Securing Vehicular Ad Hoc Networks (VANET) using Blockchain

Aditya Narayanan B, Amirtha Varshni T, Serin Banu R, Shrinitha S, and Srivatsan K P

Madras Institute of Technology, Department of Information Technology

***Abstract*- In the era of the digital world everything including vehicles is connected and automated. Connected and Automated Vehicles (CAV) are an advent in transportation technology providing a huge array of functionalities and notable benefits that include a decline in congestion and road fatalities. However, it is not free from attacks of malicious entities as the level of automation and connectedness increases. There is a high likelihood of them being attacked and entirely controlled by external entities risking their safety and security. The problem is found to most likely arise from the centralized design of the in-vehicle network and issues with the availability and reliability of data in case of a successful attack of the ECUs. Hence, we propose a Blockchain Framework that has the potential to provide restricted access to information in the connected vehicle system. We use a Challenge-response mechanism between the vehicles and Road side units to verify the internal state of the ECU. Authentic communication is ensured by allowing only the CAVs with a verifiable record in blockchain to exchange messages. We also provide a mechanism to verify the identity of the sender of a message to an RSU and a Machine Learning model to detect DDoS attacks. This framework doubles the detection and reaction mechanism offering adequate security to vehicles and the vehicular network.**

***Keywords* – CAVs, ECUs, Blockchain, Merkle root, Vehicular network, RSU, Digital Signature, Random Forest, Logistic Regression, KNN, Decision Tree and XGBoost**

I. INTRODUCTION

Connected and automated vehicle (CAV) may be a transformative technology that has great potential to improve our standard of living. Connected vehicle technologies allow vehicles to exchange message to each other and to the infrastructure around them. They are embedded with electronic devices called electronic control units (ECUs) which provide advanced vehicle functionality and facilitate independent decision making. ECUs receive input from various sensors present in the CAV and runs computations for their required tasks. However, these vehicles could be exploited remotely when they are proliferated. External entities could inject malware in ECUs which would compromise the internal network of the vehicle affecting the communication between multiple ECU in the vehicle and the communication with the road side units. These entities can gain full control of the CAV. They are a threat to the security of smart vehicle and Road side units. We focus on monitoring the internal

state of the ECUs to detect an ECU compromise.

Earlier works were based on intrusion detection and integrity checking which detected unauthorized access to in-vehicle network. But these models were based on a centralized design which relies on a Central ECU. The entire in-vehicular network would be put risk when this central ECU is compromised leaving the CAV vulnerable to security attacks.

The problem arises due to centralized design of in-vehicle network and issues with data availability and reliability in case of successful attack of the ECUs. Hence, we propose a Blockchain Technology that has the potential to address the aforementioned challenges including centralization, availability and data reliability.

Blockchain Technology provides a decentralized security framework for in-vehicle networks. It introduces an initialization operation for authentication, a challenge-response mechanism to ensure the integrity of the vehicle's in-network and a mechanism based on digital signature to verify the identity of the CAV sending a message. A qualitative evaluation of this framework is conducted to evaluate its resilience to identified attacks.

II. LITERATURE SURVEY

CAV related research has been advanced significantly in the recent years. For intrusion detection and integrity checking many authentication approaches using Machine Learning and Blockchain are proposed.

A Code-Based authentication scheme was used by authors in [1] for Lightweight Integrity checking of smart vehicles. Code Splitting and bivariate polynomial-based secret sharing scheme was used in their model. The proposed model was found to be computationally efficient and had no additional communication requirement. But it provided a weak authentication to mitigate denial of services attacks against public key protocols.

A Machine Learning model was proposed by authors in [2] for securing smart vehicles from relay attacks. CART- a powerful machine learning for both classification and regression was used to detect abnormal behavior of key fob and LSTM recurrent neural network was used to identify the identity of the current driver. But the main challenge was to adapt the proposed solution for multiple drivers, as the proposed solution only uses a three-month dataset of the key fob rather than the driver-specific data.

A Traceable Blockchain-based Access Authentication System with Privacy Preservation in VANETs was proposed by authors in [3] for a secure communication among smart vehicles. This system employs Blockchain technology, Internet of Things and Vehicular Adhoc Networks (VANET). The main concern was found to be increasing network congestion.

