The previous section has described the possible problems in the state of art solution AIM. In order to solve the significant problem of "single point of failure", we have considered the following set of requirement for our thesis.

1. The AIM should be functioning in case of intersection manager failures.
2. A car cannot impersonate its presence at the road junction.
3. The intersection manager should communicate only with the authorized cars.
4. The history of recent vehicle passing should be tamper-free.

1.4 Methodology

The methodology followed in our research is "Design Science Research Methodology(DSRM)". It is a popular and a standard methodology for research in designing an information technology solution. As per [16], there is a cycle of the process involved in this methodology. The following elements comprise the DSRM:

1. Awareness of the problem
2. Suggestion
3. Development
4. Evaluation
5. Conclusion

As an outcome of the first step, we get a precise knowledge of the problem and its domain. In the next step, we come up with an idea and design of a solution. Once the preliminary design is acquired, the development of an artifact would take place. It is followed by the evaluation process where the artifact would be tested and calculated its performance. The final step of a single iteration in DSRM cycle is the conclusion. The knowledge gained from the whole processes of that cycle is noted and discussed in this step. Based on the time availability, further iteration of this cycle of the design process would be carried out. We have tried to adopt this methodology in our thesis to design our solution.

1.5 Suggested Solution

In this thesis, we have identified the problems in the state of art technology (AIM) and discussed in the Section 1.2. To improvise and solve the known problems, we consider the cryptocurrency technology called Blockchain. The anonymous researcher Satoshi Nakamoto proposed this blockchain technology in the year 2008. This technology became popular after its successful cryptocurrency called Bitcoin [4]. In this project, we tried to design a blockchain based solution to benefit the C-ITS.

The blockchain technology has several benefits and features that possibly help to enhance the intersection management mentioned above in Section 1.2. Some of the potential features are trustful system, no downtime and distributed system. This technology maintains a distributed ledger among blockchain nodes, and deploys several cryptographic elements to make the transactions tamper-free. This is explained in detail in section 2.2.

1.6 Research Goal and Questions

The primary goal of this project is to make the Cooperative Intelligent Transport System (C-ITS) robust to the system failures and protective against the illegitimate requests from the autonomous vehicles. By achieving this goal, we aim to address the problems mentioned in the section 1.2 and improve trust in the system to help the adaptation of autonomous vehicles on the road. In particular, we consider the road intersection as our problem domain.

Following are the list of research questions that we resolve to steer the project in an appropriate direction and contribute to the academic community.

1. What is the level of availability improvement in the C-ITS through the redundancy from blockchain technology.
2. How can the environment sensor on autonomous vehicles be used for crowdsourced environment surveillance.

1.7 Contributions

This section describes the list of things achieved in this thesis,

1. Designed and implemented blockchain based Autonomous Intersection Management. This also provides the following benefits.
   (a) By using blockchain, the history of recent vehicles passing are stored tamper-free.
   (b) Intersection manager neglects the request from unauthorised cars.

(c) As the blockchain is used, the load sharing is sharing of computation. The IM's are made to share the load by contributing in the block formation.

2. On abrupt shutdown of the IM's and with at least one replica up and running, the AIM functions regardless.

3. The sensors from autonomous cars are used by IM to witness the presence of other cars on the road.

The evaluation strategies used for validating the above achievements are described in Section 3.6 of this report.

1.8 Limitation

In this project, we use the blockchain technology on intersection management. To make it possible, we also assume that all vehicles on the road have capabilities for V2X communication [8] and are equippped with powerful sensors like vision-based on-road vehicle detection systems, high-definition lidar based mapping systems [27] to sense the environment. Notably, these sensors will have limited distance range. In this project, we do not concentrate on improving the scheduling algorithm for intersection management instead we just have simple first come first serve by default.

1.9 Structure of the paper

In chapter 2, the background technology used in our project is described. It is aimed to help understand the working of blockchain and the simulation used in our project. Chapter 3 explains the methodology used in our project and how it aids in the design our solution. This chapter also describes the experimentation based evaluation plan that is framed for our implementation. Next on this report, the chapter 4 explains the procedure followed to implement our solution as a working prototype. Following this, chapter 5 displays the results for the evaluation conducted on our solution. In the next chapter, we discuss and analyze the results acquired. It is then followed by the related work chapter where works similar to this project is mentioned. Finally, the conclusion chapter provide us the things to be remembered from this project.

2
Background

2.1 Autonomous Intersection Management

The research on automating the road intersection has increased along with the idea of autonomous driving. One of the important related works is the Autonomous Intersection Management(AIM) designed by Dresner and Stone [1], which could be an efficient way of handling road junction for autonomous vehicles. It is necessary to understand the basic design structure of this AIM to know the state-of-art in the intersection management.

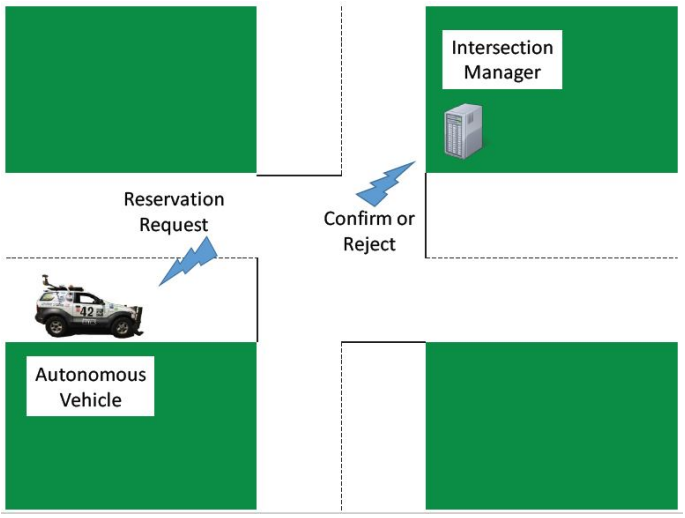

Figure 2.1: AIM: Intersection System [2]

Figure 2.1 shows the components of the AIM and their interactions. Among the components used, the main component is the intersection manager. To make this manager accessible, it should be installed on every road intersection, and it is responsible for deciding which vehicle gets priority to cross the junction. Autonomous vehicles are assumed to have communication facilities to make contact with the intersection manager. When a vehicle approaches the intersection, it contacts the intersection manager and based on the response, it crosses the intersection safely. The functioning of AIM and the communications involved is explained in the following section.

Every vehicle has a program module called the driver agent, which controls the vehicle's communication with the Intersection Manager(IM). When these vehicles

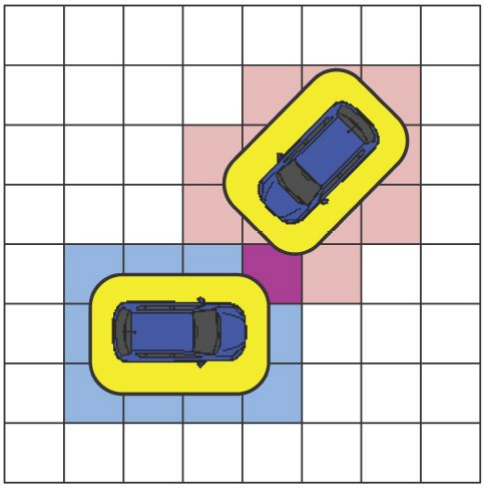

Figure 2.2: AIM: Intersection Simulation [2]

approach a road intersection, the driver agent shares the information of velocity, position, desired departure lane, arrival lane and predicted arrival time with the IM. By sharing this information with the intersection manager, the vehicle requests for the reservation of space-time in the intersection.

The intersection manager continuously listens for the requests from the vehicles and with the information from these requests, it internally simulates the vehicle's trajectory on that intersection. Figure 2.2 is a pictorial representation of the simulation where two cars collide. In the case of a collision resulting from the simulation, the intersection manager rejects the request from the vehicle. On collision-free simulation, it reserves the space-time and sends clearance to the corresponding vehicle. Then the vehicle will drive pass the intersection. Similar to an arrival at the intersection, the departure will also be communicated to the intersection manager.

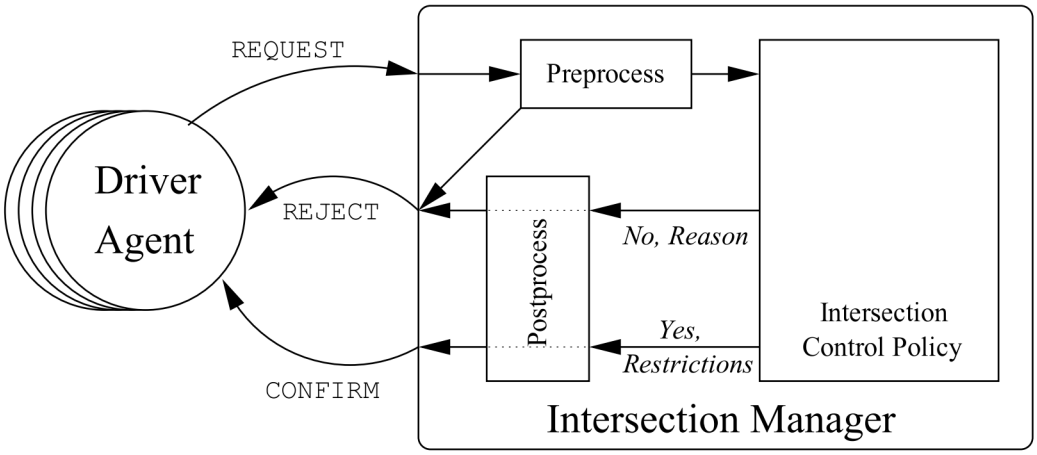

Figure 2.3: AIM: Interactions [1]

Figure 2.3 shows the interaction between the driver agent and the intersection manager. The request from the driver agent contains information about the car, speed, position, arrival and desired departure lane. These requests will be sent to the Intersection Control Policy which is part of IM and does the simulation. Then the results of this simulation will be sent to the Postprocess. This Postprocess is responsible for generating and sending an appropriate response to the car. In case of a simulation without collision between the cars, the response would contain Confirm message with supplemental information on restrictions that driver agent should follow for a safe passage. Typically, restrictions on time interval to pass. On the other hand, when the simulation shows collision, then Rejection message would contain a counter offer for the cars.

