Securing Intelligent Transportation System: a Blockchain-based Approach with Attack Mitigation

Le Su[1,2], Yao Cheng[2], Huasong Meng[2,3], Vrizlynn Thing[2,3], Zhe Wang[4], Linghe Kong[4], and Long Cheng[5]

[1] Nanyang Technological University, Singapore le.su@ntu.edu.sg
[2] Institute for Infocomm Research, Singapore
[3] National University of Singapore
[4] Shanghai Jiao Tong University, China
[5] Clemson University, USA

Abstract. In recent years, the intelligent transportation system has become an inseparable component for the smart city ecosystem. Through information sharing among the entities in the ITS such as roadside units, traffic cameras and local controllers, the entire system aims at improving road safety and overall traffic efficiency. However, as drastic amount of data have been exchanged and logged, how could such a system maintain the integrity, authenticity, and non-repudiation of the information for constant verification and audit is one of the biggest challenges. At the same time, as being considered as part of the critical infrastructure, in the smart city ecosystem, the security of the ITS also needs to be addressed carefully. In this work, we propose a novel blockchain-based architecture for ITS, to record actions and information sharing among the entities, at the same time assist in mitigating common types of attacks. Experiments are conducted to assess the overhead of critical cryptographic primitives.

Keywords: blockchain · intelligent transport · data recording · security · attack mitigation

1 Introduction

The Intelligent Transportation System (ITS) has become an inseparable component for the smart city ecosystem. The modern data-driven ITS [1] functions based on rich data collected from multiple sources such as connected vehicles, inductive-loop detectors, sensors, and traffic cameras. Accordingly, ITS runs various algorithms based on these data such as optimal traffic light decision and statistical traffic data analysis. Compared to the traditional transportation system whose objective is to facilitate the orderly traffic flow, the modern data-driven ITS aims at both orderly and timely traffic flow by optimizing the use of the existing transportation infrastructure, enhancing overall traffic efficiency, and improving road safety. As an infrastructure that is closely related to personal safety, the corresponding security, reliability, and stability of an ITS are the prerequisite for all the benefits it brings.

Technically, ITS is a highly distributed system with heterogeneous subsystems in terms of both hardware and software. There is a centralized ITS center for the purpose of overview, management and policy adjustment based on the data from the subsystems under its regime. Under the regime of the ITS center, there are thousands of local controllers deployed out in the field. Local controllers strive to derive actionable intelligence from connected smart devices and sensors. According to the rich information it gathers, the local controller is able to make timely distributed decisions that fulfill the overall aim of the ITS system.

Despite the common threats to an information communication system, ITS faces several severe challenges caused by its distributed and heterogeneous nature. Firstly, ITS is a target with multiple attack vectors. The attacker can choose to compromise the sensors or the local controller to achieve her goal. For example, to cause slow traffic or a bad jam, the attacker can force ITS to come to an incorrect decision by impersonating the sensor, falsifying the data, or tampering with the decision-making algorithm. Secondly, the large number of heterogeneous sensors in ITS poses a great challenge in guaranteeing the data authenticity, which is essential to ITS since the authentic data is the foundation for an optimal decision. For different sensors with various specifications, it requires a specific effort to make sure the provided data is authentic. Thirdly, in case any attack happens, the distributed nature of ITS makes it difficult to carry out an efficient post-security audit. It is a cumbersome task to inspect a large number of distributed devices. In addition, the distributed storages may not be reliable since they are facing the threat of being tampered with.

With the above challenges in mind, we propose *SecBITS*, a *Sec*ured *B*lockchain-based *ITS*. The blockchain intrinsically fits the scenario in the ITS system where the algorithms are executed on the data from the decentralized smart devices and sensors. Briefly, SecBITS is able to record the sensor data and the decisions based on that data in a tamper-proof blockchain. Every sensor data and the corresponding decision are associated with a piece of smart contract, so that the data authenticity and the decision correctness are guaranteed. We also introduce specific contract terms to counter against certain attacks. Our contributions are:

- Propose a novel architecture for ITS based on blockchain called SecBITS which could enhance the ITS security by mitigating certain types of attacks.
- Provide security analysis on the capability of SecBITS in mitigating record tampering, DoS attack, rogue sensor attack, and logic compromising.
- Further carry out experiments to assess the overhead of critical infrastructure primitives such as cryptographic computations.