The security and Privacy of smart vehicles that uses a Wireless linking and Tamper-proof GPS was proposed by authors in [4] to secure smart vehicles and to improve road safety and traffic management. Electronic License Plate and a Location verification system using GPS pave are used for cooperative driving. Though the proposed work had many advantages like Dynamic pricing, Identification of culprits, Cooperative driving, Electronic License

plates used for identification were found to be vulnerable and might lead to Impersonation attack, Blocking and jamming of signals (Denial of service).

A Blockchain Framework for Securing Connected and Autonomous Vehicles was proposed by authors in [5] to provide security and transparency in a CAV network of IoT devices through a blockchain technique. In this technique, data captured through IoT devices are stored in Blockchain. But the system was found to be vulnerable to data falsification attack, traffic jam, compromising IoT sensors.

III.PROPOSED WORK

A. Architecture Overview

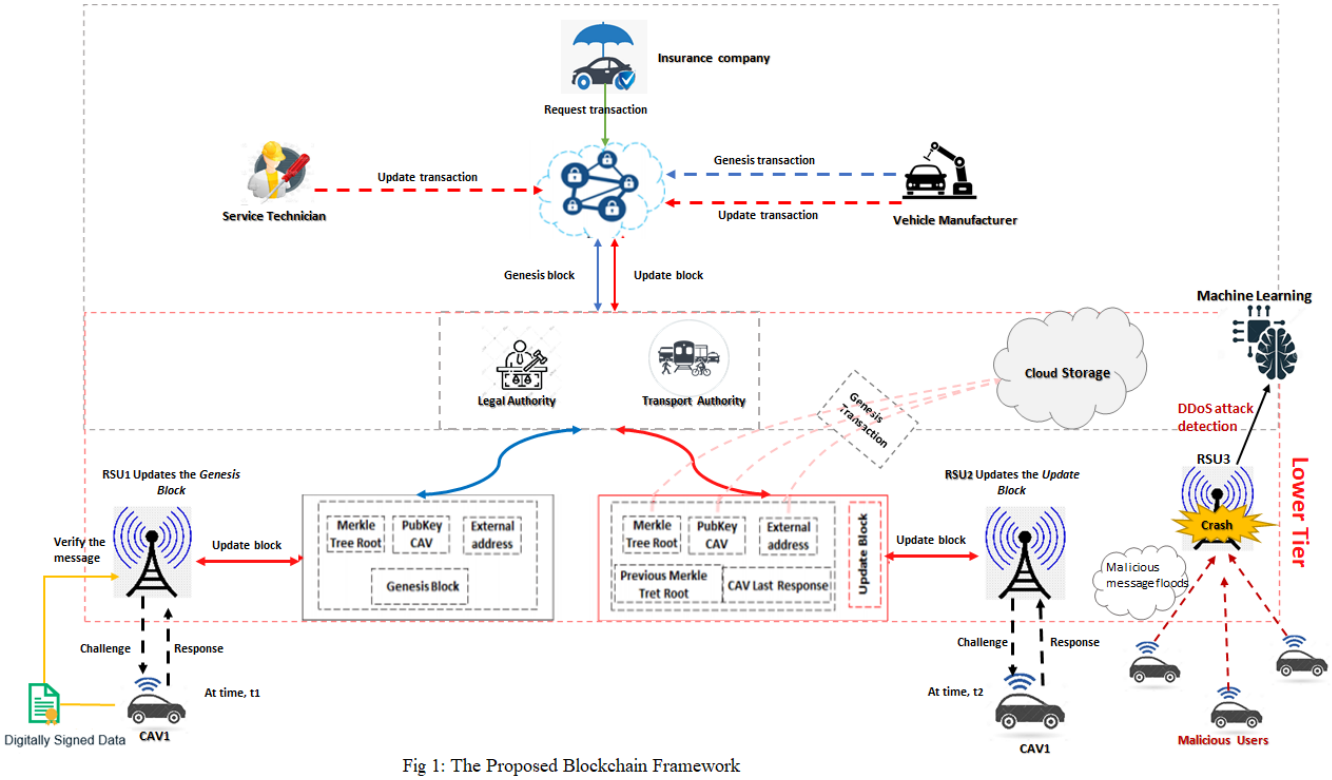

Fig 1: The Proposed Blockchain Framework

The architecture proposed focuses on monitoring the ECU state and keeps track of the changes to ECU states. So, the Blockchain is divided into two tiers namely upper and the lower tiers. These tiers make the role of the interacting entities clear and ensure that these entities are given only the information they need to know. The upper