Even if any of these messages are dropped/lost in the network, the car will not cross the road junction until it receives a confirm message. In the event of a delay, the car sends a new request to the IM.

2.2 Bitcoin

Bitcoin proposed by Satoshi Nakamoto [4] is the first successful cryptocurrency. This currency system comprises several cryptography techniques and is evolving as an alternative for the typical physical currencies [24]. The trusted party are the core element of those typical digital currencies where its job is to govern the transaction and validate the double spending. Double spending in an electronic money transaction is to spend the same money multiple times. Also, the third parties charge money for their service.

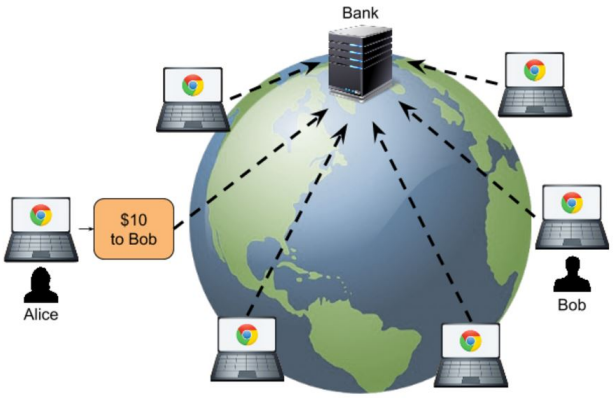

Figure 2.4: Typical digital currency transaction[9]

Figure 2.4 displays the transaction flow in a typical digital currency system. The problem in the typical digital currency system is its nature of centralization where the trusted third party is the coordinator. The centralization has several disadvantages like single point of failure and bottleneck among others. Bitcoin overcomes these by the peer-to-peer networking. It is a complete decentralized solution. In the

absence of a central coordinator or trusted party, bitcoin allows each node in the network to validate the transactions.

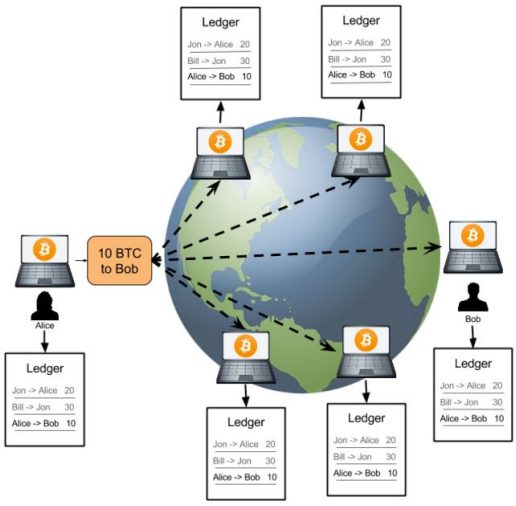

Figure 2.5: Bitcoin Transaction [9]

Bitcoin works in the peer-to-peer computer network where nodes are connected using a software called bitcoin wallet. The transaction flow in bitcoin would look like Figure 2.5. The nodes represent pseudonymous users and the system identifies them by their public key. The private keys are possessed only by their legit owners which will be used for digital signature. In the typical digital currency system, the balance amount of each user and history of transactions are stored in the bitcoin system. Unlike the typical way, these information are stored in all the network participants.

A transaction in the bitcoin system represents the transmission of money from the sender node to the receiver node. In this transaction, money is called Bitcoin. As already mentioned, there will not be any account balance maintained in the nodes. Instead, they calculate Unspent Transaction Output (UTXO), which is the balance that the sender node can spend calculated based on the previous transaction output. To avoid the impersonation, each and every transaction is digitally signed by the sender's private key. These digital signature made by the private key can be validated using the corresponding public key.

The sender node initiates a transaction by broadcasting the information of UTXO and the amount to be dispensed. When the other nodes of the network receive this broadcast, the volunteer nodes which are called miners will check the validity by comparing the UTXO and the amount to dispense. For example, UTXO of the sender is 100 bitcoins, and the amount described for the transaction is 80 bitcoins then any of these miner nodes will validate the integrity of this UTXO in the blockchain and confirm that the sender has sufficient balance for an 80 bitcoins transaction. Miners store the valid transactions locally, and on regular time interval, it combines these transactions and computes the proof-of-work to form a block.

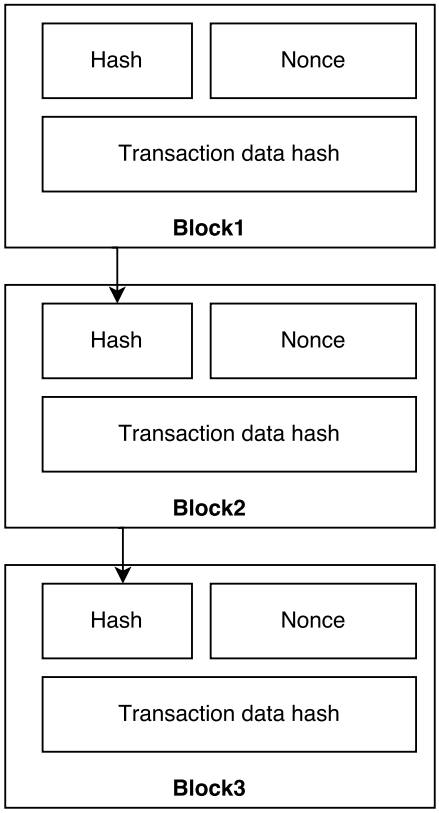

Figure 2.6: Chain of Blocks[10]

In the bitcoin system, the proof-of-work is the SHA256 hash of the block. In short, hashing is a conversion of large text into a smaller fixed length text, the resultant of hashing is called hash digest. Any one can produce a hash digest from the input large text but not vice versa. As shown in Figure 2.6, the blocks also contain a reference to the proof-of-work of the previous block forming a chain and therefore called a blockchain. We will see, how it makes the system tamper free in the following section. To keep the miners motivated, incentives are allocated to them based on their computational work. The miner node which computes the hashing first, will broadcast this to all the nodes and it will be accepted and stored in their chain of blocks. This miner will be incentivized with the bitcoins which encourages different miners to compete by contributing their computation for producing proof-of-work.

To summarize, a bitcoin transaction has the following steps:

1. A user tells the bitcoin wallet software to send a specific amount of bitcoins to a receiver.
2. The bitcoin wallet signs this transaction with the private key of the sender. This transaction contains the amount to be sent and the UTXO.
3. The transaction is then broadcasted to all the nodes of the network.
4. The volunteer nodes called miners will validate this transaction by checking the integrity of the UTXO and confirm if the sender has sufficient bitcoins to send.
5. On regular time interval, the list of transactions validated after the previous block are combined to form a succeeding block.

6. Miners compute to produce the valid proof of work (SHA 256) and broadcast the block formed. If the miner tries to cheat the system by adding a fraudulent transaction in a block, the nodes of the network just reject this block.
7. All nodes of the network add the block to their public ledger blockchain. At this point of time, the receiver can see the bitcoins received.

Apart from a digital currency system, bitcoin has also given us the distributed ledger, consensus algorithm (proof-of-work) and a secure way of transactions. We will discuss the features in details in the upcoming section of this document.

2.3 Techniques from Bitcoin

This section explains the benefits of blockchain by describing how it makes the bitcoin system secure.

i) **Distributed Ledger:** The bitcoin system stores the copy of transaction history on all nodes in the network. These nodes exist all around the world, so the whole system cannot be shutdown by any single government or regional calamities. Any node can join and leave the network at anytime. On joining, the nodes update their copy of blocks from the neighbour nodes. Also, this system presents additional challenges for hacking. When the ledger is distributed and several copies of them exist, the hacker cannot steal money by just changing one node's copy of blocks. For example, if the hacker managed to change a UTXO of a node, it will end up being different from the copies in the other nodes. Therefore, during the transaction validation, it will be identified as a malformed one and ultimately dropped. To succeed hacking it, the hacker should make changes to more than half of the nodes in the bitcoin network. It will not be practically possible when the node count in the network is high.

ii) **Consensus Algorithm:** The nodes of the bitcoin network reaches consensus on their blockchains using proof-of-work. On conflict, the nodes agree with the node which has the longest valid blockchain because it takes lot of computational power to generate a valid hash for each block. So if a change has to be made to the content of a block, then a similar hashing has to be computed. Since the blocks are chained (referenced to previous blocks), it is not enough to redo the hashing for one particular block but for all the succeeding blocks as well. This will increase the difficulty in generating the valid proof-of-work and that makes the blockchain tamper free. This brings the immutability to the distributed ledger.

The proof-of-stake is an alternative for the proof-of-work and is used in other cryptocurrencies like Peercoin [5]. In proof-of-stake, the miners prove the validity of a block by exhibiting itself as big stakeholder of the system. If the big stakeholder wants to cheat the system then it is his/her money at risk first. By depending on the fact that big stakeholder do not cheat the system, the proof-of-stake is accepted to the reality. In this, the miners compete by showing big stakes to be the miner for a paricular block and win incentives which keeps them running.

2.4 Ethereum and Smart Contract

Ethereum is an open source software implementation of blockchain technology [11]. It is used to develop decentralized software application on the blockchain. Unlike the bitcoin, it has its own coin called Ether and the miners work for this. Ether act as the fuel for running that decentralized application on nodes of the blockchain. The decentralized applications are developed with the smart contract of Ethereum. These are the software program that is written using the Ethereum specific language like Solidity, Serpent etc. Solidity is a high level programming language that runs on Ethereum Virtual Machine [12]. The smart contracts are deployed on Ethereum Network and act as a service. Based on the event of transactions, the smart contracts will have its program logics.

In Ethereum, like the bitcoin, the users are identified by an account. This account is protected with public key cryptography. Each account has their own smart contract codes (if deployed) and Ether balance. The transaction to the smart contracts would cost Ether for the sender based on the computational energy. Such smart contracts can be used for several applications, for example, it can be used to buy assets online by transacting Ether. Here the user and the smart contract are the seller/buyer of an asset. The program logic for this smart contract would be to check the identity of the caller and validate the ether transaction. If the logical conditions are met, then the asset's owner is updated accordingly. It has the same benefits and protection of bitcoin. There are several software alternatives available for Ethereum but it is the one of the popular among the developer community. There are more than 700 software applications developed using Ethereum framework [28].