2 Related Work

Intelligent transport system (ITS) is the modern transportation system that applies information and communication technologies to infrastructures and vehicles to achieve efficient and effective traffic and mobility management. ITS is also referred to as data-driven ITS since it heavily relies on data from sensors (e.g., cameras [2], laser radars [3]) and vehicles [4] to estimate a real-time road

condition based on which optimal traffic-related decisions such dynamic traffic light signal timings are made. In terms of data flow in ITS, vehicles report data through DSRC (Dedicated Short Range Communications) to the RSU (Road-side Unit) [5]. RSUs, as well as various sensors feed their collected data to the local controller, which is a microcomputer located near the intersection. The local controller is responsible for making real-time local traffic light decisions based on the fed data and issuing traffic light command. Such ITS architecture is common in nowadays and more details about it can be found in related literature [1]. Despite the architecture and technique have been widely studied, the security of ITS has rarely been analyzed. The goal of our work is to propose a secured ITS design with the help of the novel blockchain techniques.

Recently, there is a stream of research on the blockchain application to CPS scenarios. Boudguiga et al. [6] analyzed how the use of blockchain can meet the requirements of confidentiality, integrity and availability during the Internet of Things (IoT) updates. In [7] the authors provided a high-level description of the adaptation of blockchain into ITS and furnished a particular case study on ride-sharing. However technical details on how to achieve such a blockchain-based system are lacking and no experiment testing on feasibility have been provided. The authors of [8] discussed vehicular ad-hoc network based on blockchain, similarly, only a high-level description over the concept was provided. Another work [9] studied the problem of using blockchain for simplifying the distributed key management in a heterogeneous vehicular communication system, whereby the communication and computation overhead could be reduced due to the blockchain system employed. Besides, there are also works focusing on the blockchain-based vehicular network in smart city [10] and inter-vehicle data sharing [11]. However, none of the above literature has conducted a systematic analysis of the defense capability of their proposed architecture against specific attacks so far.

3 System Design

3.1 System Overview

A typical ITS could usually be segregated into three layers. At the bottom are those data collection devices (or basic infrastructure) such as sensors, RSUs, cameras, etc. They constantly collect traffic information and send to the second layer, the local controllers (LC). With the traffic data obtained from nearby devices and possibly other information from adjacent LCs, the LC at a particular intersection then makes the decision for the traffic light timing adjustment. All these data and action taken could be sent to the top ITS center (ITSC) layer for storage, further processing, analysis, and traffic optimization. The entire ITS naturally forms a distributed network, where each entity is capable of communicating to others for information exchange. This property lays the foundation for empowering a blockchain framework on top of the current ITS.

SecBITS is built upon a *permissioned* blockchain system. In such a permissioned system, access control is enforced and one could only act based on the rights granted to her. We define three types of access rights in SecBITS:

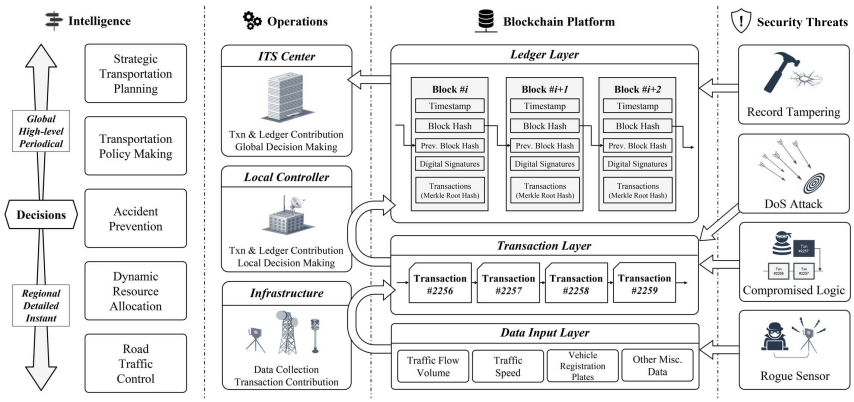

Fig. 1. SecBITS Architecture

Read: allows the system participant to view the historical record of the system
Write: refers to the capability of submitting a "transaction" into the system, of which contains traffic information (either data or action).
Verify: allows the system designated entities to validate the transactions written by other participants, and form them into immutable blocks. This action is similar to the commonly understood "mining" process in the Bitcoin system.