tier comprises of vehicle manufacturers, service technicians, insurance companies, legal and road transport authorities. The interaction in this tier mainly focuses on vehicle registration and maintenance. The registration of a smart vehicle with the legal and transport authorities is carried out by the vehicle manufacturer and used to create a block for the vehicle in the upper tier. This block stores the current state of the vehicle and hashes of the ECUs. These hash values are then used to perform vehicle validation in the lower tier whenever it comes to the proximity of an RSU. However, when a maintenance occurs, the service technician or the manufacturer updates current state of the vehicle to the authorities for the future assessment of the integrity of the CAVs.

The lower tier comprises of roadside units (RSUs), smart vehicles, legal and road transport authorities. Here, whenever a vehicle connects to an RSU, the RSU sends a challenge request to prove its integrity of the ECUs. The response is provided by computing the cumulative hash values of all the ECUs. When a vehicle moves from one RSU to another, the timestamps are compared and verified to identify replay attacks. We only store two transactions for a vehicle and push the rest of the transactions stored in a RSU to cloud storage. The proposed framework makes use of digital signature to verify the authenticity of the sender of a message to an RSU.

B. Blockchain based Framework for Securing VANETs

This section elaborates on the architecture and the working of the proposed framework. The entities involved in this framework are CAVs, Vehicle manufactures, RSUs, insurance companies, service technicians, transport and legal authorities which is described in Figure 1. Based on the roles of an entity it can be classified as a verifier or a proposer. The entities that fall under the category of verifiers are responsible for verifying the data sent to the Blockchain while those under the category of proposers are responsible for sending data to the blockchain or providing response to a challenge request. RSUs, transport and legal authorities are termed as verifiers and vehicle manufacturers; CAVs, insurance companies and service technicians are termed as proposers.

Actions performed by CAV in this framework can be considered as transactions and upon being verified they are appended to the blockchain. These transactions can be classified as registration of CAV, challenge response mechanism, updation of a CAV block and authenticating a message.

i. **Registration of CAV**

A CAV when assembled by the vehicle manufacturer, it is given a SS_{ID} which in turn is computed by the Merkle root value of all ECU hashes in the CAV. This value is then forwarded to the Transport and legal authorities. These authorities are responsible for validating the value and creating a

block for the CAV in the blockchain. This block becomes a public block for the RSUs to access them when verifying the integrity of the CAV in the lower tier. Registration of a CAV creates a public block for the CAV and this transaction occurs in the upper tier.

Genesis Transaction

The transaction initiated by the vehicle manufacturer when the CAV is assembled is termed as genesis transaction. This transaction upon being verified by transport and legal authorities creates the genesis block for the CAV.

A genesis block consists of:

- A public key/id
- Firmware hashes of all ECUs (Electronic Control Units)
- SS_{ID} – Merkle tree root from all firmware hashes
- Timestamp of block

