

B2VDM: Blockchain Based Vehicular Data Management

Rohit Sharma and Suchetana Chakraborty

Department of Computer Science and Engineering,

Indian Institute of Information Technology Guwahati, Assam, India 781015

Email: rosha1401042@gmail.com, suchetana@iiitg.ernet.in

Abstract—Smart interconnected vehicles generate a huge amount of data to be used by a wide range of applications. Although cloud based data management is currently in practice, for many applications serving road safety or traffic regulation, it is utmost important that applications access these data at the site itself for improved quality of service. Road side units (RSUs) play a crucial role in handling these vast amount of vehicular data and serving the running applications in turn. In this current era of edge computing, in-place data access is also proven to be advantageous from cost point of view. As multiple applications from different service providers are interested to access different fragments of these data, a robust access control mechanism is needed to ensure desired level of security as well as reliability for these data. In this paper, we introduce B2VDM, a novel architecture for vehicular data management at RSUs, that provides a seamless access control using Blockchain technology. The proposed B2VDM framework also implements a simple load distribution module, which maintains the reliability by minimizing the number of packet drops at a heavily loaded RSU during peak hours. An extensive evaluation using Ethereum Blockchain validates the effectiveness of the proposed architecture.

I. INTRODUCTION

Recent advancement in the field of Information and Communication Technologies (ICT) and Intelligent Transportation Systems (ITS) has conceived a new research scope to explore Big Data solutions [4] for vehicular ad hoc network (VANET) [2]. With more number of smart (on board unit-enabled) vehicles on road and installation of robust road side infrastructure, the data generated from vehicular network is increasing day by day. Moreover, a wide spectrum of various applications and their different quality of service (QoS) requirement add to the dimensions of these data. In a typical VANET scenario, interconnected smart vehicles generate data and forwards them to the RSUs or cloud servers either directly (V2R communication) or through multi-hop path (V2V communication) using any underlying communication technology like WiFi, IEEE 802.11p or 3GPP/LTE [9], [11]. Depending on the type of applications running or the set of services subscribed, data packets generated by the vehicles may include attributes like speed, location, direction, noise, distance, or safety information and multimedia data. These huge amount of heterogeneous data are processed and analyzed at the back-end servers to infer knowledge and trigger actions so that to offer improved transport services. Applications that make use of these data mostly offer services towards road congestion detection and traffic management, environment

and road monitoring, smart navigation, parking, safety and infotainment.

Road side Units (RSU) play a vital role in disseminating this vast amount of data from the VANET platform to the back-end servers. Although certain applications support vehicular data to be directly sent to the cloud, for other applications it is more convenient and cost-saving to manage the data stream at the RSUs itself. RSUs are placed optimally in an urban area and act as the intermediate platform for vehicular data collection. Usually, these are the static components of a VANET architecture and connected with each other through high-speed backbone infrastructure. A simple vehicular network has been shown in Figure 1(a), where the corresponding communication graph comprising the set of RSUs has been shown in Figure 1(b).

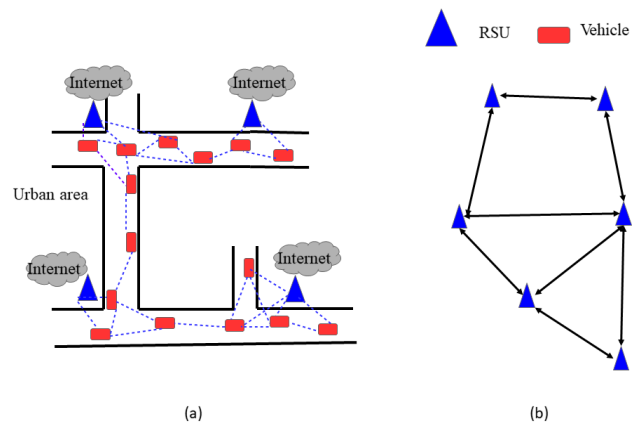


Figure 1. A simple VANET scenario with RSU forming a network

Efficient management of vehicular data mainly depends on robust and smooth collection procedure. Following are the major challenges associated with vehicular data management.

- 1) *Big data*: Highly dynamic nature of vehicular network with variable density of vehicles generate large volume of data. An increasing number of smart vehicles along with escalating subscription to multiple application services add to the volume. Moreover, escalating advancement in the field of ICT has triggered fast delivery of these data to the RSUs adding up to the complexity.
- 2) *Heterogeneity*: Different applications focus on different attributes in order to provide the promised services. This

in turn generates multimodal data with attributes having variable type, size, value, dimensions, scale, frequency and meaning.