To assign the access rights to the entities in different system layers, we consider their typical roles in an ITS, and their storage and computational capability. The data collection devices at the bottom layer will have the *read* and *write* capability only, as they are usually resource-constrained. Further, as typically the security protection over these devices are limited, granting them *verify* capability may affect the overall system security. The middle layer LCs and the upper layer ITSC will have the full access rights, particularly the ITSC might comprise a set of designated servers within its premise. The ITSC, as the authority, grants the different rights to the entities.

Each entity in SecBITS could be uniquely identified by its public key, generated and issued by the ITSC. Associated with the public key, a private digital signature signing key is also provided and should be stored locally at each entity. We assume the LCs and the servers under the control of ITSC have reasonable storage and computational capacity to perform operations such as digital signature generation, verification, and hash computation. The devices at the bottom layer only have basic functionalities, such as storing its own private key, a list of public keys and perform signature generation when data need to be transmitted.

Fig. 1 provides a system overview for the proposed SecBITS. The overall system is segregated into four different pillars. The "Intelligence" pillar provides a list of exemplary actions in an ITS, sorted based on the characteristics such as time-relevance and geographical impact. "Operations" maps and reflects the three layers of ITS discussed above: the infrastructure for data collection, local

controller for regional traffic decision making and ITS center as the governing entity. The "Blockchain Platform" is segregated into three layers as well, where the raw traffic data are submitted to the system and forms the transaction layer. The system designated entities such as LCs will verify the transactions and form the ledgers. The rightmost pillar, "Security Threats" lists four attacks that may occur at different layers and will be discussed further in a later section.

3.2 Transaction, Smart Contract, Block and Consensus

Transaction One of the fundamental differences between SecBITS and the Bitcoin blockchain system is the definition of *transaction*. Instead of the traditional intuition of reflecting a financial exchange, in SecBITS the term refers to a broader and more generic concept: a transaction is a recorded activity between two parties, of which may contain data exchange or decision executed. We define three types of transactions. All actions could be covered by one of them.
Entity registration: the system requires each participating entity to register itself and record this action. The parties involved in this transaction would be the registering entity, and the ITSC. Information to be recorded could include entity public key, device type, geographic location, access rights, etc.
Data transmission: the data transmission transaction is to record the data exchange between two parties, typically from the lower layer devices to the LCs, and from LCs to the ITSC. This type of transaction will trigger the smart contract to make traffic light decisions.
Decision transmission: when a decision has been made, the corresponding decision-making entity will generate a transaction to send the instruction for execution. Although one could consider the decision transmitted is also a type of data, we separate this from the above for ease of exposition.

Fig. 2 illustrates the format of a transaction. It contains three main fields, the transaction header, payload, and digital signature. The header includes basic information such as the IDs, the timestamp and transaction type, where the payload will include the data or decision to be conveyed between the parties. The digital signature field is included to assure the integrity and authenticity of this particular transaction. The signature is generated by the transaction sender and is based on the transaction header and payload.

Smart Contract In SecBITS, the logic for controlling the traffic condition is embedded into a smart contract, where different smart contracts could be implemented for each intersection based on its location, past traffic data, etc. These contracts are further stored in a duplicated and decentralized way within each capable nodes such as LCs. As each smart contract has its unique identifier, it could be considered as part of the system entity, that is, same as the devices and LCs, but just virtual instead of physical. One would be able to communicate with the smart contract directly by sending a transaction including the smart contract identifier as the receiver ID, and structure the payload accordingly.

Block Transactions generated by the entities will be broadcast into the entire network for verification, before they could be immutably recorded into the ledger, in the form of "blocks". Fig. 3 illustrates the key components included in a block.