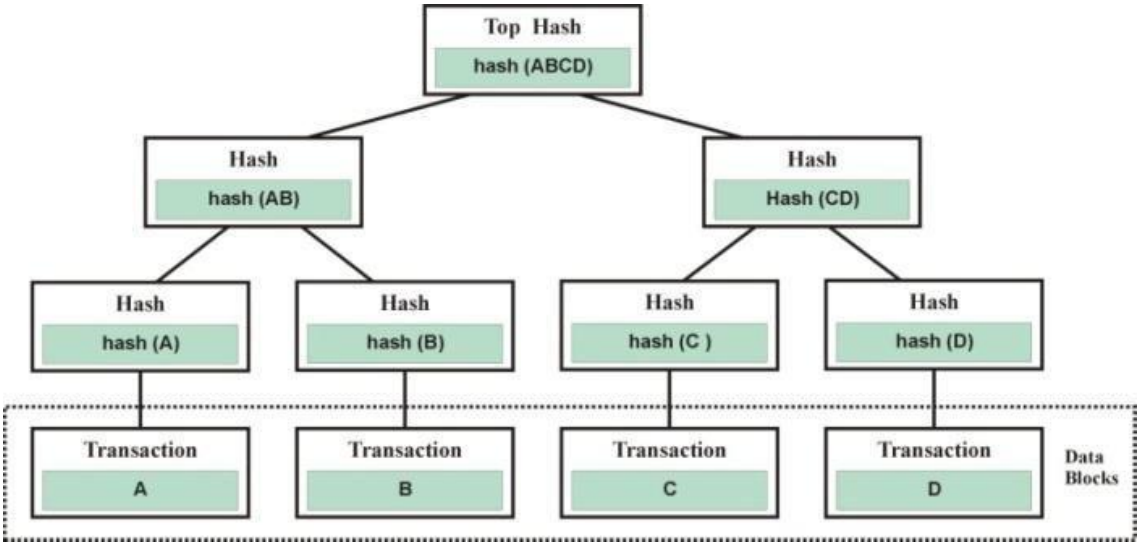

Fig. 2: Generating the Merkle tree root value

This block becomes a permanent block in the blockchain after the authenticity of the transaction is verified by the transport and legal authorities. This block is used to validate the CAV in the lower tier until any further update.

ii. Challenge Response Mechanism

When a CAV approaches the coverage area of RSU, the RSU initiates the challenge response mechanism and upon validation the RSU updates the CAV block. When CAV approaches an RSU, the RSU challenges the CAV to prove the integrity of its ECUs and the Cav provides a response. For this response, CAV computes the SS_{ID} (Merkle tree root value) and hash values of the selected ECUs. The response including its signature, time stamp and public key is sent to the RSU for the verification. From the response received from the CAV the RSU first checks the validity of the CAV. This requires the RSU to check for the

presence of the corresponding CAV block in Blockchain. If no such block is found, then it notifies the road transport and legal authority about the malicious CAV. If the validation succeeds, then RSU compares the SS_{ID} of response received from CAV with the SSID in the block. If SS_{ID} are verified then it checks the hash value of a random ECU. When the verification is successful, updation occurs in the lower tier. Any failed verification will lead the RSU to inform the road transport and legal authority about the malicious CAV.

When the same CAV encounters another RSU, challenge and response activity begins all over again. It follows the same steps as above but in-addition there will be another level of verification. The additional level verification is the timestamp, the RSU compares the time stamp on the response data to the immediate previous record stored on the lower tier blockchain. The time stamp value is expected to continuously increase as the vehicle travels, if this is the case, RSU executes updates on the CAV's block and if this fails, the RSU can detect a malicious action and also classifies the type of attack. RSU then informs road transport and legal authority about the CAV and the type of attack.

iii.Updation of a CAV block

The updation transaction is usually initiated by the service technician or by the vehicle manufacturer. When an ECU (Electronic Control Unit) in the CAV is updated, the update transaction is initiated and this is done during the process of scheduled maintenance or diagnostics. An updation leads to the change in the initial SS*ID* value and this value contains the updated SS*ID* value, timestamp, public key of the CAV in the process, and the public key of the manufacturer or the technician. This updation process can occur either in the upper-tier or lower tier.

The upper tier consists of the Vehicle Manufacturer or the Service Technician or the Vehicle Insurance Company. When an update is initiated in the upper tier, the transaction records the CAV in the lower tier so that it can be used to validate the authenticity of the vehicle in the lower tier. The lower tier consists of RSUs (Road Side Units). The transaction is initiated when the CAV provides a response to the challenge request, which is usually done by the RSU. Then the updated transaction contains the *CAV's response* which consists of CAV's SS_{ID} and timestamp. The response also contains the hash values of ECUs and the public key of RSU.

iv. Authenticating a message using Digital signature

V2I (Vehicle to Infrastructure) Communication allows the wireless exchange of data or message between vehicles (CAVs) and the road infrastructures like Road Side Units (RSUs). This enables a CAV to send message regarding traffic and accidents that occur on the road. But there has to be a mechanism to ensure the integrity of the message and the authenticity of the sender. So, a digital signature is used to authenticate the identity of the CAV sending the message.

Digital signatures employ asymmetric cryptography. Asymmetric cryptography also termed as Public Key Cryptography (PKI), uses public and private keys to encrypt and decrypt data. Public key can be shared with everyone while the private key is kept secret.

Signing a message with private key: The message is hashed using a one-way hash function. Then the hashed message is encrypted using the private key. This way a message is signed with private key.

Verifying the message with public key: This process involves two steps namely generating hash of the message and decrypting the signature. If the decrypted hash matches the second computed hash of the same message, then it proves that the message hasn't been changed but when failed to match, the message has either been tampered with in some way (integrity) or the signature was created with a private key that doesn't correspond to the public key provided by the signer.

