Deep Learning and Smart Contract-Assisted Secure Data Sharing for IoT-Based Intelligent Agriculture

Kumar Randhir, Kumar Prabhat, Aljuhani Ahamed, Islam A. K. M. Najmul, Jolfaei Alireza, Garg Sahil

This is a Author's accepted manuscript (AAM) version of a publication published by IEEE in IEEE Intelligent Systems

DOI: 10.1109/MIS.2022.3201553

Copyright of the original publication: © IEEE

Please cite the publication as follows:

This is a parallel published version of an original publication. This version can differ from the original published article.
Abstract—The recent development of Internet of Things (IoT) and Unmanned Aerial Vehicles (UAVs) has revolutionized traditional agriculture with intelligence and automation. In a typical Intelligent Agriculture (IA) ecosystem, massive and real-time data is generated, analyzed, and sent to the Cloud Server (CS) for the purpose of addressing complex agricultural issues like yield prediction, water feed calculation, and so on. This helps farmers and associated stakeholders to take correct decision that improves the yield and quality of agricultural product. However, the distributed nature of IA entities and the usage of insecure wireless communication open various challenges related to data sharing, monitoring, storage and further makes the entire IA ecosystem vulnerable to various potential attacks. In this paper, we exploit deep learning and smart contract to propose a new IoT-enabled IA framework for enabling secure data sharing among its various entities. Specifically, first we develop new authentication and key management scheme to ensure secure data transmission in IoT-enabled IA. The encrypted transactions are then used by the CS to analyze and further detect intrusions by a novel deep learning architecture. In CS, the smart contract-based consensus mechanism is executed on legitimate transactions that verifies and adds the formed blocks into blockchain by a peer-to-peer (P2P) CSs network. In comparison to existing competing security solutions, a rigorous comparative research demonstrates that the proposed approach provides greater security and more utility characteristics.

Index Terms—Blockchain, Deep Learning, Internet of Things (IoT), Smart Contracts, Intelligent Agriculture (IA).

I. INTRODUCTION

According to a UN study, the world’s population will reach 9.8 billion people by 2050. This rise in population demands nearly 70% increase in current food production rate. Agriculture is the world’s most important industry, contributing significantly to social stability and economic progress [1]. The transition from traditional agriculture (also known as Agriculture 1.0) to Intelligent Agriculture (IA) (also known as Agriculture 4.0) is the only alternative to meet the growing demand efficiently [2]. IA is a new approach that use the current information and communication technologies in conjunction with conventional farming practices to improve the quality and quantity of agricultural products [3]. The IA helps in intelligent decision making and provides various personalized services through different key technologies including, Internet of Things (IoT), Unmanned Aerial Vehicles (UAVs), cloud computing and Artificial Intelligence (AI) [4].

In a typical IA ecosystem, several data acquisition technologies such as IoT devices and actuators are deployed to collect both field and crop growth information. In addition, UAVs are used to gather data from IoT devices and, in certain cases, they may collect data directly from particular flying zones. The data acquired is forwarded to Cloud Servers (CSs) for the purpose of addressing complex agricultural issues such as yield prediction, water feed calculation, and so on, assisting farmers and other stakeholders in making smart decisions that increase agricultural production and quality [5]. However, the distributed nature of IA entities (including IoT devices, UAVs and CSs), and the usage of insecure wireless communication open various challenges related to data sharing, monitoring, storage and further makes the entire IA ecosystem vulnerable to various potential attacks including impersonation, replay, man-in-the-middle, data poisoning, brute-force, physical smart devices and UAVs capture attacks [6].