Transaction Header	transaction type
	sender ID
	receiver ID
	timestamp
Payload	traffic data, decision, …
Digital Signature	digital signature by sender

Fig. 2. Transaction Format

Block Identifier	Hash of current block
Block Header	Timestamp
	Miner ID
	Previous block's identifier
	Proof of consensus
Payload	List of enclosed transactions
Digital Signature	Digital signature signed by the miner

Fig. 3. Block Format

Each block contains a block identifier, a header, the payload, and the digital signature field. The block identifier is the hash output of the concatenation of the header and payload fields. This identifier will be used for the next block generation in the aim of chaining the blocks together. The header field contains information such as timestamp, miner ID (the identifier of the entity who creates this block), the previous block identifier (as to chain the blocks), and proof of consensus. The payload includes all the transactions details collected by the miner for the past epoch. The digital signature is created based on the concatenation of the first three fields.

Consensus Protocol In SecBITS, we employ Byzantine Fault Tolerance (BFT) protocol. For each time period where a block needs to be formed, according to the protocol, the SecBITS will select one "leader" from the designated entities (e.g., the LCs and servers under ITSC's control). This leader will collect the unconfirmed transactions, form a block as illustrated above and include its ID into the miner ID field. This particular block will be broadcast to the entire network and verified by the community. As long as the number of successful verification passes a threshold, this particular block is considered as valid and written into the ledger. The "proof-of-consensus" to be included are the digital signatures generated by the entities who have successfully verified the blocks.

BFT is a well-studied protocol with many variants. By selecting different variants, the algorithm execution procedure could slightly differ and the consensus threshold level could be adjusted based on the security level required. For the basic BFT scheme where the threshold is set to be 2/3, an attacker still needs to compromise at least 1/3 of the entities to launch a successful attack, which is almost impossible considering the number of entities in the system.

4 Security Analysis

In this section, we discuss four different types of attacks that the SecBITS could assist in prevention and thus enhancing the security of the entire ITS.

Record Tampering The ITS continuously collects, processes and records a large amount of data such as road condition reported from RSUs and local controllers, as well as log information that include decisions made based on the reported data. This information are stored at either LCs or ITSC with different granularity and lifetime, for later analysis, decision making, and investigation. Although different approaches could be enforced, such as enhancing the firewall for protecting the data storage server, or using digital signatures for each data

block to ensure integrity, this recorded information are still subjected to tampering, as firewalls could be breached and signing keys could be compromised.

The proposed SecBITS provides additional assurance for record security due to the data block chaining nature of the blockchain itself. That collected information are validated, "mined" and chained together by the system designated entities to form immutable data blocks. Assume at the time of the attack, there exist n many blocks. An attacker (regardless of insider or outsider) who attempts to manipulate a particular transaction which is contained in block b_k from the current chain would need to re-generate all the block hashing from block b_k to b_n based on the block contents. Further, all the signatures that are contained in the affected blocks need to be re-computed based on the new block hash. This is almost impractical as the attacker needs to breach *all* the entities who participated in creating these blocks and obtain their private signing key in order to form the legitimate signatures.

Denial-of-Service Attack The DoS attack is one of the most commonly observed cyber attacks. The attacker aims at making the system resource unavailable to the legitimate entities either temporarily or indefinitely, typically by flooding the system with a large amount of requests. In SecBITS, as all data and actions are transmitted in the form of a transaction, the attacker could launch a DoS attack by submitting a large number of transactions to the system. To prevent such an attack, we introduce an additional step for generating a valid transaction: one needs to solve a *client puzzle* and embeds the answer into the transaction for validation.

Client puzzle is an approach to increase the cost of a client to carry out certain actions in order to obtain services from the server. Generally, a client puzzle includes puzzle generation, puzzle solving by the client (requires client's effort) and puzzle answer verification by the server (typically easy). In SecBITS, we propose a puzzle that is similar to the "proof-of-work" consensus used in Bitcoin mining process: for the generation of each valid transaction, the submitter needs to guess a random value r such that the hash of r combined with the transaction content tc should satisfy a pre-defined bound (also called *difficulty level*). Further, this bound is a function with respect to the number of submitted transactions n for a fixed period of time t, i.e., the system will adjust the difficulty level in guessing r. As the only way to produce r is by random guessing, with the increased number of transactions submitted, the number of guesses to obtain a suitable r will drastically increase, and for a DoS attack, this will occur significant computational overhead for the attacker. One could define a suitable client puzzle, set the difficulty based on the system requirement and the capability of an attacker. The selection of a suitable puzzle is out of the scope of this paper.