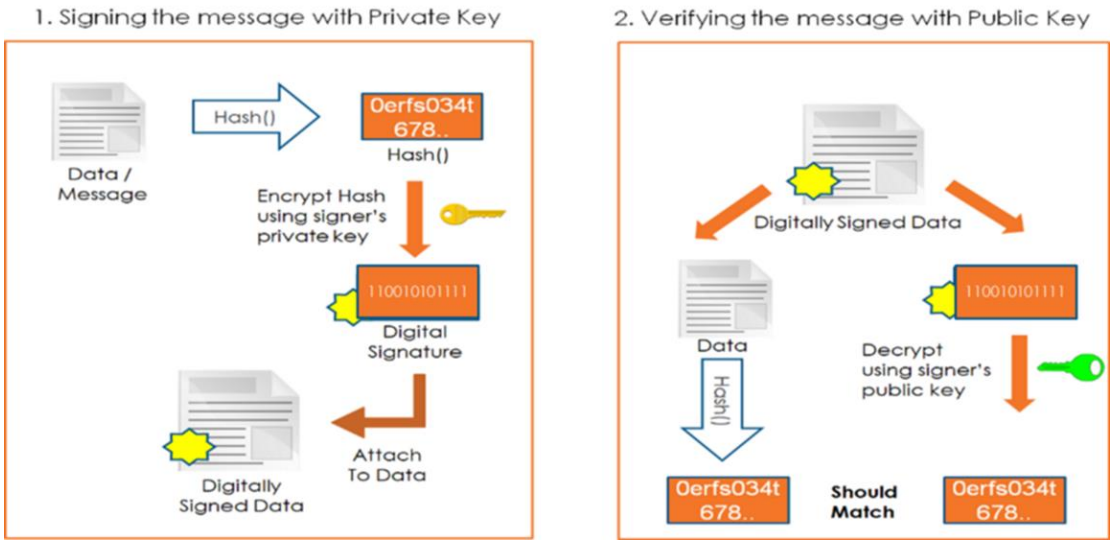

Fig. 3: Signing and Verifying a message using Digital Signature

The proposed framework uses Eth-sign mechanism, a ECDSA (Elliptic Curve Digital Signature Algorithm) cryptographic hashing technique to generate the signature and ecrecover function that returns the address of the signer to verify the authenticity of the message.

ECDSA signatures composes of two numbers r and s. It also requires an additional variable v (recovery identifier). The signature can be denotated as {r, s, v}. Creating a digital signature requires the sender to sign the message with the private key, d_a. The following are the simplified steps for signing process:

1. Generating hash of the message (e).
2. Generating a random number (k).
3. Calculating a point (x_1, y_1) on the elliptic curve which is a result of multiplying k with the G constant of the elliptic curve.
4. Calculating $r=x_1 \bmod n$. If r equals zero then go to step 2.
5. Calculating $s=k^{-1}(e + rd_a) \bmod n$. If s equals zero then go to step 2.

The function ecrecover takes hash of the message, v, r and s to return the address of the signer.

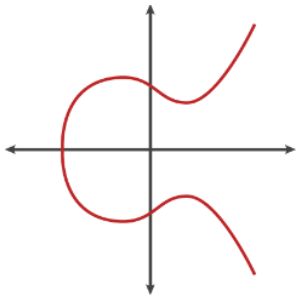

Fig. 4: An example of Elliptic Curve

Security Analysis

This section is to discuss on the resilience of the proposed model against various possible attacks.

Fake data: This occurs when one or more ECU is compromised. This attack can be detected by comparing the SS_{ID} values during the integrity check in challenge-response mechanism.

Code injection: This occurs when an external entity injects or alters the code of the ECUs. This can be prevented as the state of the ECUs and actions performed on them

is continuously monitored using blockchain.

Sybil attack: This occurs when a vehicle tries to create multiple entities. Only the verifiers in the upper tier are capable of creating a valid block for an entity and hence preventing the multiple entities falsely created to create valid blocks for themselves.

Masquerade attack: This occurs when a compromised RSU tries to create a block for CAV. This attack can be avoided as only the CAVs with their public key in the blockchain managed by the transport and legal authorities, are considered to be valid.

ECU Reply Attack: This attack occurs when valid data block is intercepted and is maliciously repeated or delayed. This attack can be detected by comparing the timestamp during challenge- response mechanism.