- 3) *Security and privacy issues*: Smart vehicles have an increasing number of autonomous driving functions. A malfunction due to a security breach (e.g., by installing malicious software) could endanger the safety of on-board travellers as well as other people around. Privacy is another critical factor associated with vehicular data and it is completely justified that the service provider operates with limited access to data that are relevant.
- 4) *QoS requirement*: Some real-time applications put hard threshold on delay for processing these data, where as for another set of applications reliability criteria remains critical. Different applications impose different QoS requirement on the collection and processing process of vehicular data stream.

The collection framework should take care of all these factors to guarantee desired level of performance for all running applications. Moreover, connected RSUs share a coherent spatio-temporal relationship with respect to data item generated from passing by vehicles. This increases the chances of redundant data delivery and handling these data in such a distributed environment can lead to inconsistent view. In this paper, we propose a reliable, secure yet simple architecture for vehicular data management using Blockchain technology. The rest of the paper is organized as follows. Section II provides a brief discussion about the existing data management protocols for vehicular network while Section III describes the working of the proposed architecture. Results and analysis have been provided in Section IV followed by conclusion at the end.

II. RELATED WORK

There has not been much work in the literature addressing the problems of vehicular data management. Centralized broker based communication models have been adopted by most of the current smart vehicle architectures where all the access control for vehicles or RSUs happen only at the cloud servers by a central authority. In case, the central authority is compromised, the entire data gets exposed. This model is unlikely to scale as large number of vehicles are connected. Additionally, the cloud servers will remain a bottleneck and a single point of failure that can disrupt the entire service. Further, some of the applications need to access the RSU data at the field itself to save time as well as cost. Centralized database providing access control facility will not serve the purpose. Also, a distributed solution may suffer from inconsistent view.

Illari et al. [7] have provided a comprehensive study on data management for vehicular networks. They have explained various challenges of data management in VANET including information relevance determination, data dissemination and aggregation, resource management, etc. Their elaborate work includes a detailed discussion on managed data types and representation, query models, dissemination techniques, competitive resource management strategies and other related topics.

Interestingly, according to their finding the complexity of the environment makes experimental and comparative evaluation of these frameworks very challenging.

Blockchain being a powerful technology in providing a decentralized solution towards secure transaction management, many applications starting from cryptocurrencies to Internet of Things have adopted this framework to enforce trust. Vehicular network having many challenges also aims at exploring the advantages of this cutting-edge technology as few works have already adopted the concepts. Authors in [17] have introduced an ITS oriented layered Blockchain model. They have identified few research directions including decentralized autonomous transportation, software-defined trust for ITS ecosystem, privacy and security for ITS data etc., that can utilize their proposed model. In [14], authors have proposed a blockchain based framework to create a trust environment for intelligent vehicles to support data sharing in a peer-to-peer manner. Dorri et. al. [6], have focused on security and privacy challenges of VANET users. They have proposed a Blockchain based architecture to preserve privacy of vehicle users. They have considered wireless remote software update and other emerging vehicles services as potential use cases. In [8] Ethereum based decentralized and self-managed VANET system has been proposed with challenge-response-based authentication.

Similar to security and privacy, reliability is another important factor in maintaining the data quality. It is imperative to minimize the rate of data packet loss at an overloaded RSU during busy hours. Load distribution is one of the important challenges of distributed environment considering the overhead of data transfer including the cost of consensus among the participants. Various cloud-based load balancing techniques have been discussed in the literature [12], [13], [1]. In general, load distribution algorithms can be static or dynamic on the basis of system criteria. Static algorithms require a prior knowledge of the system resources and hence not susceptible to run-time changes. In dynamic algorithms, the lightest server in the whole system is searched and preferred for load balancing. Again, with respect to initiation criteria, load distribution can be either source-initiative or server-initiative. As vehicular environment supports a wide range of applications, the amount of data can be huge along with the variation in generation rate. Considering optimal placement and dense distribution of RSUs along with high-speed infrastructure network support, it is still challenging to come up with an optimal solution of load scheduling. A cooperative load balancing for RSU based VANET environment has been proposed in [3]. With the prior knowledge of RSU position and vehicle heading direction, their scheme aimed at scheduling the overloaded requests from the source RSU to a neighboring RSU in the same direction. The authors claimed to maximize the overall performance by preferring edge-RSUs than junction-RSUs for request transfer.