Rogue Sensor Sensors located at the data input layer are one of the most vulnerable points that are subjected to security breaches. An attacker may well be interested in aggregating her resources to compromise a single RSU and sending malicious data to the LC, in order to sabotage the traffic decision made for a specific intersection of interest. SecBITS would help to mitigate such an attack through a set of pre-defined rules embedded in the smart contracts. As mentioned

earlier, each reported data is embedded in a transaction. Before this transaction calls the smart contract that makes the decision for a particular traffic condition, the validity of this data needs to be checked against relevant information such as data reported in last epoch, or compare with adjacent traffic intersections. Only if the data is within a reasonable range bounded by such relevance check, it then could be fed into the smart contract for current decision.

With carefully designed logic and smart contracts, it would be extremely challenging for an attacker to launch such an attack. As the entire system is connected, the relevant information could be extracted from the ledger efficiently for cross-validation. To launch a successful attack, simply sending a manipulated data is not enough, but rather one needs to understand the relationship for decision making and subsequently modify a chain of data to fool the system. However, as discussed in the *Record Tampering* section, modifying recorded information is also extremely difficult.

Compromised Logic The road traffic monitoring and the subsequent decision made is heavily based on the logic implemented in the LC. In the case of the LC is compromised, the attacker may manipulate the embedded *logic* to sabotage the traffic system. Under such an attack, note that the decision is still made based on the *correct* data input, whereas the algorithm itself has been changed. For example, an attacker may change the decision of granting 60 seconds green light based on 100 vehicles reported to only 40 seconds, causing the traffic jam

Traditional ITS might not be able to resolve such an attack as the logic is embedded into each individual LC. With SecBITS, the logic could be implemented as a form of smart contract, each with a unique identifier and stored across the entire network on the nodes. A decision made by an LC needs to be verified by the network by calling the designated smart contract used for such decision before it could be executed. As long as the attacker could not possibly compromise a majority of the entities in the network and change the associated smart contract , such a compromised logic attack would be easily detected and the transaction associated with the decision will be rejected.

5 Experiments

We implemented the RSA and ECDSA digital signature schemes and SHA-2 hash function used in SecBITS, to provide assessment for the computational overhead with a varying security level. The schemes are implemented using Python 2.7 on a Windows machine with Intel Core i7-6700 3.40GHz CPU and 64 GB RAM.

Fig. 4 reflects two types of digital signature schemes with a varying security level. RSA-1024 (represented as DSA-1024 in the figure) has the lowest overhead, takes around 1.6ms for each signature generation. Next, three ECDSA variants and RSA-2048 have relatively close overhead, take approximately 3.3-4.8ms for each signature generation respectively. Consider the devices collect and transmit data typically in hundreds of millisecond or even second granularity, the overhead for these schemes are acceptable. The overhead for RSA-3072 observed a sharp increase, takes roughly 16.5ms for one signing procedure. Although the security

level is high, the computational time seems a burden for the system participants. Fig. 5 plots the signature verification time. The RSA variants could complete the task within extremely short to almost negligible time, approximately less than 1ms for one verification of all variants. For elliptic curve schemes we observed a 2x−3x blow-up of the computational overhead compare to its signing operation.

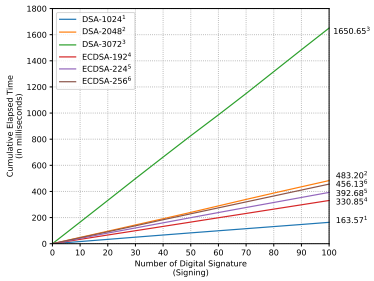

Fig. 4. Signature Signing Time

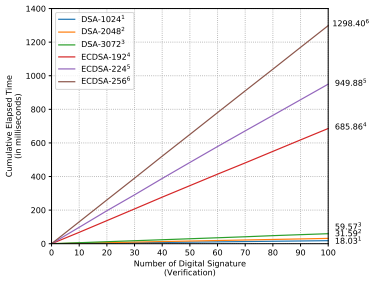

Fig. 5. Signature Verification Time

We also test the block formation time, i.e., to generate the block identifier by hashing the formed block including header and payload with collected transactions over a period of time. Typically, SHA-256 is sufficient for current security needs, and roughly 680ms is required if a block contains 100 transactions (Fig. 6).