2.5 Analogy

Following are the list of analogies that could be helpful to correlate the blockchain terms with the problem domain.

- **Blockchain nodes vs Intersection Managers:** In blockchain, the nodes are connected to the peer-to-peer network. Similarly, the intersection managers could also connect to form the peer-to-peer network.
- **Smart contract vs Intersection Control Policy:** In Ethereum, the smart contracts are the software program that is deployed on the network. Likewise, the implementation of the Intersection Control Policy could be deployed on the blockchain network.
- **Ethereum accounts vs Vehicle authentication:** For a transaction in Ethereum, a registered account (public key encryption) is required. Correspondingly, vehicles would need to possess a registered account to send a request/response.
- **Transaction vs Vehicles request/response:** In cryptocurrency, a transaction represents a money transfer from a sender to receiver. In our problem, each transaction expresses a request or response between the IM and vehicles.
- **Miners vs Intersection Managers:** Typically in blockchain, miners could be any volunteer node which involves in the computation of proof-of-work.

Likewise, any IM part of blockchain can take part in the computation of proof-of-work.

3
Design of solution

3.1 Problem Identification

In Section 1.2, the Autonomous Intersection Management (AIM) was analysed and the possible major problems were listed. The following list summarizes the identified issues currently present in AIM:

- Centralized Architecture - single point of failure
- Security of the IM
- Safety of the vehicles on road

In this project, we have addressed the issue of single point of failure and securing the IM against illegitimate requests from the vehicles. The AIM has a central coordinator for decision making which could lead to single point of failure. The illegitimate requests from the vehicles could make the IM serve a non-existing vehicle's request. For example, if an intruder sends an illegitimate request to the IM for a passage and this request for a non-existing vehicle would consume the space-time slot allocated by the IM. Similarly, a vehicle can lie about its position in the traffic queue and ends up getting priority.

3.2 Blockchain of Five Nodes

Contemplating the limitation of the vehicle's computational power and storage capacity, we consider a design that addresses the blockchain's compatibility for intersection management and that is outlined in the following.

State of the art AIM has the IM unit deployed at a physical location near junction which would require space, power supply and internet access. In our solution, we propose to make the intersection management as a remote service that runs on a blockchain network. Compared to the state of art, it is easier to establish and maintain this infrastructure. Intersection design in urban traffic may include three way (T-junction), four way or multi leg designs. Since multi leg intersections are commonly avoided in most cases, we consider a four way intersection model as opposed to a three way intersection in this project since it provides us the opportunity to explore the design with more blockchain nodes. For the same reason, we consider the four way intersection to be surrounded by four other road intersection as shown in Figure 3.1. Since the proposed solution is independent of the number of legs in the

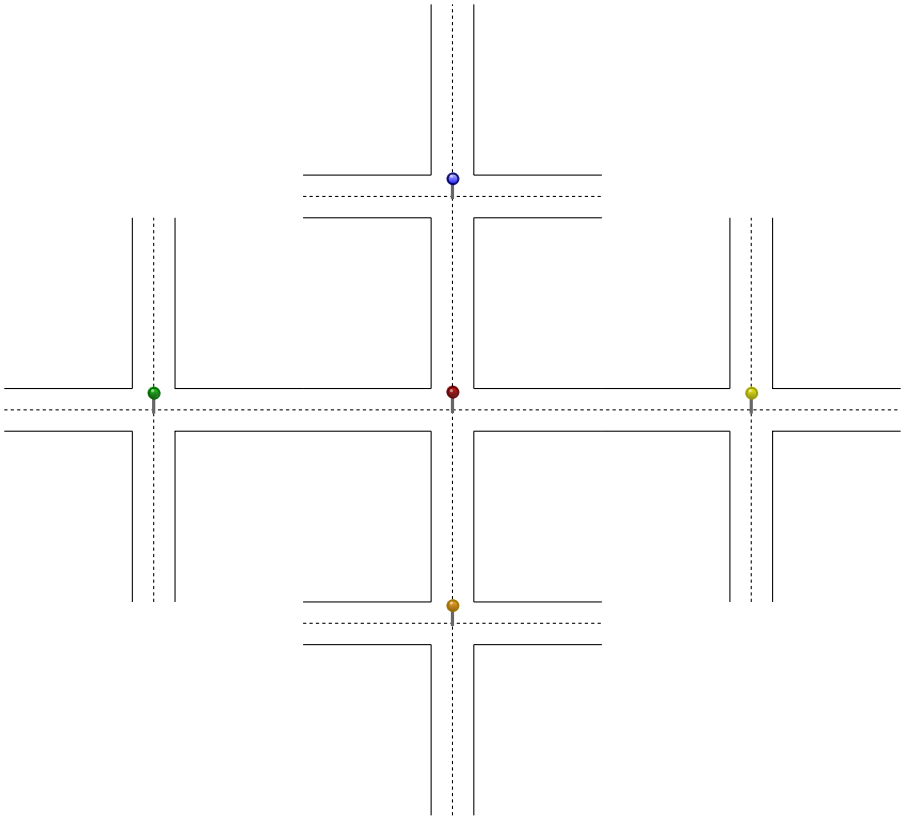

Figure 3.1: Blockchain nodes on Intersections

intersection, the same method could be used for other types of intersection models as well. For example, in the four way model shown in Figure 3.1 the IMs combine to form a five node blockchain network whereas a three way intersection model would have a four node blockchain network. The centre red pin is the intersection that is managed by the five node blockchain, and the neighbour intersection pins represent the other nodes. These blockchain nodes need not be present physically at the intersection location but instead at a remote location like a computer server room and the traffic department could govern this infrastructure.

In this project, the blockchain framework called Ethereum is used to form this blockchain network of five nodes. As explained in Section 2.4, the smart contract is a software that is deployed in the blockchain network and runs on specified events (transaction). Thus the intersection manager can be deployed as the smart contract and run on every blockchain transactions which are vehicle's request and response in this problem.

In blockchain, only a valid user can communicate with the nodes. In this case, the users are the vehicles, and the nodes are the Intersection Managers. So the vehicles (users) should possess a valid public and private key to send digitally signed messages to the IM. Also, these public keys should be stored at the Intersection Managers, along with the identity of the corresponding vehicle. This registration of the public key could be governed by the traffic department, trusted by the vehicles. Using this public key encryption for digitally signed messages, the IM can believe the messages

are produced only by a valid vehicle rather than impersonation. This authentication will become void if these keys are stolen from the vehicle and used by a hacker to produce a valid message. This aspect is currently not considered as within the scope of this project.

When a vehicle that drives towards a road intersection, from a specified distance, it will try to establish a connection and send the passage request to one of the IMs (nodes) in the blockchain. If it fails to make a connection with an IM, it will send the request to other IMs of the same blockchain network. In case of an IM failure in between this request/response handling, the vehicle will switch the IM and send the request again. To make the vehicles aware of the IM's status of failure (system down), the blockchain framework will timeout this transaction (passage request) and notify the corresponding vehicle.

When an IM receives a request, it performs a simulation for collision detection similar to the state of the art AIM [1] before a response is sent to the vehicles. In a blockchain point of view, these requests/responses are stored to form blocks. As described in the background Section 2.2, forming a block would require great computational power. To aid that, in this setup the other nodes of the blockchain combine to share the computation and contribute to form a block. Since these blocks could result in the requirement of a large amount of storage, there should be some cleanup in place to store only the transactions in recent time.

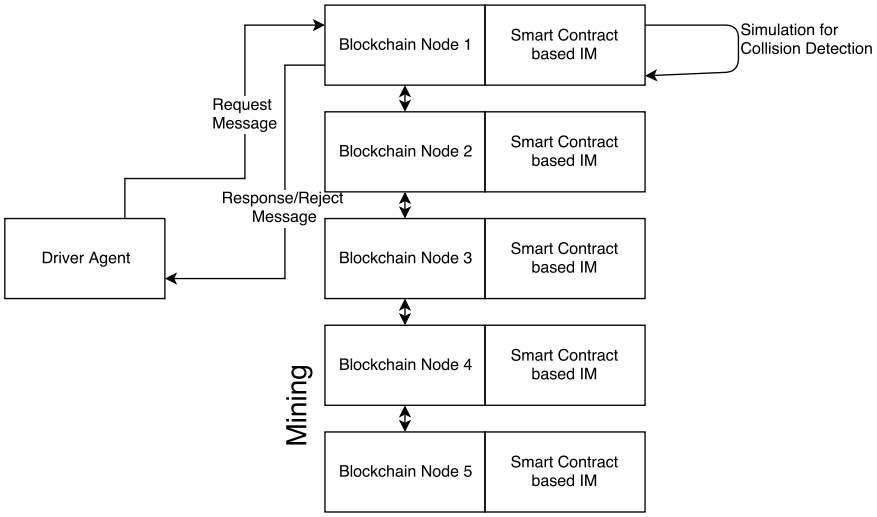

Figure 3.2: Communication between Driver Agent and Blockchain based IM

Figure 3.2 shows the message flow in our solution. Driver Agent initializes the process by sending request message to one of the available Blockchain node, and the smart contract(IM) deployed on this blockchain network performs the simulation to check if there is a collision. In case of no collision, the blockchain nodes combine to form a block through mining. It stores the passage of this particular vehicle and forwards the response to the driver agent. In the event of a collision resulting from simulation, it sends a reject message to the driver agent and a request message is created again for a different time.

3.3 Crowd Sourced Environment Information

In the second part of this thesis, we have tried to increase the trust between IM and the autonomous cars. For an autonomous car to function, it requires a lot of powerful sensors to recognise the environment. The idea is to use this sensor information from several autonomous cars to have better awareness about the IM's perception of the environment.

In the scope of this thesis, it is assumed that the autonomous cars have a 360° coverage of the surrounding and can identify the presence of a vehicle within its range. So when these cars send passage requests to the IM, it appends this sensed information to the request message. IM fuses all these similar information from vehicles and has a database of vehicle's position at a particular time.