In the literature, several key management mechanisms, blockchain- and smart contract-based authentication strategies for enhancing security of IoT-enabled IA have been put forth. For instance, works presented in articles [7], [8], [9], [10] were mainly based on user authentication/authorization and session key management. However, all above solution used blockchain as a distributed storage mechanism to store entire agricultural transactions. Unfortunately, blockchain becomes inefficient when complete transaction are offloaded to the distributed ledger but works better with data hashes [11]. Furthermore, we believe that all of the above authorization and authentication techniques are insufficient for addressing security issues in IoT-enabled IA networks since they only ensure that data transmission is secure but do not guarantee or check the type of data (attack or normal) before it is added to blockchain [12], [13].

Motivated from the aforementioned challenges, we exploit deep learning and smart contract to propose a new IoT-enabled IA framework for enabling secure data sharing among its various entities. Specifically, first we develop new authentication and key management scheme to ensure secure data
transmission in IoT-enabled IA. The encrypted transactions are then used by the CS to analyze and further detect intrusions by a novel deep learning architecture. The latter is a novel architecture that is designed using a Contractive Sparse AutoEncoder (CSEA), Gated Recurrent Unit networks (GRU), Multi-Layer Perceptrons (MLPs) and softmax classifier for attack detection. In CS, the smart contract-based Proof of Authority (PoAAura) consensus mechanism is executed on legitimate transactions that verifies and adds the formed blocks into InterPlanetary File System (IPFS) by a peer-to-peer (P2P) CS network. The returned cryptographic hash if further stored on blockchain.

II. SYSTEM MODELS

We introduce a network model in this part, followed by a threat model, both of which were used in the design of the proposed framework.

A. Network Model

The network model of the proposed framework is illustrated in Fig 1. In this model, we have mainly eight entities, Trusted Authority (TA), IoT Device (IoT), Unmanned Aerial Vehicles (UAV), Intrusion Detection System (IDS), Cloud Server (CS), InterPlanetary File System (IPFS), Smart Contract (SC) and Blockchain Network (BN). In IoT-enabled IA, the TA is responsible to register IoT, UAV and CS prior to their deployment. Initially, the authentication and key management phase includes mutual authentication and key agreement between two IoT, between IoT and its associated UAV, and between UAV and CS using the established session keys. This phase ensures secure communication among the participating entities. Once the communications starts, the IoT placed in each flying zone (FZ) has the capability to extract crop readings from its zone. Each FZ is associated with a UAV that collects the readings from IoT. This data or transaction includes the status of standing crops, quantity of chemicals used as pesticides at various locations and so on. These transactions are forwarded to CS, where the proposed IDS checks and marks the transaction as normal and abnormal based on the behavior. The valid transaction are then used by each CS for mining using smart contract. Specifically, each CS mines and stores the valid transactions into IPFS, keeping the returned transaction hash into the global BN.

The IPFS hashes of the verified transactions are packed into the current block by each miner, who also creates the merkle root and block hash while calculating the subsequent block. If CS calculates a block hash that satisfies the difficulty, it will be broadcast to miners CS, CS and CS, and so on. After receiving the block, miners CS, CS and CS must check the transactions and block hash. The majority of transactions received by miners CS, CS and CS throughout the mining process are similar to those received by miner CS. They only alter a small number of transactions in their transaction pool because of network transmission delays. As a consequence, the vast majority of IPFS hashes for transactions in the new block match those in miner CS, CS and CS’s personal transaction pool. If the local transaction pools of miners CS, CS and CS include the IPFS hash of a transaction in a block delivered by miner CS, then the transaction has already been confirmed by these miners and does not require downloading from IPFS. The IPFS network must be accessed using the proper IPFS hashes in order to receive the data for the remaining transactions. The authenticity of the block and the transactions would then be verified. The BN could then be updated with the new block.

B. Threat Model

The “Dolev-Yao” threat model, often known as the DY model, is the first one we employ in this paper [14]. This theory states that an adversary designated as A has the ability to not only intercept, alter, or delete communication messages between any two participants, but also to introduce harmful messages into the channel. (TA) stands for Trusted Authority, which is meant to be a totally trustworthy organization. IoT devices (IoT) and Unmanned Aerial Vehicles (UAV) are regarded as untrusted entities, although cloud servers (CS) are regarded as semi-trusted. The Canetti and Krawczyk adversary model (also known as CK-adversary), which is another threat model, is also used [15]. An attacker A in this scenario has the ability to hijack the session key/state on a live session between two network users, and steal confidential credentials.