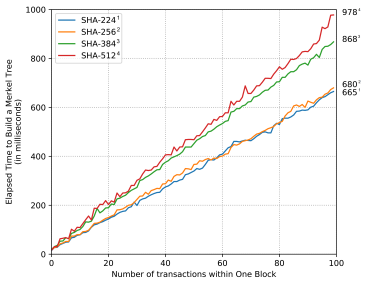

Fig. 6. Block Generation Time

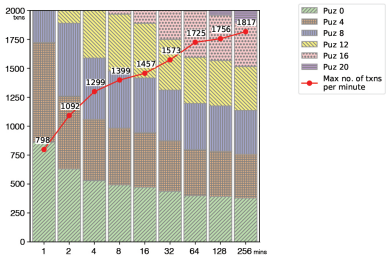

Fig. 7. Client Puzzle Solving

Last, we test our client puzzle idea on a Raspberry Pi (Model 2B). The "X" in Puz X of the legend refers to the number of leading zero bits required in the hash output. We first obtain the number of transactions the device could generate within one minute with no difficulty level as a benchmark, and on average of 800 transactions could be formed (for the compactness of the figure, this value is not included). Subsequently, we set the puzzle level increment threshold to be 320 (i.e., on a 0.4x threshold, 0.4×800), that is, for every 320 transactions submitted within the past time epoch, the difficulty level will be increased. The x-axis indicates the varying time epoch, and the y-axis indicates the maximum number of transactions the device could generate within the epoch (red dotted line). One could observe that, even with a geometric growth of the time epoch (i.e., giving an attacker with more time to flood the system), the number of

transactions generated only grows (seemingly) linearly, which greatly limits the severity of the attack. The colored bar indicates to which level of difficulty the device could reach within each time epoch (e.g., within four minutes, the device could reach a level of generating 12 leading zero bits for the hash output).

6 Conclusion

In this work, we propose SecBITS, a secured blockchain-based intelligent transportation system. We provide detailed system architecture, define transactions and ledger, as well as consensus to ensure a secured global view for the data and information recorded. Further security analysis of four types of attacks and how the proposed system could mitigate such attacks are discussed. Experiments are also conducted to illustrate the feasibility of the proposed solution.

References

1. Junping Zhang, Fei-Yue Wang, Kunfeng Wang, Wei-Hua Lin, Xin Xu, and Cheng Chen. Data-driven intelligent transportation systems: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2011.
2. V Kastrinaki, Michalis Zervakis, and Kostas Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and vision computing*, 21, 2003.
3. Samuel Gidel, Paul Checchin, Christophe Blanc, Thierry Chateau, and Laurent Trassoudaine. Pedestrian detection and tracking in an urban environment using a multilayer laser scanner. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):579–588, 2010.
4. Toru Seo and Takahiko Kusakabe. Probe vehicle-based traffic state estimation method with spacing information and conservation law. *Transportation Research Part C: Emerging Technologies*, 59:391–403, 2015.
5. Multi-modal intelligent traffic safety system (mmitss). `https://www.its.dot.gov/research_archives/dma/bundle/mmitss_plan.htm`.
6. Aymen Boudguiga, Nabil Bouzerna, Louis Granboulan, Alexis Olivereau, Flavien Quesnel, Anthony Roger, and Renaud Sirdey. Towards better availability and accountability for IoT updates by means of a blockchain. In *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2017.
7. Yong Yuan and Fei-Yue Wang. Towards blockchain-based intelligent transportation systems. In *IEEE 19th International Conference on Intelligent Transportation Systems*, pages 2663–2668. IEEE, 2016.
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*, pages 137–140. ACM, 2016.
9. Ao Lei, Haitham Cruickshank, Yue Cao, Philip Asuquo, Chibueze P Anyigor Ogah, and Zhili Sun. Blockchain-based dynamic key management for heterogeneous intelligent transportation systems. *IEEE IoT Journal*, 4(6):1832–1843, 2017.
10. Pradip Kumar Sharma, Seo Yeon Moon, and Jong Hyuk Park. Block-VN: A distributed blockchain based vehicular network architecture in smart city. *Journal of Information Processing Systems*, 13(1):84, 2017.
11. Madhusudan Singh and Shiho Kim. Blockchain based intelligent vehicle data sharing framework. *arXiv preprint arXiv:1708.09721*, 2017.