In this paper, we propose a novel architecture for vehicular data management at RSUs while keeping the focus mainly on efficiency, cost and security aspects of such data manipulation. To the best of our knowledge, the proposed B2VDM frame-

work is the first effort to address both the problems of access control and load distribution in a RSU network for efficient data management.

III. B2VDM FRAMEWORK

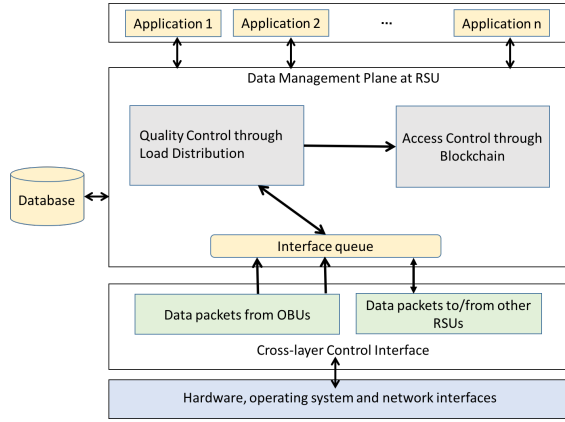


Figure 2. Architecture of proposed B2VDM middleware

A. System model:

The proposed B2VDM works as a middleware in between the application layer and the network layer of RSUs as shown in Figure 2. The proposed middleware is responsible for both quality control and access control. The quality controller mainly focuses on *reliability*, which is enforced by distributing the data load to nearby RSUs using a suitable load sharing algorithm. This actually controls the loss of vehicular data due to buffer overflow at a heavily loaded RSU during busy hours. Access control is achieved using Blockchain technology. RSUs act as the nodes in blockchain and control the interaction between database and multiple service providers with respect to read or write permissions. It is safely assumed that RSUs are equipped with sufficient amount of storage space and computational resources. The interface queue as shown in Figure 2 is responsible for managing both the in-flows of data received from the vehicles (on-board units) and other RSUs (as a result of load distribution). As the data packets are read from the queue, they first pass through the quality controller and then the access control transactions are included into the blockchain where as data itself is stored into local database. From time to time, both the database and the blockchain entries are archived and transferred to some remote server while the local history is removed to save the space. Multiple applications can run on top of this proposed middleware and can perform read or write actions on these stored data at a RSU as they are approved through access controller. The detailed functionality of quality controller and access controller of B2VDM middleware is discussed below.

B. Quality Control:

Variable rate of generated data traffic from huge amount of connected vehicles puts uneven load on different RSUs in close vicinity having high spatio-temporal co-relation

Algorithm 1: Algorithm for load distribution

Let \mathcal{N}_i be the neighbor set of a RSU R_i such that $\forall R_j \in \mathcal{N}_i, \delta(R_i, R_j) < \mathcal{X}$, where δ is the euclidean distance between any two points and \mathcal{X} denotes a predefined distance threshold;
Let \mathcal{T}_i be the threshold load for interface queue at R_i ;
Let C_{Max} be the maximum capacity of the interface queue;
if $C_{Max} > C_{curr} \geq \mathcal{T}_i$ **then**
 $\forall R_k \in \mathcal{N}_i$ send *REQUEST*(load) to R_k
 if $\exists R_j | \mathcal{T}_j > \mathcal{T}_i \forall R_l \in \mathcal{N}_i$ and $C_{curr}[R_j] + load \leq \mathcal{T}_j$ **then**
 | $R_{ALT} \leftarrow R_j$
 else
 | repeat search with lower load;
 end
end