III. THE PROPOSED FRAMEWORK

A. Deep Learning Module

In this section, a deep learning model is proposed that is used to detect intrusion in the IoT-enabled IA ecosystem.
When dealing with large amounts of data in IoT-enabled IA, deep learning models surpass conventional statistical or machine learning techniques. This phenomenon has been discovered and validated in a number of research articles and publications [1], [11]. As a consequence, when compared to other statistical or machine learning methodologies, deep learning is a better option. This article introduces a novel deep learning architecture for developing a better IDS for IoT-enabled IA. In this approach, we have combined Contractive Sparse AutoEncoder (CSAE), Gated Recurrent Unit Network (GRU), Multi-Layer Perceptrons (MLPs), and softmax classifier. Each of them is explained below.

1) Contractive Sparse AutoEncoder Layer: The AutoEncoder (AE) is a technique for unsupervised learning that consists of two components: the encoder and the decoder. As seen here, the encoder uses a deterministic affine transformation matrix with nonlinearity to transform the input \( X \) into a hidden representation \( Y \) [4].

\[
\mathcal{Y}_f = f(L_1 D_f + \beta_1)
\]  

(1)

where \( L_1 \) is the weight between the input \( D_f \) and the hidden representation \( \mathcal{Y}_f \) and \( \beta_1 \) denotes the bias. The \( \mathcal{Y}_f \) variable is used by the decoder to recreate the output \( \hat{D}_r \).

\[
\hat{D}_r = f'(L_2 \mathcal{Y}_r + \beta_2)
\]  

(2)

where the weight of the hidden representations \( \mathcal{Y}_r \) and \( \hat{D}_r \) is denoted by \( L_2 \) and bias is represented by \( \beta_2 \). \( \hat{D}_r \) is the name given to the reconstruction of \( D_r \). The purpose of AE is to minimize the reconstruction error for a given training set, which is performed by decreasing the following cost function while learning the AE parameters \( L_1, L_2, \beta_1, \beta_2 \).

\[
\begin{align*}
\{L_1, L_2, \beta_1, \beta_2\} &= \arg \min_{L_1, L_2, \beta_1, \beta_2} \{L_{AE} (L, \beta)\} \\
&= \arg \min_{L_1, L_2, \beta_1, \beta_2} \left\{ \frac{1}{2N} \sum_{n=1}^{N} L(D_f, \hat{D}_r) + \lambda \sum_{r=1}^{2} \| D_r \|_F^2 \right\}
\end{align*}
\]  

(3)

The training sample and its reconstruction output are represented by \( D_f \) and \( \hat{D}_r \), respectively. \( N \) is the total number of training samples, and \( L(D_f, \hat{D}_r) \) represents the loss function. Using square error or cross entropy, this can be decreased. \( \lambda \) stands for regularization term, which aids model generalization. From the original dataset, the Sparse AutoEncoder (SAE) attempts to learn sparse yet inherent features. The SAE loss function is stated as, and it is produced by adding a sparsity penalty term to the AE loss function [16].

\[
L_{SAE} (L_1, L_2, \beta_1, \beta_2) = L_{AE} (L, \beta) + \eta \sum_{j=1}^{M} KL (\theta \parallel \hat{\theta}_j)
\]  

(4)

where \( \eta \) determines the weight of the sparse penalty item. To produce a fairly sparse representation, it is typical to use a small value, such as 0.05. Most nodes in the hidden layer are suppressed by SAE using the KL divergence. The following formula is used to determine the KL divergence:

\[
\sum_{j=1}^{M} KL (\theta \parallel \hat{\theta}_j) = \sum_{j=1}^{M} \left\{ (1 - \hat{\theta}_j) \log \frac{1 - \theta}{1 - \hat{\theta}_j} + \hat{\theta}_j \log \frac{\theta}{\hat{\theta}_j} \right\}
\]  