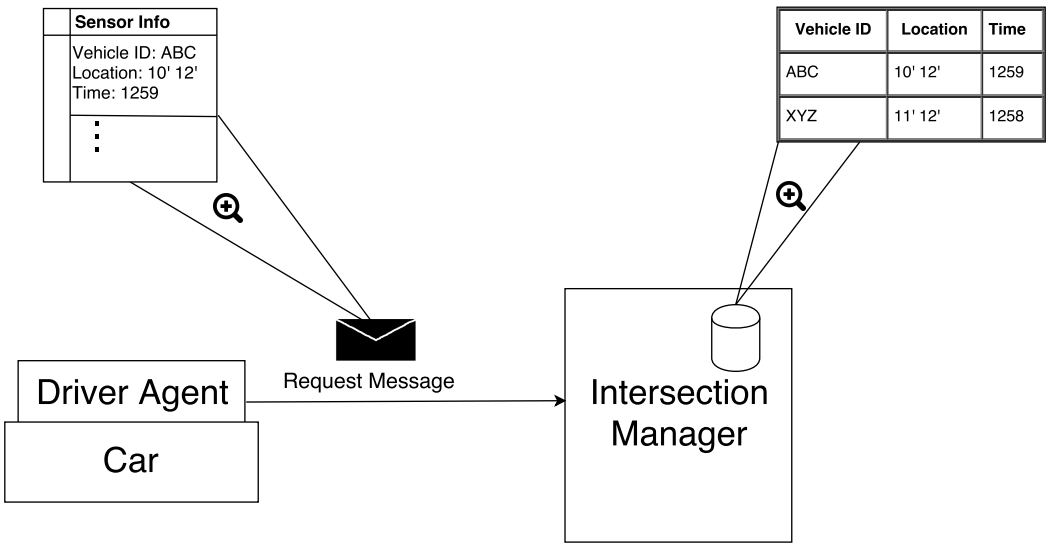

Figure 3.3: Sharing Sensor Information

The sensed information from the vehicles will be shared with the IM through messaging and it contains the vehicle ID, location and time. Figure 3.3 shows the sample message flow and its content. The IM combines this information from several vehicles and on regular interval will clean old information based on the time of witnessing. On the event of a request message, it checks the location of that vehicle in its database and validates if that vehicle exists physically on the road. If it is valid, then the IM proceeds to process the request, otherwise it ignores the message.

This process can be better understood with the help of a scenario as shown in Figure 3.4. In the image, the circle around the car A shows the range of the sensor and the circle at the junction is the range of the sensor monitored by the IM. When car B approaches the junction, a request for passage is sent to the IM. As this car is not seen by any other car, this request will be ignored. When the cars approach towards the junction as shown in the Figure 3.5, car B falls inside the sensing range of car A and that of the sensors at the junction. This makes the IM aware of the presence,

and it processes the request from that car B.

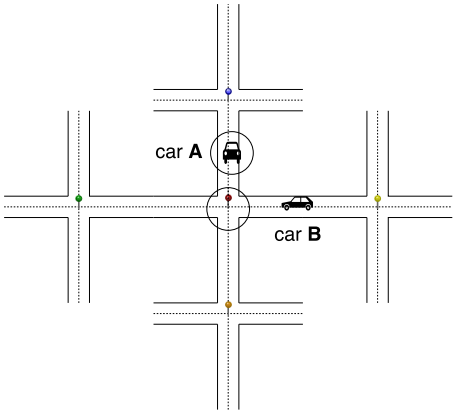

Figure 3.4: Sample Scenario I

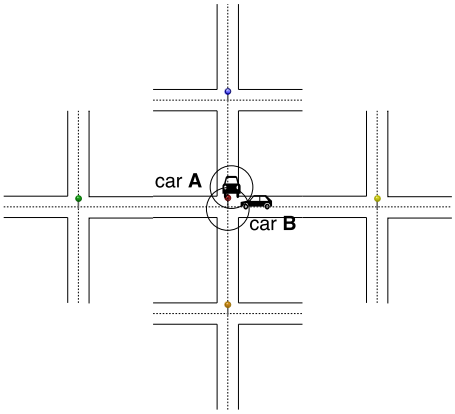

Figure 3.5: Sample Scenario II

3.4 Motivation and benefits over alternatives

This section contains the reason and benefits for our choices in the solution design.

One of the critical choices made in this thesis is the choice of the blockchain for the IM replication. There are several choices available for replication, for example having backup IMs would have solved the single point of failure, but it requires reliable syncing algorithm implemented and improved security of the data stored in the back-ups, in addition to the extra hardware. The blockchain solution provides us with the replication along with high syncing algorithm implemented. It also contributes to the reliability on the IM. There are numerous frameworks available to implement blockchain in software applications [26].

From the cryptography techniques followed in the blockchain, each block is equivalent to the history of requests/approvals, and it is a tamper-free ledger as explained in the background Section 2.5 of this report. These tamper-free records prevent any intruders from erasing their passage.

Another design choice made in this thesis is the use of five node blockchain network. In most cases, a road junction would have four surrounding junctions. So in this thesis, we choose this common case. By having several nodes, computation load required for the proof of work can be split equally. Also, a blockchain network of five nodes can give improvement in the uptime (availability) of the system. This system will be up and be running until all the nodes fail. On the recovery of a failed node, the other nodes will supply the missing blocks to the new node joining the network. During the downtime of a node, the other nodes in the system will continue to work instead of stopping the service.

This blockchain service will be allowed to use by the vehicles after authentication using the public key cryptography. This feature can avoid impersonation, i.e. the ve-

hicle cannot pretend to be another vehicle and cause accident without being noticed.

In a typical road transportation, vehicles are required to pay the traffic department for using a road, and it is called Toll charges. Likewise, on vehicle's communication with the IMs(nodes), it has to send gas (crypto money Ether) for its corresponding computation at the nodes. By accounting the gas/Ether spent by each vehicle, we can be very specific in the toll payment for the vehicles.

One last important choice made in our thesis is to share sensor information from the autonomous cars to identify the physical presence of the other cars on the road. i.e. to make the IM more robust against vehicle's lies on positions/locations. One of the alternatives to this solution could be to deploy several sensors on the sides of the road near the junction to have the better vision for the IM. However, the idea of using the sensors from autonomous cars could reduce the cost. Also, a car's trust on IM could be improved as this information is available from the cars as well. The cross verification of information (from requests) performed by the IM makes the output from the system always legit. So the vehicles (users) can rely on this service where the human lives are involved.

3.5 Security Analysis of the Solution

The thesis suggests using blockchain to store the recent passage of the vehicles. As only five nodes of the blockchain store these data in blocks, there is no other way to access and read these data. For the hacker, a chance to acquire access is by taking over a blockchain node and this can be mitigated by an authentication process in place. Also, these nodes are statically added to the network, unlike the public blockchain networks where anyone can join and listen to the transactions.

When it comes to the trust model, the IMs (blockchain nodes) are operated by the traffic department and the cars can rely on them as they do in a typical traffic light system. In case of a compromised traffic department, a typical scenario would be get the same space and time alloted to more than one vehicle. In such cases the autonomous vehicles should still be able to sense the collision by themselves and avoid a collision. For autonomous vehicles, this should be a minimum possibility as there are such emergency braking technology already developed and becoming standard in the automotive market [25] [29].

On vehicle sending requests to the IM, it will be identified as a legit vehicle using authentication. In case the credentials of a vehicle are stolen by the hacker, he/she could impersonate and misuse the identity. It should be the car manufacturer's responsibility to handle credentials in a secure way.

3.6 Evaluation

In this section, we explain the experimentation conducted on the proof of concept and their results are described in the Chapter 5.

3.6.1 Proof of Concept

To prove the feasibility of the solution design, a proof of concept has been made and uploaded to Github. It is uploaded as a publicly available open source project. The implementation is divided into two repositories, one for the modified AIM simulator and the other other for the blockchain setup. The projects can be accessed through the following links:

- Modified AIM simulator at "https://github.com/logeshveerakumar/Thesis-AIM-Blockchain"
- Ethereum Blockchain setup at "https://github.com/logeshveerakumar/Thesis-Ethereum-Setup"

3.6.2 Performance Evaluation

This experiment gives us the performance of our proof of concept and also helps to examine if it can be suitable for the real-time use. Specifically, we calculate the time taken for each vehicle to get served by the Blockchain based IM and cross the road junction. To examine the dependency of performance on the computing power, we do this experiment on two computers with different hardware specifications.

1. Computer I:
   (a) Memory: 4 GB DDR3
   (b) Processor: Intel Core i5-4200U
   (c) Cores: Dual-core processor
2. Computer II:
   (a) Memory: 16GB DDR4
   (b) Processor: Intel Core i7-6920HQ
   (c) Cores: Quad-core processor

A Java function is coded to calculate the time taken from the event of a car sending a request and getting the response back, to cross the junction. With data on several such times, it calculates the average time that would take for car to get served by the IM.

This experiment running on the machines with different capabilities would show us the variation in the performance and suitability of this solution in the real-time systems. Steps to run this experiment are described in the Section 4.6.

3.6.3 Service Availability

The proposed solution has five blockchain nodes on use with an intention to provide service with high availability. We can examine this by stopping some of the nodes and leaving at least one node running.

This experiment is carried out with a small shell script that is coded to stop the blockchain nodes one by one at random time intervals. Manual validation is then carried out to determine if the running nodes can still serve the vehicles. Also, performance is examined (explained in subsection 3.6.2) when each node is taken down individually and the remaining are in service. This helps to understand the availability of the Intersection management on unexpected node failures.

3.6.4 Load Sharing

In the context of Blockchain, the huge load issue is due to the computation of the proof of work that is done for every block formation as described in the background Chapter 2 of this document. Now we have given a solution for the centralised Intersection management by using a replica of the blockchain system. To examine the load sharing between these replicated blockchain nodes, we have to calculate the contribution of each node in block formations. In order to measure the contribution, we will calculate the Ether (virtual gas) received by each blockchain node for their computational contribution.

This evaluation would be conducted with an implemented Java function that fetches the amount of Ether received by each blockchain node for certain numbers of vehicle crossings. For example, we would fetch for every 20 vehicles crossings.