with respect to data. Furthermore, multiple applications from various service providers try to access these data at the same time. This results in overflowing interface queue at the RSUs and poor quality of service due to high rate of packet loss. Even with the queue with largest capacity, this problem can not be avoided due to uncontrollable inflow traffic. Therefore, the proposed B2VDM middleware aims to address this problem by including a quality controller module which essentially performs the load distribution among nearby RSUs. According to the proposed work, every RSU maintains a threshold value for load, which is a function of queue capacity. If the current load is greater than the set threshold, then load distribution algorithm will trigger and distribute the excess load to one of the available RSUs nearby. The distance parameter for establishing the neighborhood relation is purely scenario specific. The threshold is dynamic in nature as it is calculated on the basis of inflow and outflow of data packets at a RSU. As the ratio of inflow and outflow data rates increases the threshold value for a RSU also increases. The selection of a candidate RSU from among the set of neighboring RSUs for load distribution is a repetitive process and follows a greedy strategy as stated in Algorithm 1. Ideally, the initiator searches for the neighbor having maximum amount of free space in queue by broadcasting the excess load information through *REQUEST* message. If fails, the search process continues as the initiator RSU performs ACK/NACK based handshaking with rest of the neighboring RSUs. Let α denote the ratio between input rate and output rate. We have taken α time the maximum queue capacity as the amount of data packets needs to be stored at initiator RSU, while searching for an alternate RSU. Threshold is calculated using the given formula:-

$$\alpha = \frac{\text{Input Rate}}{\text{Output Rate}}$$

$$\text{Threshold} = C_{Max} - \frac{C_{Max}}{\alpha}$$

Requester	Request For (read/write /both)	App_ID	Permission (Deny/Allow)
-----------	--------------------------------------	--------	----------------------------

Figure 3. Policy Header

App_ID	Action (read/write)	Timestamp	Data Item Descriptor
--------	------------------------	-----------	-------------------------

Figure 4. Transaction Header

C. Access Control:

An uncontrolled access to the RSU data by multiple applications from different service providers opens up the possibility of various security threats by compromising confidentiality, integrity and authenticity of data. An authorized access (read/write action) to a data item is what the proposed middleware aims to enforce as the main security feature. As multiple service providers may wish to access different fragments of data from different RSUs, an independent effort of keeping records for access history would not be able to enforce globally consistent access control in a distributed environment. In this regard, a blockchain can be used to seamlessly provide a consistent view of authorized transactions in a tamper-proof manner. A transaction is essentially a tuple describing the access information as shown in Figure 4.

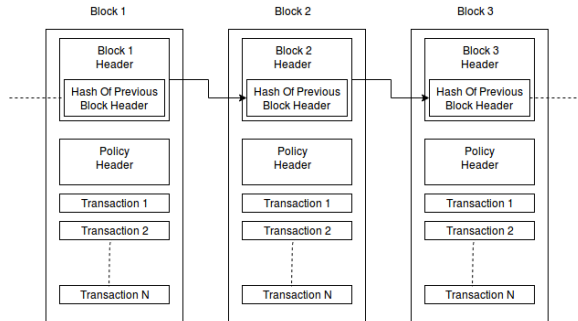


Figure 5. A simple blockchain for access control

A blockchain can be explained as a distributed database that keeps track of the growing list of blocks that are linked to each other as shown in Figure 5. Each block contains a set of transactions that are by principle immutable and secure. For multiple parties with conflicting interests, blockchain provides a decentralized platform of logging the transactions among the involved parties in a transparent manner. By exploiting these advantages of blockchain, the proposed B2VDM framework provides a robust platform to log the transactions in an efficient manner as well as enforces seamless access control. When a data item is accessed by an application at some RSU, it is logged in a form of transaction, a series of what makes a block.

Algorithm 2: Consensus protocol at every RSU

```

Let  $A : \{A_1, A_2, A_3, \dots, A_n\}$  are application IDs;
Let  $R : \{R_1, R_2, R_3, \dots, R_n\}$  are RSU IDs;
Let  $P : \{read, write\}$  are access permission for application in A;
Let  $T : \{T_1, T_2, T_3, \dots, T_n\}$  are transaction IDs;
Let  $B : \{B_1, B_2, B_3, \dots, B_n\}$  are block IDs as  $B \in \mathcal{B}$  be the blockchain;
Let  $A_i$  places an access request  $x \in P$  at  $R_i \in R$  such that  $T_i = \{A_i, x, \tau, D_i\}$  where  $\tau$  is the time stamp of access and  $D_i$  is the corresponding data item;
if  $A_i \in A$  then
  if  $\{x = write \text{ and } P[A_i, D_i] = write\}$  then
    if  $T_i \in B_i[R_i]$  then
      return False
    else
      return True
    end
  end
  if  $\{x = read \text{ and } P[A_i, D_i] = read\}$  then
    return True
  else
    return False
  end
end

```

Each transaction needs to be validated by other RSUs through consensus before being added to the blockchain. A transaction is checked against various security parameters like application ID, whether the specific application has a permission to access the data or not. Each block will be associated with a policy header as shown in Figure 3. Policy header maintains a list of all permissions and application IDs along with the type of action (read/write) an application is allowed to do with the corresponding data item. A valid transaction added to the blockchain reflects the authenticity of the authorized access.