(5)

The average activation value of all training samples on the jth neuron in the hidden layer is \( \hat{\theta}_j \), and the sparsity parameter is \( \theta \). As a result, we may write the SAE loss function as

\[
\begin{align*}
\{L_1, L_2, \beta_1, \beta_2\} &= \arg \min_{L_1, L_2, \beta_1, \beta_2} \{L_{SAE} (L, \beta)\} \\
&= \arg \min_{L_1, L_2, \beta_1, \beta_2} \left\{ \frac{1}{2N} \sum_{n=1}^{N} L(D_f, \hat{D}_r) + \lambda \sum_{r=1}^{2} \| D_r \|_F^2 + \eta \sum_{j=1}^{M} KL (\theta \parallel \hat{\theta}_j) \right\}
\end{align*}
\]  

(6)

Finally, the input data \( D_f \) of the aforementioned cost function is given an explicit regularizer in the form of a Jacobian matrix \( J_f (D_f) \). The model becomes less sensitive to modest changes in the input values as a result of this process. It simply instructs the neurons to ignore little data changes and respond only to larger, more meaningful ones. This ‘penalty’ is only applied during the training of the model, therefore it has no bearing when the network is employed. As a result, the cost function of the Contractive Sparse AutoEncoder (CSAE) can be written as follows:

\[
\begin{align*}
\{L_1, L_2, \beta_1, \beta_2\} &= \arg \min_{L_1, L_2, \beta_1, \beta_2} \{L_{CSAE} (L, \beta)\} \\
&= \arg \min_{L_1, L_2, \beta_1, \beta_2} \left\{ \frac{1}{2N} \sum_{n=1}^{N} L(D_f, \hat{D}_r) + \lambda \sum_{r=1}^{2} \| J_f (D_f) \|_F^2 + \eta \sum_{j=1}^{M} KL (\theta \parallel \hat{\theta}_j) \right\}
\end{align*}
\]  

(7)

where \( \| J_f (D_f) \|_F^2 \) represents the square of the Jacobian matrix’s Frobenius norm. For attack detection, the acquired features are fed into the following module, which combines GRU+MLP+softmax classifiers.

2) Gated Recurrent Unit Network Layer: The GRU network receives the low dimensional feature vector from CSAE layer. The GRU can be single-layered or multi-layered (stacked), depending on the hyperparameter optimization. The \( D_f \) is the input for a given time step \( T \), and the computations are:

\[
\begin{align*}
\mathcal{R}_T &= \sigma (D_f L_K + H_{T-1} L_K + B_K) \, , \\
\mathcal{Z}_T &= \sigma (D_f L_Z + H_{T-1} L_Z + B_Z) \, , \\
\mathcal{C}_T &= \tanh (D_f L_C + H_{T-1} L_C \odot \mathcal{R}_T + B_C) \, , \\
\mathcal{H}_T &= \mathcal{Z}_T \odot \mathcal{H}_{T-1} + (1 - \mathcal{Z}_T) \odot \mathcal{C}_T 
\end{align*}
\]  

(8)

The previous time-hidden step’s state is \( \mathcal{H}_{T-1} \), the reset gate is \( \mathcal{R}_T \), the update gate is \( \mathcal{Z}_T \), the weight parameters are \( L_K \) and \( L_Z \), and the biases are \( B_K \) and \( B_Z \). \( \mathcal{C}_T \) is the hidden candidate state, whereas \( \mathcal{H}_T \) is the new state. ReLU function is denoted
by the letter ⊙, which stands for Hadamard product. The output of a multi-layered GRU network is the hidden state \( h_f \) of the preceding layer, and there is no dropout between the layers.