3.6.5 Authorization

In our blockchain based solution, every communication is digitally signed. To do that, public and private key pairs are used. When a vehicle sends a request message to a node, it digitally signs it with its private key, and at the blockchain node, it decrypts the message with the public key of the sender vehicle. Also, these vehicles have to be registered with the IM by sharing their corresponding public keys. As mentioned in the design of the solution, this could be carried out by the traffic department. When a blockchain node receives a message from a non-registered vehicle, the message is ignored since the registration check is failed. This registration check can be examined by sending a message with a vehicle ID that is not present in the blockchain node's registry (list of registered vehicles).

The Ethereum framework stores the private key encrypted with the password provided by the user. To examine this, we can provide a wrong password intentionally and check the blockchain behaviour.

3.6.6 Crowd Sourced Environment Information

We have designed the solution to share the environment information from the autonomous cars and also information from the sensors at the centre of the junction to the intersection manager. With that collected data, the system has the chance to examine if a request from a vehicle is legit i.e. if the vehicle is physically present at the location. If it is present and is witnessed by any other car, then IM would have that in its database and processes the request message from that car. If it is physically present but not witnessed by any other cars then IM would just ignore the request message. In that case, the car approaching the junction in its course will resend the request and when the sensor at the centre of the junction can witness the car, the IM will process the request message.

To verify this part, we implement a prototype which mimics the above mentioned behaviour. In the event of a request, the vehicle's position and time will be cross-checked with the information about the environment obtaied from peer vehicles. Essentially, we have to find a statistical mean of the number of requests that can be verified with this available environment information from the peer vehicles.

4

Implementation

In Chapter 3 the design of our solution is explained. To examine this idea, we have simulated the scenario shown in Figure 3.1 using Autonomous Intersection Management (AIM4) simulator that is available on the website of Texas University [13]. This simulator was used for several researches on intersection management. In the below section, more details about the simulator is explained.

4.1 AIM4 Simulator

AIM4 Simulator is a Java-based road intersection simulator where the AIM algorithm mentioned in the Section 2.1 is implemented. The source code of this simulator is available on the same website. This can be easily compiled to get an executable jar file using the project management tool called Apache Maven. This simulator can simulate several intersections and cars. Importantly, the communication between the driver agent (inside car module) and the intersection manager is simulated. The simulator can be configured to set the speed of the vehicles and the number of lanes on each road. Figure 4.1 and 4.2 shows configuration used and the simulation screen.

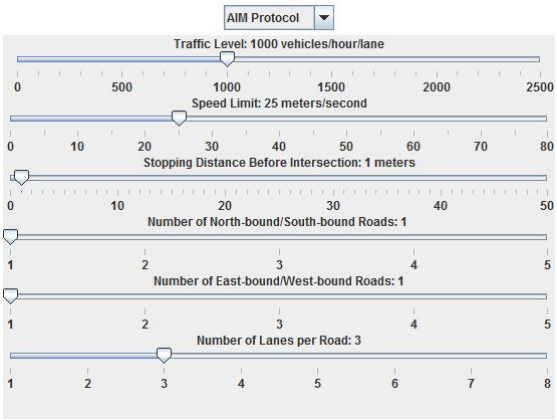

Figure 4.1: AIM4 Configuration

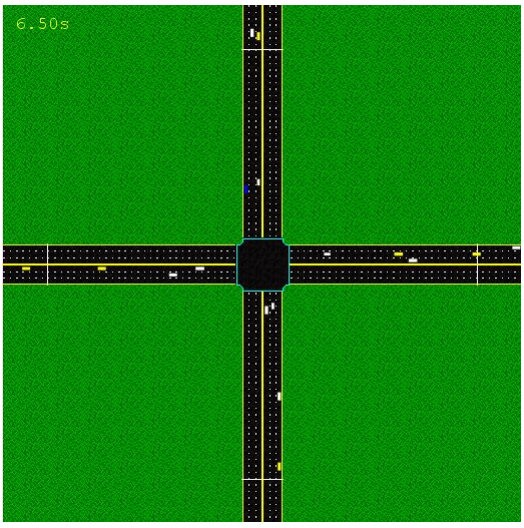

Figure 4.2: AIM4 Running UI

4.1.1 Working of AIM Simulator

In the previous section , the features of the AIM simulator is mentioned and in this section more technical details are presented. The V2I manager is the java class that maintains the V2I message queue. This queue stores all the messages from the individual driver agents and it will be accessed by the Intersection Manager class to process them. Figure 4.3 shows the flow of the request/done messages between the driver agents and IM objects. This queue has the requests and done messages from the driver agents. These messages from the queue will be handled by the IM sequentially.

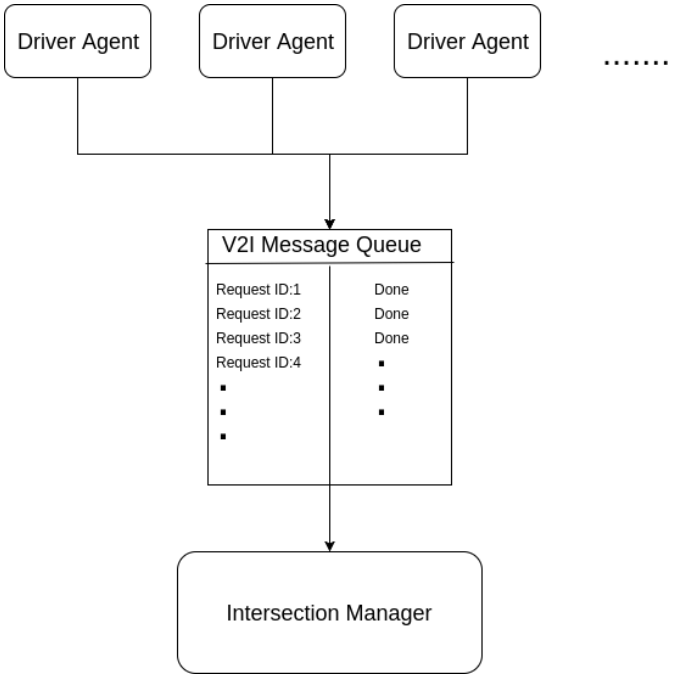

Figure 4.3: AIM4 V2I communication

The Intersection Manager processes all the messages from the queue by simulating the passage of vehicles to check if there is a collision with other vehicles. Based on the result a corresponding message is added to the I2V message queue. Then the I2V manager class fetches and sends those message to the matching vehicle's driver agent. The Figure 4.4 shows the I2V message flow.

The code implementation was forked and developed to store and pass the V2I/I2V messages through the Blockchain to prove the feasibility of our design. Five blockchain nodes created and a smart contract was developed to support this setup. More description about setting up the blockchain network with nodes and smart contract deployment is available in Section 4.2.

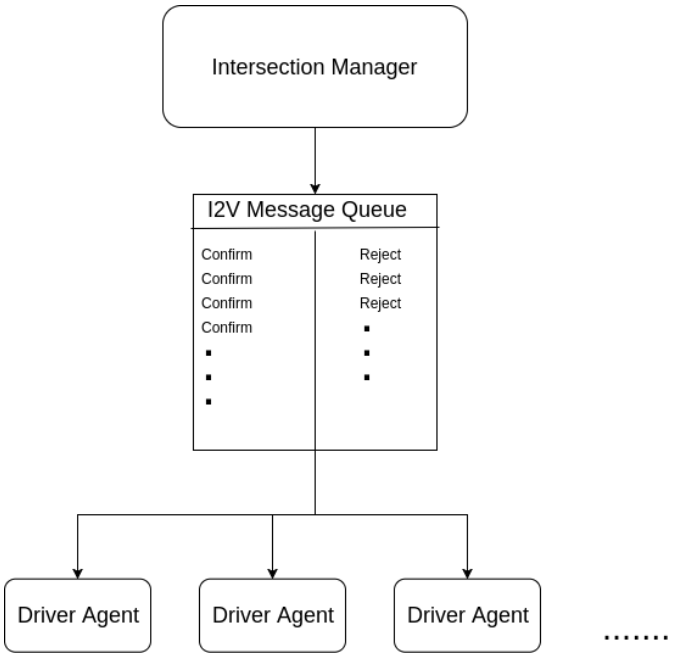

Figure 4.4: AIM4 I2V communication

4.2 Simple Blockchain Network and Smart Contract Deployment

The Ethereum was found to be one of the reputed and powerful Blockchain tools to develop a distributed application. To start a private Ethereum network, we used a client software called "EthereumJS-TestRP" [14]. It is a Node.JS platform based Ethereum client, and it is popularly used for easing the development of a single node network. By default, it creates ten accounts with some Ether balance. The first account will be the default account of the node connected to the blockchain.

There are other alternatives for this like Geth (Ethereum implemented in GO language) and Parity. But TestRPC avoids the mining process by giving away free Ethers at the beginning. Although it is not an ideal setup to experiment with Blockchain, it was handy in the earlier stage where smart contract and deployment is new. Once it is started, we got a private blockchain running with a single node in it.

As a second step, we needed a smart contract to deploy to the network. As an initial experimentation, we have written a contract to store the approval messages from intersection manager using the Solidity language. Even though, it is written in Solidity, the compilation and deploying of this smart contract was done using the Node.JS script with Web3 and SOLC packages. These packages are beneficial for other operations on deployed smart contracts. Figure 4.5 displays the blockchain storing the approval messages from the IM using the smart contract.

So far we have set up a blockchain network of a single node and a smart contract to store the I2V messages. To make the simulation much realistic, we need five

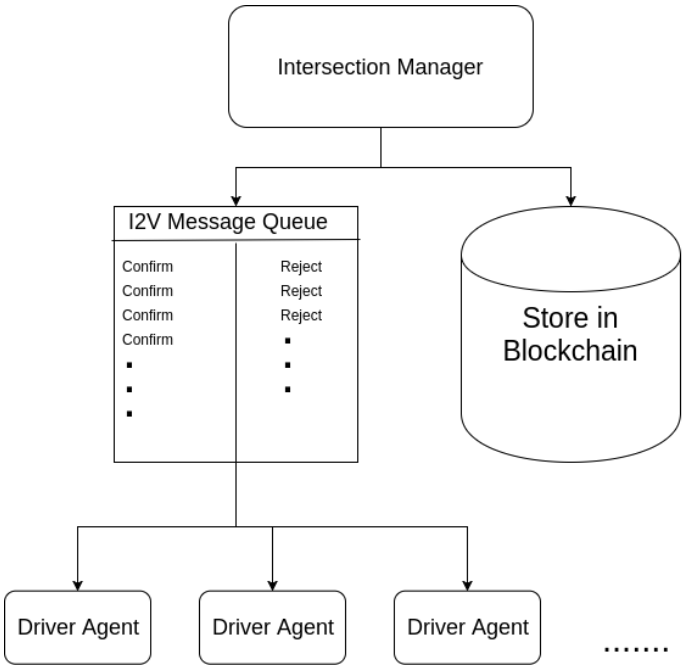

Figure 4.5: Smart Contract Storing Approvals

nodes running in the blockchain network. For establishing such network, there are only two options namely Geth and Parity. Among these, Geth was the well-evolving software and more accessible to achieve our requirement.