1) *Security Enforcement*: These are three stepped security measurements that need to be adopted in order to enforce access control using blockchain technology.

A *Encryption* Firstly, each RSU generates its private and public key. The blockchain maintains the public key of all RSUs in the network and helps in establishing the communication between any two RSUs. A RSU adds a transaction into the blockchain by signing it with its public key so that to ensure the confidentiality.

B *Digital Signature* Secondly, each RSU digitally signs(encrypt with its private key) the transaction to ensure the integrity and non-repudiation of the transaction. With digitally signed message, receiver can easily find, that the message is not tampered, and the sender of the transaction is a valid roadside unit in the network.

C *Verification* Lastly, the receiver, after receiving messages, identifies the sender by verifying the digitally signed message. After verification, the receiver decrypts the message.

2) *Consensus Protocol*: The validity of a transaction to be added into the blockchain is ensured using consensus protocol. The validation happens by majority voting in the RSU network. The physically constrained layout of road network shows a strong spatio-temporal co-relation among the vehicular data. Therefore, for a specific location at a specific time, data generated by vehicles are co-related only for a set of closely located RSUs, and thus, for either load distribution or consensus only a set of close neighboring RSUs will be sufficient to take the decision. We have adopted the principle of Proof of Existence(POE), similar to the one used for validation of decentralized documents [5].

a) *Proof of Existence*: The road network exploits the advantage of using private blockchain and offers transparent platform for access control without the intervention of any third-party coordinator. Whenever an access request is placed at a RSU, it first checks the existence of such a transaction in the blockchain. In one way, it avoids redundant transaction to be stored into the blockchain, on the other way, it ensures that only the authorized applications are accessing the concerned data. This is how the mutual conflict of accessing data among multiple service providers can be resolved while maintaining the desired privacy parameters for vehicular data at the same time. The proposed consensus has been described in Algorithm 2.

IV. RESULTS AND ANALYSIS

The performance of the proposed architecture(B2VDM) is evaluated in two parts, one for access control and other for load distribution.

A. Access Control

We implemented the vehicular data management as a smart contract on Ethereum blockchain. A smart contract is a protocol for digital facilitation towards enforcing a conflict-free agreement among multiple parties without the need of third-party coordinator. Ethereum uses smart contract on top of blockchain so that developers can write programs on blockchain. In other words, using smart contract, we can use Ethereum as a computing platform. We have used Remix platform to implement the smart contract in Solidity language. The miner nodes include an authority node, and the set of RSUs. The working of proposed consensus protocol has also been checked as a part of the same contract.

1) *Transaction Management*: The set of transactions as described in Figure 4, are stored in blocks of a blockchain as tuples. The complexity of this part of architecture depends upon the blockchain transaction rate as most of the current blockchain architecture including Ethereum has very high transaction period of almost few seconds [15]. Usually all data-rich applications heavily suffer from bad performance due to this low transaction rate. However, in past few years with the improvement in Blockchain technology various new platforms like BigChainDB have been developed, in which transaction period has been decreased significantly to perform almost 1000 transactions per second. In our proposed architecture the

transaction complexity mainly depends upon the consensus protocol similar to many other blockchain architecture.

2) *Consensus Protocol*: Results of consensus protocol are depicted from Figure 6 to Figure 8. In consensus protocol, we are checking whether a given application is authorized to access a specified data item. This can be done in constant time as we can store the permissions by indexing it on ID of entity asking permission for a transaction. Figure 6 depicts the successful execution of a transaction as a new node is added to the blockchain network. Figure 7 shows a failed transaction due to unauthorized access, where as a successful transaction for authorized read access is shown in Figure 8. We have implemented the protocol as an application on Remix due to which *execution failed* is shown as a rejection to a transaction for being valid, given in Figure 7. Redundancy can be removed by restricting a transaction to be added to the blockchain when it is already there. Figure 9 shows how a transaction fails on an attempt to add the redundancy. The validation of transaction is shown under *status* section.