3) Multi-Layer Perceptrons Layer: The dense layer of MLP uses the output vector of the GRU layer \( h_f \) to represent the output activation of its node in the following way:

\[
D_1(\alpha) = f(L_1^T h_f + b_1)
\]

\[
D_2(M) = f(L_2^T D_1(\alpha) + b_2)
\]

The weight matrix \( L_1 \) connects the output of the GRU layer with the first hidden layer, and the bias vector \( b_1 \) is connected with that layer. where \( L_2 \) is the weight matrix linking the first and last hidden layers, and \( b_2 \) is the bias associated with the weight matrices.

4) Softmax Classifier Layer: Finally, the softmax classifier is integrated with the proposed deep learning architecture to determine the likelihood that the projected type belongs to a certain class. Eq.12 is used to compute the loss function.

\[
f(M_k) = \frac{e^{M_k}}{\sum_{j=1}^{C}e^{M_j}}, \quad k = 1, \ldots, C
\]

\[
LOSS(y, M) = -\sum_{k=0}^{C} y_k \log (f(M_k))
\]

B. Smart Contract Module

In this subsection, we have discussed the steps used by the proposed smart contract-based authentication and key management module.

1) Initialization Phase: This phase explores, how trusted authority (TA) chooses the parameters to register the entities of framework. The detailed process is discussed below. First, a non-singular elliptic curve is selected by the TA i.e., \( E_{\beta}(\beta, \gamma) \)

\[
S^2 = T^3 + \alpha T + \gamma \pmod{W_n},
\]

where \( W_n \) is a large prime value and \( \beta, \gamma \in \mathbb{F}^* = \{1,2,3,\ldots,W_n\} \) are the two points i.e, infinity point and zero \( \mathbb{F}^* \). Further, the TA chooses a base point \( BP \in E_{\beta}(\beta, \gamma) \) of order \( \mathbb{F}^* \) as bigger as \( W_n \). Furthermore, TA chooses a cryptographic hash function \( (\mathcal{H})() \) using SHA-512. In addition, TA chooses an identity \( ID_{TA} \), and picks a private key \( TA_{pk} \) and evaluates a public key \( TA_{pk} = TA_{pk} * BP \). Finally, the TA preserves a private key \( (TA_{pk}) \) secret and disseminates public parameters \( \{E_{\beta}(\beta, \gamma), BP, (\mathcal{H})(),\} \).

2) Registration Phase: This phase describes a registration process of each entities and shares the communication parameters. (a) \( ID_{TA} \) Registration : The TA registers a IoT nodes \( ID_T \), where \( \{ID_T\} = \{1,2,\ldots,ID_D\} \)

Step-1: The TA chooses an unique identity \( ID_{TA} \) for registration of IoT devices. Further, the TA evaluates a pseudo identity \( PSID_{TA} = H(ID_{TA}) \) and generates a certificate \( CRT_{TA} = TA_{pk} + H(PSID_{TA}) \pmod{W_n} \).

Step-2: TA chooses a random number \( RN \), and evaluates a partial private key i.e., \( pP_{TA} = H(ID_{TA}) \pmod{RN} \), and evaluates a public key \( PB_{TA} = pP_{TA} * B \) for each \( ID_T \) and preserves registration information \( (PSID_{TA}, pP_{TA}, CRT_{TA}) \) on board units \( OB_{TA} \) of IoT devices. Finally, TA deletes a partial private key \( pP_{TA} \) and disseminates the public key \( PB_{TA} \) for communication.