4.3 Five Node Blockchain Network

The following steps are executed to establish our simulation setup on a developer laptop.

1. Start five individual Geth node instances on same Localhost and Network ID, but on different ports. The genesis block that is the preset origin of the chain is used for the instance creation.
2. Create Public Key Encryption key-pairs in Geth node instances, and as a result, we get accounts created with these keys. Each key can be assigned and shared to the vehicles by the traffic department.
3. Add Ether (Crypt money) to these accounts by editing the Genesis file and re-initiating the instances for our simulation. These money will be spent by the vehicles on their passages to form blocks. This project did not use the crypt money in the solution design so it creates no impact for solution.
4. To make these nodes part of same blockchain network, create a static-nodes file (contains address of peer nodes) at each instance.
5. Start miners on each node so the transactions can be formed into blocks. We made a script that starts mining on pending transactions. This could save time and computation. It also avoids creating empty blocks.
6. Enable RPC (Web3) support for each node so it can be contacted remotely for communicating with the IM .

These steps successfully start a five node blockchain network which was very helpful for our evaluation. Appendix A shows the content of some essential script files that are coded and used for executing the steps mentioned above.

4.4 Implementation of Blockchain based Intersection Manger

In this section, we will see how blockchain is made to serve as an Intersection manager. Ethereum is the blockchain framework we have chosen for our experiment and its capabilities are explained in Section 2.4. The smart contract is the software program that is installed or deployed on the blockchain network to run on certain events or transactions. Our solution design is to implement the Intersection Manager as a smart contract. Figure 4.6 shows the communication flow in the system where the request from the driver agent will be sent to the smart contract on the Ethereum network. Based on the request, the smart contract will run the collision test and send q response back to the driver agent. In AIM4 simulator, this collision test is done by the Intersection manager.

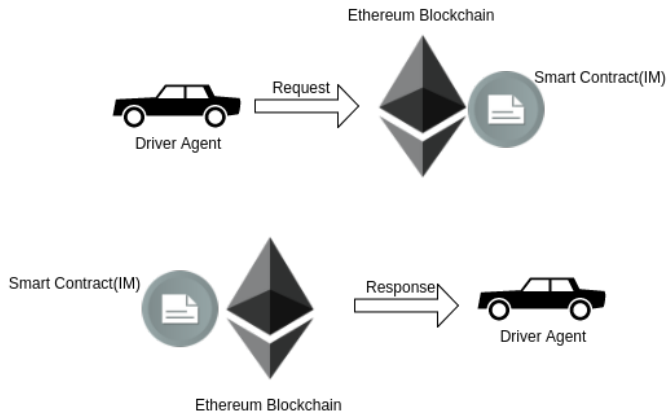

Figure 4.6: Driver agent and Blockchain communication

There is a time constraint in implementing entire Intersection manager as a smart contract because the AIM4 simulator was developed in Java and the smart contract should either be in Solidity or Serpent (smart contract specific languages). So as a workaround, we made an alternative to this. Figure 4.7 shows our current setup where the driver agent sends the request to the smart contract in the blockchain. Then it stores the requests and forwards that to the IM instance that is running in the AIM4 simulator. The response comes back in the same way. For the purpose of verifying our design concept, we consider the setup shown in Figure 4.6 and in Figure 4.7 are equivalent in working.

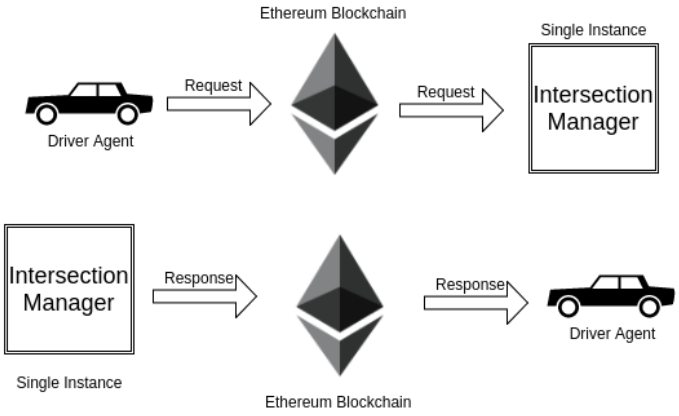

Figure 4.7: Driver Agent and Blockchain communication in Proof-of-Concept

4.5 Crowd-sourced Environment Information

At the IM, we store the sensor information like position, location and time of the vehicles present in the request messages. When a new request arrives, we look through all the position and time information from other cars to check if this sender car existed in the same place and time as any other car. This is considered to be equivalent in collecting neighbor vehicles information. If a peer car is found, we flag it as *verified* and for the opposite as *not verified*.

When a request is send from the vehicle, a function is implemented to check whether the vehicle's position lies within the range of the sensor present at the center of the road junction. We flag it as *verified* and *not verified*. If a vehicle is flagged as *not verified* then the request will be dropped by the IM. In that case, the vehicle approaches to the center of the junction and ultimately falls within the range of sensor at the center of the junction. Now this vehicle will be flagged as *verified* on new requests. One way or another, it can be witnessed by the IM before letting it cross the junction.

4.6 Steps to run the Simulation

This section presents the steps required to run the simulation setup used in this project.

1. Download the modified AIM4 simulator code and blockchain setup code from the public repositories given in section 3.6.1.
2. Run the blockchain network of five nodes using the directions given in Appendix A.
3. Open the modified AIM4 simulator using the NetBeans IDE. The following steps can be helpful for the same :
   - Click File –> new project
   - Click Maven –> POM Project
   - Select the POM file from the project directory
4. Run the Simulator.

- Click Run –> Build Project
- On successful build, Click Run –> Run Project.
- Click the start button.

5. In the output console, list of vehicles passed and the results of the performance test will be displayed.

5
Results

In this chapter, we have listed the results acquired from the evaluation that is conducted on our proof of concept. The explanation of the individual evaluation is given in the Section 3.6.

5.1 Performance Evaluation

We have conducted the performance test as described in Section 3.6.2 on two machines with different hardware configurations. The average time taken after several vehicle passages is given in Table 5.1.

After No. of vehicle passages	**Average Time in Computer I (seconds)**	**Average Time in Computer II (seconds)**
10	8.4	3.4
20	8.5	3.9
30	8.0	4.0
40	8.2	3.4
50	7.9	4.2

Table 5.1: Average Time of the Service

The average times from the Table 5.1 includes the time taken for a request reaching the blockchain node, processing the request by the IM and sending back the response to the vehicle. Notably, it includes very less network latency because the simulation is done on a single machine, so the request and response messages are just loopbacks. From the numbers, it is obvious that the performance of the blockchain based IM increases when the computing capability increases. Also, all five blockchain nodes were made to run on a single machine, but in a real-time situation, we would have better-dedicated machines that can make our proof of concept feasible for the real-life situations.

5.2 Service Availability

The service availability is evaluated by shutting down a random blockchain node one at a time using the shell process kill commands. The situations where at least

one node is up and running, the service was uninterrupted. When a vehicle requests for a service, it checks the status of the node and switches nodes if a node is found to be down. If a node fails during the processing of a request, then the vehicle would request for the status after waiting for some time and switches to different blockchain node for the service when no response is received. The failed nodes can be joined and synced back to the blockchain network easily. Restarting these failed nodes commences the syncing of recent vehicle passages. In Table 5.2, we also gathered information on the performance with a different number of nodes. It shows that the performance is high when more nodes are up and serving.

Number of Nodes in service	**Average Time for the Response in Computer II (Seconds)**
5	~ 4
4	~ 4.8
3	~ 5.3
2	~ 6.2
1	~ 6.9

Table 5.2: Performance on Node Failures

5.3 Load Sharing

In this part, the results from the evaluation of the load sharing performance as described in Section 3.6.4 is presented. Remember that the amount of Ether received is seen as a way to measure the computation. Each vehicle is initially loaded with a specific amount of Ether as explained in Section 4.3. Table 5.3 shows the contribution of each blockchain node by the data on Ethers received by them. The value given under node 2,3,4 and 5 are the percentage increase/decrease compared to node1. As mentioned, this data is fetched on the mark of every 20 vehicles crossing.

After	**Node 1 (Eth)**	**Node 2**	**Node 3**	**Node 4**	**Node 5**
20 Vehicles	503,510	-3.45%	+4.79%	+2.62%	+1.14%
40 Vehicles	1,042,813	+4.79%	+13.05%	-11.26%	-2.79%
60 Vehicles	1,496,178	+6.14%	+10.84%	-2.53%	+6.51%
80 Vehicles	2,019,251	+7.72%	+5.72%	-3.95%	-0.04%
100 Vehicles	2,425,375	+8.09%	+4.50%	-0.31%	+3.09%

Table 5.3: Ether Received by Each Nodes

The information from the table shows that the Ether received for the computational

contribution by each node is approximately same and proves us the load sharing in our replication based solution.

5.4 Authorization

We have seen the evaluation plan for authorization in Section 3.6.5. To highlight that here, the first case is to send a request message as though it is being sent from a non registered vehicle. In this case, we have replicated a request message from a registered vehicle and the replicated message is sent to the IM as though it is initiated by a non-registered vehicle. A non registered vehicle is one whose public key is not present in the IM database. As expected, this message is not processed by the IM by identifying that the sender vehicle is not registered.

In the second case, we have planned to try the wrong password for decrypting the private key of a vehicle and use it for digital signature. As a result, the Ethereum framework displays Authentication error and restricts the intruder to initiate the communication with the IM using the wrong password.