B. Load Distribution

We have implemented this part in Java by creating multiple instances to represent a set of RSU. For a small connected graph of 10 nodes with maximum degree to be 4, each node represents a RSU and an edge between two nodes define the neighborhood relationship. We have not considered the physical distance threshold to define the connectivity as the focus of this work does not lie in evaluating the network-centric QoS parameters like delay, throughput and so on. Also, we assumed that each RSU will have sufficient queue space, which is constant and equivalent over all RSUs. The input data rate has been varied by involving a random component, where as output data rate, which is essentially the service rate, is assumed to be constant. The threshold has been calculated as mentioned in Section III-B. The threshold is proportional to input-output ratio and increases proportionally to the increment in input load. However, it always remains below the maximum capacity of the queue to avoid packet loss, shown in Figure 10. Figure 11 and Figure 12 depicts the variation of overhead with respect to input load. We define the term *overhead* as the ratio of the number of transferred packets to the total number of generated packets. It has been observed that this overhead increases initially with the growth of input-output ratio, and becomes almost constant after a certain point, as shown in Figure 11. On the other hand, if we consider absolute incoming load at a RSU and try to check the variation in overhead, it shows an opposite trend, shown in Figure 12. This actually reveals an interesting fact that a controlled input-output ratio can severely affect the performance of the protocol in terms of amount of data needs to be transferred. A suitable selection of threshold and regulation on output rate can improve the performance of the load distribution.

V. CONCLUSION



In this paper we have proposed a novel architecture, called B2VDM, for vehicular data management, which uses

[illegible]

```
[vm] from:0x4b0...4d2db, to:Implementation.write_to_block(string,string,string) 0x692...77b3a, value:0 wei, data:0x5eb...00000, 0 logs, hash: 0x5a4...79610
```

Details

Debug



status	0x0 Transaction mined but execution failed
from	0x4b0897b0513fdc7c541b6d9d7e929c4e5364d2db 
to	Implementation.write_to_block(string,string,string) 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a 

[vm] from:0xca3...a733c, to:Implementation.Read_data(string, string) 0x692...77b3a, value:0 wei, data:0xc7d...00000, 0 logs, hash:0xe9d...7cf2a

status	0x1 Transaction mined and execution succeed
from	0xca35b7d915458ef540ade6068dfe2f44e8fa733c
to	Implementation.Read_data(string, string) 0x692a70d2e424a56d2c6c27aa97d1a86395877b3a

```
[vm] from:0xca3...a733c, to:Implementation.write_to_block(s
string,string,string) 0x692...77b3a, value:0 wei, data:0x5e
b...00000, 0 logs, hash:0xbd7...bd2c9
```

[Details](#)[Debug](#)

status	0x0 Transaction mined but execution failed
from	0xca35b7d915458ef540ade6068dfe2f44e8fa733c 
to	Implementation.write_to_block(string,string,string) 0x692a70d2e424a56c 2c6c27aa97d1a86395877b3a 