(b) \( ID_{TA} \) Registration : The TA registers a UID \( UID \), where \( \{UID\} = \{1,2,\ldots,UID_{TA}\} \)

Step-1: The TA chooses an unique identity \( ID_{TA} \) for registration of UID. Further, TA evaluates a pseudo identity \( PSID_{TA} = H(ID_{TA}) \) and generates a certificate \( CRT_{TA} = TA_{pk} + H(PSID_{TA}) \pmod{W_n} \), and evaluates a public key \( PB_{TA} = pP_{TA} * B \) for each \( UID_{TA} \) and preserves registration information \( (PSID_{TA}, pP_{TA}, CRT_{TA}) \) on board units \( OB_{TA} \) of IoT devices. Finally, TA deletes a partial private key \( pP_{TA} \) and disseminates the public key \( PB_{TA} \) for communication.

Step-2: A random number is chosen by TA i.e., \( RN \), and evaluates a partial private key i.e., \( pP_{TA} = H(ID_{TA}) \pmod{RN} \), and evaluates a public key \( PB_{TA} = pP_{TA} * B \) for each \( UID_{TA} \) and preserves registration information \( (PSID_{TA}, pP_{TA}, CRT_{TA}) \) on board units \( OB_{TA} \) of IoT devices. Finally, TA deletes a partial private key \( pP_{TA} \) and disseminates the public key \( PB_{TA} \) for communication.

3) Key Agreement and Authentication Phase: We have discussed various steps used in key agreement and authentication.

(i) IoT nodes to UA Authentication

Step-1: Each node \( ID_T \) chooses an unique random number \( dr \) with valid timestamp \( T_{STP} \), and generates a certificate \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)

Step-2: After receiving successful message \( M_c \) timestamp gets validated \( T_{STP} \), using UA \( |T_{STP} - T_{STP}| < \Delta T \), after successful verification of timestamp, UA checks certificates using \( CRT_{TA} \). If it matches successful then UA receives \( PSID_{TA} \) respect to \( PB_{TA} \) from the database and evaluates \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)

(i) IoT nodes to UA Authentication

Step-1: Each node \( ID_T \) chooses an unique random number \( dr \) with valid timestamp \( T_{STP} \), and generates a certificate \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)

Step-2: After receiving successful message \( M_c \) timestamp gets validated \( T_{STP} \), using UA \( |T_{STP} - T_{STP}| < \Delta T \), after successful verification of timestamp, UA checks certificates using \( CRT_{TA} \). If it matches successful then UA receives \( PSID_{TA} \) respect to \( PB_{TA} \) from the database and evaluates \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)

(i) IoT nodes to UA Authentication

Step-1: Each node \( ID_T \) chooses an unique random number \( dr \) with valid timestamp \( T_{STP} \), and generates a certificate \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)

Step-2: After receiving successful message \( M_c \) timestamp gets validated \( T_{STP} \), using UA \( |T_{STP} - T_{STP}| < \Delta T \), after successful verification of timestamp, UA checks certificates using \( CRT_{TA} \). If it matches successful then UA receives \( PSID_{TA} \) respect to \( PB_{TA} \) from the database and evaluates \( L_{TA} = H(dr) \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \pmod{\mathbb{F}^*} \)
Step-3: Further, UAV picks a unique random number $u_{AVT}$ ∈ $Z_q$ and valid timestamp gets recorded $TSTP_2$ and generates a temporary identity $PPR_{new}$ and evaluates $UAV_i=h(PPR_{new} || TSTP_2 || UAV_i)$. $UAV_i$ and $UAV_1$ as $E_{PPR_{new}}(u_{AVT})$. Next, UAV ($UAV_i$) evaluates a session key $S_{ES} = h(PPR_{new} || TSTP_2)$, $PPR_{new}^* = PPR_{new} \oplus h(PSID_{UAV} || PPR_{new} || TSTP_2)$, and $UAV_i = h(PPR_{new}^* || CRID || PSID_{UAV} || TSTP_2)$ and creates reply message $M_1 = \{PPR_{new}^*, UAV_i, CRID, PSID_{UAV}, TSTP_2\}$ and transmit to $CS$ through open channel.