ECU State Reversal Attack: This attack occurs when the state of an ECU has been altered. This can be detected by checking the integrity of a random ECU during challenge- response mechanism.

C. Machine Learning Model for DDoS attack detection

A Denial-of-Service (DoS) attack is an attack that is meant to shut down or corrupt a machine or network and make it inaccessible to its intended users or clients. The important type of DoS attack is Distributed Denial-of-Service (DDoS) attacks. A Distributed Denial-of-Service attack mainly involves multiple connected online devices that are collectively known as a botnet, that are used to overwhelm a target website or user with fake traffic. It can be used as a smokescreen to do some other malicious activities and compromise the security of devices. We could train the hosting servers with machine learning algorithms to be able to detect such scenarios and not affect the service it provides to its legitimate users.

The main objective is to detect DDoS attacks in smart vehicles using the classification algorithms in machine learning to build a decentralized security framework for in-vehicle networks.

1) Data Preprocessing

Data Preprocessing is a process of collecting raw data from different sources to process it and making the processed data suitable for training a model using machine learning algorithms. It is the first and foremost method to create machine learning models.

In data pre-processing, the raw data generally contains missing values, values in the wrong format and values out of range, etc. So, the data has to be cleaned and processed before using it, so that the data can produce more accurate results.

Dataset Used

In this DDoS detection, we use the Canadian Institute of Cybersecurity IDS (CIC-IDS 2017) dataset. With the help of the generated dataset, we are proposing a new classification approach based on a set of network flow features in the dataset. In this dataset, we can detect different types of DDoS attacks along with their corresponding weights in the dataset.

It contains benign and one of the most up-to-date common DDoS attacks, which also resembles the true real-world data (PCAPs) as it also includes the results of the network traffic analysis using CICFlowMeter-V3 with labelled flows based on their time-stamps, source IP, and destination IPs, source and destination ports, protocols and attack in CSV files. The dataset has been provided and updated for each day to record the raw data including the network traffic and event logs per machine.

Viewing the imported dataset

The first and foremost thing is to take a look at the dataset you are going to work with. Pandas provide the functions to take a look at some portion of the dataset using the head or tail function to view the dataset either from the first or last rows of the dataset.

Replacing Zeros with NaN values

To detect whether the vehicle has been affected or not, we take a particular set of columns as an independent variable to determine the result which is known as the target variable. So, to ensure that all the rows of the independent variable are filled, we replace the rows with zero to NaN value rows to determine the accurate mean value.

Replace NaN values with their mean values.

To find the mean values of the respective columns, we need to convert the data type of the columns to float, so that we can easily find the mean value.

Then, we can replace the NaN values with the calculated mean values in the respective columns.

Remove Infinity Float Values

It replaces infinite values with NaN and then drops rows with NaN. First, we replace the values using df.replace() function and then drop the rows with NaN using dropna() function.

Data Visualization

Data visualization provides qualitative understanding and can be helpful when we try to explore the dataset and extract some information about a dataset so that we can help with identifying patterns, corrupt data, outliers of the data, and much more.
With the help of Seaborn and Matplotlib libraries, we can plot the graph using the Label column using the countplot function to count the number of attacked and non-attacked vehicles.

Feature Scaling

Feature Scaling is an important technique to standardize the independent features present in the data in a given fixed range. Feature Scaling is performed during the data pre-processing to handle highly varying values or units. If feature scaling is not done, then machine learning algorithms tend to produce inaccurate values.

We have imported the MinMaxScaler from the Sklearn library to fit the values in the dataset in the range of 0 to 1.

And then we have split the independent and dependent variables into X and Y variables. Then, import train_test_split from sklearn model selection to split the proportion of the dataset for training the model. And then input the remaining portion of independent variable values into the model to predict the target value and compare with the actual values to predict the accuracy of each algorithm to find the best one by comparing each other.

2) Algorithm Evaluation

After data pre-processing is completed, we begin the process of evaluating various ML algorithms against the dataset. We split the dataset into a training set and test set with the ratio of 70% and 30% respectively. After splitting the dataset, we apply five most used classification algorithms and identify the most effective algorithm in DDoS attack detection. The five algorithms chosen are Logistic Regression, K-Nearest Neighbor, Random Forest, Decision Tree and XGBoost.