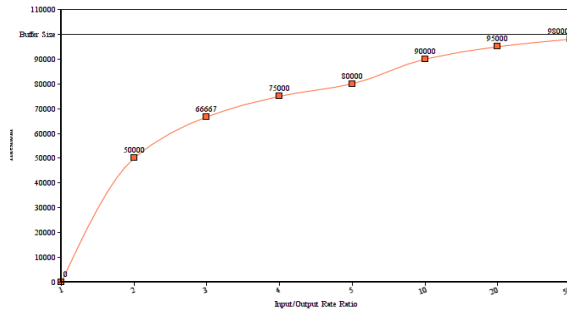


Figure 10. Variation of threshold with input/output ratio

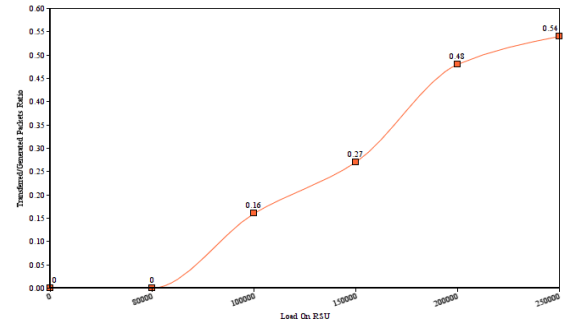


Figure 12. Transferred/generated packet ratio vs. Load on RSU

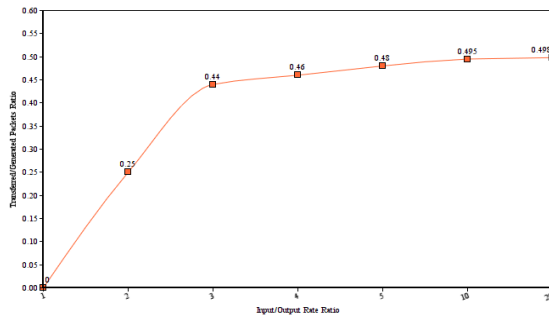


Figure 11. Transferred/generated packet ratio vs. Input/Output Ratio

Blockchain technology for maintaining consensus among distributed RSU network and thus providing security and privacy of data. The consensus used in this work will help in maintaining authenticity of data added by roadside units into the blockchain and thus avoid redundancy. The overall time complexity of consensus protocol depends upon the tuple size and the communication infrastructure used, as the set of RSUs need to validate every transaction. This is also one of the limitations of using Blockchain technology as it consumes a lot of time. However, benefits of using blockchain with respect to having a secure, transparent yet decentralized platform of access control supersedes any such drawback. The proposed protocol also aims to improve the quality of data by adopting a load distribution approach and thus reducing the rate of packet loss. The performance evaluation of the proposed B2VDM architecture verifies and validates the correctness as well as opens up a new possibility of managing vehicular data in a smarter way.

VI. ACKNOWLEDGEMENT

This work has been partially supported by SERB, DST project no. ECR/2017/000813.

REFERENCES

[1] Aayush Agarwal, G Manisha, Raje Neha Milind, and SS Shylaja. A survey of cloud based load balancing techniques. In *Proceedings of*

Int. Conf. on Electrical, Electronics, Computer Science & Mechanical Engg., 27th April-2014, Bangalore, India, 2014.

[2] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[3] G. G. Md. Nawaz Ali, Md Abdus Salim Mollah, Syeda Khairunnesa Samantha, and Saifuddin Mahmud. An efficient cooperative load balancing approach in rsu-based vehicular ad hoc networks (vanets). pages 52–57, 2015.

[4] Juan Contreras-Castillo, Sherali Zeadally, and Juan Antonio Guerrero Ibañez. Solving vehicular ad hoc network challenges with big data solutions. *IET Networks*, 5:81–84, 2016.

[5] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, and Vignesh Kalyanaraman. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2:6–10, 2016.

[6] Ali Dorri, Marco Steger, Salil S Kanhere, and Raja Jurdak. Blockchain: A distributed solution to automotive security and privacy. *IEEE Communications Magazine*, 55(12):119–125, 2017.

[7] Sergio Ilarri, Thierry Delot, and Raquel Trillo-Lado. A data management perspective on vehicular networks. *IEEE Communications Surveys & Tutorials*, 17(4):2420–2460, 2015.

[8] Benjamin Leiding, Parisa Memarmoshrefi, and Dieter Hogrefe. Self-managed and blockchain-based vehicular ad-hoc networks. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, pages 137–140, 2016.

[9] Fan Li and Yu Wang. Routing in vehicular ad hoc networks: A survey. *IEEE Vehicular technology magazine*, 2(2), 2007.

[10] Trent McConaghy, Rodolphe Marques, Andreas Müller, Dimitri De Jonghe, Troy McConaghy, Greg McMullen, Ryan Henderson, Sylvain Bellemare, and Alberto Granzotto. BigChaindb: a scalable blockchain database. *white paper, BigChainDB*, 2016.

[11] Panos Papadimitratos, Arnaud De La Fortelle, Knut Evensen, Roberto Brignolo, and Stefano Cosenza. Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. *IEEE communications magazine*, 47(11), 2009.

[12] Nadeem Shah and Mohammed Farik. Static load balancing algorithms in cloud computing: Challenges & solutions. *International Journal Of Scientific & Technology Research*, 4(10):365–367, 2015.

[13] Amandeep Kaur Sidhu and Supriya Kinger. Analysis of load balancing techniques in cloud computing. *International Journal of computers & technology*, 4(2):737–741, 2013.

[14] Madhusudan Singh and Shiho Kim. Blockchain based intelligent vehicle data sharing framework. *arXiv preprint arXiv:1708.09721*, 2017.

[15] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32, 2014.

[16] G Yan and S Olariu. A probabilistic analysis of link duration in vehicular ad hoc networks. *IEEE Trans. Intell. Transp. Syst.*, 12(4), 2011.

[17] Yong Yuan and Fei-Yue Wang. Towards blockchain-based intelligent transportation systems. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, pages 2663–2668. IEEE, 2016.