5.5 Crowd Sourced Environment Information

In our solution, we have proposed that the environment information from the autonomous vehicles can help identify if a request from the vehicle is legit or not. In order to examine this function, we collect the number of vehicles being witnessed by other vehicles and also the number of vehicles witnessed by the sensor at the centre of the junction. Figure 5.1 shows the number of vehicles being witnessed by the other vehicles. This data was collected in different traffic flows namely 1500 and 2500 vehicles/hours/lane as these were respectively the default and maximum available settings in the AIM4 simulator. This simulation was executed with vehicles having a maximum speed limit of 90 KM/hr.

The knowledge inferred from this statistics is that the chance for a vehicle getting witnessed by other vehicle increases when the population of vehicles on road increases. From Figure 5.1, we can see that in traffic flow of 2500 vehicles/hour/lane have more vehicles getting verified by the peer vehicles than in the 1500 vehicles/hour/lane traffic flow.

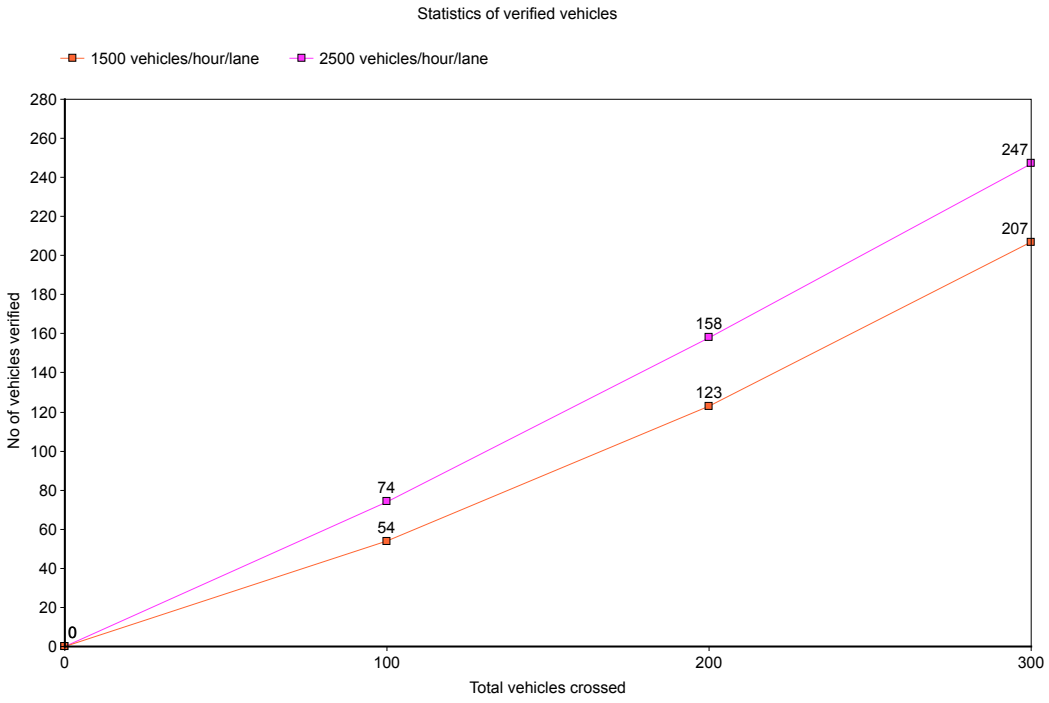

Figure 5.1: Verified vehicles on different traffic flow

6
Discussion

The chapter 5 presented the result from the experimentation based evaluation on our implemented solution. In this chapter, we will discuss and analyze the advantage and disadvantage of our solution.

The blockchain network of five nodes has created a hot-standby redundancy by which the possibility for the down time is reduced. When the nodes fail, there is no outage time in the system until at least one node is running. According to the blockchain technology [4], the hacker can control the network when more than half of the nodes in the blockchain are compromised. So we have the advantage of having five nodes in the network thus the hacker has to compromise at least three of them. The data at the blockchain nodes are encrypted and this means the hacker cannot simply tamper the data without stealing the password. The authorization test proved that the blockchain prohibit impersonation as the not registered account (vehicle) and invalid password will be caught in the blockchain. Also, the messages are digitally signed by the registered vehicle using its private key. This makes it impossible to be altered and disguised as being sent from a legit vehicle.

From the performance evaluation, we found that the average time depends on the machine used. To make this proof of concept suitable for a real-time situation, we must have appropriate computation power for the blockchain infrastructure. We have used the Computer II to run our simulation of five blockhain nodes and seen the improvement in the average response time. It should be noted that in a real time usage, five blockchain nodes are not needed to be run on a same computer, thus each would be running on dedicated individual machines. Currently, GPU manufacturers like Nvidia are producing specialized processors for automotive industry [30], one of such processor would be suitable for running our blockchain node in industries. From the result of the load sharing evaluation, we can see that the nodes of the blockchain network share the work through contributing computation in the block formation. This is further validated from the Ether spent for their contribution.

The crowd-sourced environment information is useful in identifying the physical presence of a vehicle on the road. By utilizing this information, the IM can be sure that it is not serving some non-existing vehicle (hacker). As this information is from the cars, they can rely on them. One of the alternatives to this solution is that the traffic department is deploying surveillance sensors on the roadside near the junction and identify the vehicles on the traffic line. Although one could question about the car's reliability on these sensors, it should be noted that it is comparably less reliable than the information from the car's sensor.

Privacy and Security: In our designed solution, the public key encryption is the security that protects the messages from tampering and ensures the integrity. The private keys of the individual vehicles can be stored at the vehicles in encrypted form with a user-set password. Even though the hacker gets hands-on that encrypted key, it will be of no use without the password. Another security factor in our solution is that only the registered vehicle can communicate to the IM and it can ignore request messages from a Denial-Of-Service attack. Regarding the privacy, the cars are sharing their position and the speed which the car owner may not like, but by using the blockchain based IM, the data stored is only accessible by the traffic department. Also, the blockchain network is a private network and no new unknown node can join and access those data. The additional advantage for the traffic department is that it can use these data to validate if the vehicle follows the traffic rules by confirming the speed limit.

What is the level of availability improvement in the C-ITS through the redundancy from blockchain technology ?

As the failure of a single blockchain node does not affect any other nodes, the possibilities for a whole system downtime is reduced by 5 times. Unless all the nodes are down, this remote service will be up and available to serve. One of the things to be noted is that the blockchain network becomes compromised if the hacker can control more than half of the nodes. So when the number of nodes in the network reduces, it is less secure.

How can the environment sensor on autonomous vehicles be used for crowd-sourced environment surveillance ?

The powerful sensors available on an autonomous vehicle can identify the environment elements like pedestrians, vehicles, obstacles etc and these information can be shared with the Intersection Manager. Then it will be helpful for processing the request from the vehicles for crossing the junction. Also, these information can be used to possibly ignore the fraudulent requests and act against the hacked vehicles. From our experimentation, we found that the traffic flow increases the chance for a vehicle being witnessed by other vehicle on road.

7

Related Work

Dresner et al. have designed several versions of Autonomous Intersection Manager and one notable among them is the Unmanaged Intersection Control [3]. Unlike the other models, this Intersection model has no central controller and every communication is peer-to-peer. The protocol is also similar to the AIM that is described in the Section 2.1 of this document. The difference in this version of AIM is that there is no central coordinator present. Instead the cars cooperate to decide on who should cross the road junction first. On security aspects, this peer-to-peer protocol depends on the digital signatures on communication messages. A problem in this system could be that the digital signature feature will be of no use when the vehicle gets hacked and can generate digitally signed valid messages. If there is a possibility of hacked vehicles on the road, then this peer-to-peer AIM is not a secure design, and the trust among the vehicles will be doubtful. Moreover, this could also be vulnerable to the Byzantine general problem [15].

Benjamin et al. exploited the benefits and way of using blockchain technology in Vehicular Ad-hoc Network (VANET). In their literature [18], they explained that this idea could replace the centralization in the vehicular network system. The similarities with our thesis are decentralizing the system, and the same blockchain framework Ethereum to demonstrate the possibilities. The dissimilarity is the system proposed is self-managed by the vehicles unlike the traffic system in our project.

Yuan et al. framed a protocol and listed the difficulties in the blockchain based Intelligent Transportation System in their literature [7]. As a part of this work, a case study on a ride-sharing app that runs on blockchain technology is done, and it is called "Blockchain version of Uber". This app proved that vehicles could be a part of blockchain using the smartphones and cloud computing.

Dorri, Ali, et al. have worked on the use of the blockchain in the smart functions of the car [20]. They have utilised the security benefits of the blockchain to frame an architecture for a secure transaction. Similar to our solution, the public key encryption is used in their design for the security and a smart contract for paying vehicle insurance digitally. Another similarity with our thesis is to develop a decentralised solution and deprecate the old centralised solution.

Brambilla et al. have written a literature about deploying blockchain in centralised location-based services [21]. These services run depending on user provided geographical position. Therefore it requires a central coordinator to validate the user-provided location. They have proved that using the blockchain, it is possible to bring trustworthiness to the service and privacy of their user data. Like our thesis, their design uses the crowd provided information to prove the presence of an object. These objects append the certificate of approval from the witnesses, and similarly, we use the data from the crowd regarding the location.

Dorri, Ali, et al. also have made another literature on optimised Blockchain for IoT [22]. As there are limitations on the processing capability of the IoT devices and overlays in the blockchain, they have tried to simplify and reduce the overlays. In their design, they skip checking the validity of the block and trust on the peers in the network. This trust system works on a parameter called trust rating of the peers.

Ali, Kashif, et al. have designed a Crowdsourced Intelligent Transportation System that uses the information from the sensors of the vehicles [23]. These data are sent with a human intervention by approving from smartphones. It will be sent from the same phone to a central processor. Like this thesis, their solution is an alternative for the deploying roadside sensors. Their model uses a primary coordinator to provide information from these sensor data. Also, their model includes a POC about the routing application based on road conditions that are notified by the vehicle sensors using mobile phones.