Logistic Regression is used to predict the categorically dependent variable for a set of independent variables. This algorithm predicts the probability of the categorically dependent variable. This algorithm is used when we need probabilistic results. In logistic Regression we plot logistic function and by determining a threshold value, it becomes a classification algorithm. Just like liner regression, this also uses a sigmoid function to map predicted values to probabilities.

The Logistic Regression is implemented from sklearn class by setting all the necessary parameters. We choose the 'lbfgs' solver as it is used for solving multiclass problems. Set the other parameters to its default values.

Another most commonly used classification algorithm is K Nearest Neighbor. It uses previously known datapoints to predict the position/value of a new datapoint. KNN algorithm's learning is instance-based learning and it does lazy learning. The 'euclidean' distance metric can be chosen keeping the number of kneighbors to query as 5. Set the algorithm as 'auto' and set all the points in the neighborhood as weighted equally. This trained dataset can now predict the values for newly encountered datapoints.

The decision tree algorithm builds a training model with which we predict the values of a target variable by learning simple decision rules inferred from previous training data. We begin the process from the root node of the tree and compare the values of the root attribute with the record's attribute to follow the correct branch and move to the next node. Set the criterion to 'entropy' and let the splitter be 'best'. Let the rest of the attributes take default values.

The next algorithm picked is random forest, uses ensemble learning which means it combines the result of many classifiers to predict the values accurately. To implement ensemble learning, the random forest consists of multiple decision trees. The outcome is calculated by taking the cumulative mean of all the outputs of the decision trees. We select the criterion to be 'entropy' and min_samples_split as '2'. Let the other parameters take default values.

XGBoost is also an ensemble learning algorithm fundamentally based on developing decision tree. It uses bagging

and boosting as ensemble learners to increase the accuracy of the prediction. Boosting helps in reducing the errors from the previous trees hence each tree learns from its predecessor. To avoid overfitting, boosting makes use of trees with fewer splits. Choose the 'gbtree' as booster and all its corresponding parameters as default.

After the training is completed, predict the values of the testing set using these five algorithms. By obtaining and comparing the recall score we pick the most efficient and appropriate algorithm for our dataset.

	Recall score	**Precision score**
Logistic Regression	0.9992449489689648	0.9339092347000851
KNN	0.9985680066652781	0.9997653928366613
Decision Tree	0.9985159341803791	0.9998435748364054
Random Forest	0.9985419704228286	0.9998435789144376
XGBoost	0.9991408039991668	0.9977899115964639

Since Random Forest algorithms performs and predicts better than other algorithms, we use it to train the RSUs in our lower-tier to detect any DDoS attack successfully and thus prevent the denial of CAV's requests during the attack.

lower tier verifies data from the upper tier. They both work together to establish a connected and secured vehicular system. Any further improvement could be in the direction of privacy and in integrating the Blockchain Framework and Machine Learning Model.

IV.CONCLUSION

In this paper, we have presented with a model which exploits the blockchain framework to provide security for smart vehicles. This paper primarily explains on identifying malicious CAVs by detecting any comprise in the internal state of the CAV. The upper tier is granted permission to manipulate the blockchain whereas the

REFERENCES

[1] Joosang Yoo and Jeong Hyun Yi, *Code-Based Authentication Scheme for Lightweight Integrity Checking of Smart Vehicles.* Also available on: https://ieeexplore.ieee.org/document/8444372

[2] Usman Ahmad, Hong Song, Awais Bilal, Mamoun Alazab, Alireza Jolfaei, *Securing Smart Vehicles from relay attacks using Machine Learning* Also available on: https://www.researchgate.net/publication/336927855_Securing_smart_vehicles_from_relay_attacks_using_machine_learning

[3] Dong Zheng, Rui Guo, Shiyao Gao, *A Traceable Blockchain-based Access Authentication System with Privacy Preservation in VANETs.* Also available on: https://ieeexplore.ieee.org/document/8808923

[4] J.P. Hubaux, S. Capkun, Jun Luo, *The Security and Privacy of Smart Vehicles*. Also available on: https://ieeexplore.ieee.org/document/1306972

[5] Geetanjali Rathee, Ashutosh Sharma, Razi Iqbal, Moayad Aloqaily, Naveen Jaglan and Rajiv Kumar, *A Blockchain Framework for Securing Connected and Autonomous Vehicles*. Also available on: https://www.mdpi.com/1424-8220/19/14/3165