8

Conclusion

We have started this project with an intention to improve Intelligent Transport System by converting the centralized system to distributed system. Though the blockchain technology is booming for being a secure distributed platform and the vehicles are becoming smarter for cooperative functions, we have experimented to adapt this technology to an ITS system in the presence of autonomous cars on the road. From this experimentation,it was demonstrated that the blockchain could make a secure and distributed traffic system. It has resulted in highly available service by using replication from the blockchain technology

The intersection management is improved to be robust against the hacked vehicle's lies, and this is made possible by using the powerful sensors on the autonomous vehicles. Any autonomous vehicle would require environment sensing capabilities for their self-driving function. We have used those sensor data from the crowd to have the digital environment at IM. Our experiments proved that these sensor information could be used by the intersection manager to validate the vehicles as legit and be robust. As we use the vehicle's sensor for the traffic functioning, it can be considered as a crowd-sourced system.

Future Work

As of now, in our solution design, we are sharing the location, time and identity of the peer vehicles. In future work, we could improvise by sharing speed of the vehicle, type of the vehicle and other relevant data. These could make the digital environment information more realistic. Our project concentrated on the road intersection management and these ideas could be used to other similar roadside scenarios such as lane joining, roundabouts, pedestrian crossing etc. As another future work, the automobiles could be equipped with powerful machines that they can be a part of blockchain network and the traffic infrastructures could be completely removed. As our solution is based on cryptocurrency, we could use the incentive model from it. The sharing of the environment information by the vehicles can give them the reduction in their the toll/road tax payment. It would encourage the vehicles to share their sensor information to the IM.

Bibliography

[1] Kurt Dresner, and Peter Stone. "A multiagent approach to autonomous intersection management." Journal of Artificial Intelligence Research (JAIR), Volume 31:pp.591-656, 2008.

[2] Tsz-Chiu Au, Shun Zhang, and Peter Stone. "Autonomous intersection management for semi-autonomous vehicles." Handbook of Transportation. Routledge, Taylor & Francis Group, 2015.

[3] Mark VanMiddlesworth, Kurt Dresner, and Peter Stone. "Replacing the stop sign: Unmanaged intersection control for autonomous vehicles." Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems, Volume 3:pp.1413-1416, International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[4] Satoshi Nakamoto. "Bitcoin: A peer-to-peer electronic cash system.", Available at `https://bitcoin.org/bitcoin.pdf`, 2009.

[5] Scott Nadal and Sunny King. "Peercoin–Secure & Sustainable Cryptocoin.", `https://peercoin.net/whitepaper`. Online accessed Aug 2017 .

[6] Stephen Otunba, Graphical Road Intersection, `http://www.cs.bowiestate.edu/sharad/vrlab/traffic.html`, Online accessed May 2017.

[7] Yuan Yong, and Fei-Yue Wang. "Towards blockchain-based intelligent transportation systems." Intelligent Transportation Systems (ITSC), IEEE 19th International Conference, 2016.

[8] Laurens Hobert, Andreas Festag, Ignacio Llatser, Luciano Altomare, Filippo Visintainer, and Andras Kovacs. "Enhancements of v2x communication in support of cooperative autonomous driving.", IEEE Communications Magazine, 2015

[9] Yevgeniy Brikman. `http://www.ybrikman.com/writing/2014/04/24/bitcoin-by-analogy/`, Online accessed May 2017.

[10] Manuel Fuenfrocken, and Wolfgang Schulz. "How automotive software architectures could benefit from Bitcoin.", Workshop on Automotive Systems/Software Architectures, 2016.

[11] Vitalik Buterin. "Ethereum: A next-generation smart contract and decentralized application platform." `https://github.com/ethereum/wiki/wiki/White-Paper#ethereum`, 2014.

[12] Solidity, `https://solidity.readthedocs.io/en/latest/`, Online accessed May 2017.

[13] Kurt Dresner, AIM4 Simulator, `http://www.cs.utexas.edu/~aim/`, Online accessed May 2017.

[14] Nicholas D'Andrea, EthereumJS-TestRPC, https://github.com/ethereumjs/testrpc, Online accessed May 2017.
[15] Leslie Lamport , Robert Shostak, and Marshall Pease. "The Byzantine generals problem." ACM Transactions on Programming Languages and Systems, Volume 4, Issue 3:pp.382-401, 1982.
[16] Vaishnavi Vijay, and William Kuechler. "Design research in information systems.", Association for Information Systems, Available at http://desrist.org/design-research-in-information-systems/, 2004.
[17] Storey Neil. "Safety critical computer systems." Addison-Wesley Longman Publishing Co., Inc., 1996.
[18] Benjamin Leiding, Parisa Memarmoshrefi, and Dieter Hogrefe. "Self-managed and blockchain-based vehicular ad-hoc networks." Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2016.
[19] Elizabeth Weise, Tesla hack story, https://www.usatoday.com/story/tech/2017/07/28/chinese-group-hacks-tesla-second-year-row/518430001/, Online accessed Oct 2017.
[20] Dorri Ali, Marco Steger, Salil Kanhere, and Raja Jurdak. "BlockChain: A distributed solution to automotive security and privacy." arXiv preprint arXiv:1704.00073, 2017.
[21] Giacomo Brambilla, Michele Amoretti, and Francesco Zanichelli. "Using Block Chain for Peer-to-Peer Proof-of-Location." arXiv preprint arXiv:1607.00174, 2016.
[22] Dorri Ali, Salil Kanhere, and Raja Jurdak. "Towards an Optimized BlockChain for IoT." Proceedings of the Second International Conference on Internet-of-Things Design and Implementation. ACM, 2017.
[23] Kashif Ali, Dina Al-Yaseen, Ali Ejaz, Tayyab Javed, and Hossam Hassanein. "Crowdits: Crowdsourcing in intelligent transportation systems." In Wireless Communications and Networking Conference IEEE, pp. 3307-3311, 2012.
[24] Ladislav Kristoufek. "BitCoin meets Google Trends and Wikipedia: Quantifying the relationship between phenomena of the Internet era.", Scientific Reports 3:3415, 2013.
[25] Elizabeth Weise, Emergency Brake a system that saves lives, http://www.volvotrucks.com/en-en/news/volvo-trucks-magazine/2017/jul/tech-focus-emergency-brake.html, Online accessed Dec 2017.
[26] Blockchain Frameworks and Tools https://www.opensource-it.com/software/category/domains/development-frameworks-tools/blockchain-frameworks-and-tools/, Online accessed Dec 2017.
[27] Sun Zehang, George Bebis, and Ronald Miller. "On-road vehicle detection: A review." IEEE transactions on pattern analysis and machine intelligence 28.5, 2006.
[28] Rob Yedlin, State of the Dapps, https://www.stateofthedapps.com, Online accessed Dec 2017.
[29] ETSC, Press releases, http://etsc.eu/european-parliament-demands-higher-safety-standards-for-new-cars-vans-and-lorries/, Online accessed Dec 2017.
[30] Drive PX-series, https://en.wikipedia.org/wiki/Drive_PX-series, Online accessed Dec 2017.

A

Appendix 1

Filename: **initMiners**

```
geth --datadir $ethereum_home/node1 init $ethereum_home/genesis.json
geth --datadir $ethereum_home/node2 init $ethereum_home/genesis.json
geth --datadir $ethereum_home/node3 init $ethereum_home/genesis.json
geth --datadir $ethereum_home/node4 init $ethereum_home/genesis.json
geth --datadir $ethereum_home/node5 init $ethereum_home/genesis.json
```

Filename: **cleanAndBuild**

```
cd $ethereum_home/node1
shopt -s extglob
rm -rf !(keystore)
cd $ethereum_home

cd $ethereum_home/node2
shopt -s extglob
rm -rf !(keystore)
cd $ethereum_home

cd $ethereum_home/node3
shopt -s extglob
rm -rf !(keystore)
cd $ethereum_home

cd $ethereum_home/node4
shopt -s extglob
rm -rf !(keystore)
cd $ethereum_home

cd $ethereum_home/node5
shopt -s extglob
rm -rf !(keystore)
cd $ethereum_home
./initMiners
./runMiners
./getEnode
```

Filename: **runMiners**

```
gnome-terminal --tab -e 'bash -c runMiner1;bash'
--tab -e 'bash -c runMiner2;bash'
--tab -e 'bash -c runMiner3;bash'
--tab -e 'bash -c runMiner4;bash'
--tab -e 'bash -c runMiner5;bash'
```

Filename: **genesis.json**

```
{
    "config": {
        "chainId": 15,
        "homesteadBlock": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "difficulty": "0x1",
    "gasLimit": "1000000000000",
    "alloc": {
        "44d5610ad55f2a5fbdef950e233921c5e5272c44": { "
           balance": "1000000000000000000000" },
        "422a362c728a07356957221668f52d304758ad74": { "
           balance": "1000000000000000000000" },

        "f74a21fba87fcd66853df3986d6834913ec047a1": { "
           balance": "1000000000000000000000" },
        "1c97faee32979415a5cd35c3d8e3049801523608": { "
           balance": "1000000000000000000000" },

        "73e7d57ee14fa4d9f2500d5c231cc4e6550ff2f7":{"balance
           ": "1000000000000000000000" },
        "64efa595aada142b75eaa3f6cb8a512ddef01a90":{"balance
           ": "1000000000000000000000" },

        "aa8ea13b2e87a9685af57521a8fdebd44bc53f9a": { "
           balance": "1000000000000000000000" },
        "e5b8ff166493598e5f35807392263129bff33307": { "
           balance": "1000000000000000000000" },

        "737281faf12ddaa139df8c45d3f8979986b841a9": { "
           balance": "1000000000000000000000" },
        "1289f88fcfa6b6de886bd9c846ccc3a8079fcbef": { "
           balance": "1000000000000000000000" }
    }
}
```

Filename: **runMiner1**

```
geth --datadir $ethereum_home/node1 --unlock 0 --password
    $ethereum_home/password --port 30301 --networkid 1234 --
    nodiscover --rpc --rpcapi db,eth,net,web3,personal --
    cache=1024  --rpcport 8545 --rpcaddr 127.0.0.1 --
    rpccorsdomain "*" --syncmode=fast --preload "
    $ethereum_home/mineOnDemand.js" console
```

Steps to start 5 node Blockchain:

1. Run "bash initMiners" in terminal.
2. Run "bash cleanAndBuild" in terminal.
3. Run "bash runMiners" in terminal.