Step-4: after successful receive of reply message ($M_1$) by $IoT_i$ whether $TSTP_2^* - TSTP_2 \leq \Delta T$ is denoting correct timestamp, if it matches successful, then $IoT_i$ checks certificate by $CR_{TSTP_2}$. $B=PB_{IA} + h(PSID_{UAV} || PB_{IA})$. Further, $IoT_i$ uses decryption, the $UAV_i$ computes $UAV_i = D_{PB_{IA}}(u_{AVT})$. $UAV_i$ evaluates $UAV_i = h(PSID_{UAV} || CRID || PSID_{UAV} || TSTP_2)$ and verify if $UAV_i = UAV_i$ then $IoT_i$ evaluates $PPR_{new} = PPR_{new}^*$ and $h(PSID_{UAV} || PPR_{new} || TSTP_2)$ and evaluates a session key $S_{ES} = h(PSID_{UAV} || L_1 \oplus UAV_i || TSTP_2 || TSTP_2)$ and disseminates to $UAV_i$. Further, $IoT_i$ chooses a valid timestamp $TSTP_2$ and verify session key $S_{ES} = h(PSID_{UAV} || TSTP_2)$ and makes changes to $PPR_{new}$ and $PPR_{new}^*$ in database. Furthermore, $IoT_i$ generates acknowledgement message $M_2 = \{S_{ES}, TSTP_2\}$ and transmit to $UAV_i$ through open channel.

Step-5: after successful receive of acknowledgement message $M_i$ timestamp gets validated $TSTP_2^*$, by $UAV_i$ & $TSTP_2^* - TSTP_2 \leq \Delta T$ is denoting correct timestamp. Next $UAV_i$ checks $S_{ES} = h(SID_{UAV} || TSTP_2)$. After successful match, the $UAV_i$ makes establishment of the session key $S_{ES} = h(SID_{UAV})$ by $IoT_i$. Finally, $UAV_i$ makes changes with $PPR_{new}$ and $PPR_{new}^*$ in database.

(ii) UAV to CS Authentication

Step-1: $UAV_i$ chooses an unique random number $d_{AVT}$ ∈ $Z_q$ and valid timestamp $TSTP_2$ and evaluates $L_1 = h(PSID_{UAV} || CRID_{UAV} || d_{AVT} || TSTP_2)$. $L_1$ makes encryption $L_2 = E_{PB_{IA}}(L_1)$. Furthermore, $UAV_i$ evaluates $L_1 = h(L_2 || CRID_{UAV} || PSID_{UAV} || PPR_{new} || TSTP_2)$ and creates message request for access $M_i = \{PSID_{UAV}, PPR_{new}, TSTP_2, L_2, L_1\}$ and transmit to $CS$ through open channel.

Step-2: after successful receive of message $M_i$ timestamp gets validated $TSTP_2^*$ by $CS$ & $TSTP_2^* - TSTP_2 \leq \Delta T$. If timestamp validated successfully, then $CS$ checks certificate by $CR_{TSTP_2}$. $B = PB_{IA} + h(PB_{IA} || PB_{IA})$ if it matches successfully, then $CS$ receives $S_{ES} = h(SID_{UAV} || TSTP_2)$ with respect to $PPR_{new}$ from the database and evaluates $L_2 = h(L_2 || PSID_{UAV} || CRID_{UAV} || PPR_{new})$ to verify whether $L_2 = L_1$. If it matches successfully, then $CS$ uses decryption $L_1 = D_{PB_{IA}}(L_2)$.

Step-3: Further, $CS$ picks an unique random number $c_{AVT}$ ∈ $Z_q$ and valid timestamp $TSTP_2$ and generates temporary identity $PPR_{new}^*$ and evaluates $CS_i = h(PSID_{UAV} || PSID_{CS} || CRID || TSTP_2)$ and uses encryption $CS_i = E_{PPR_{new}^*}(c_{AVT})$. Furthermore, $CS$ generates a session key $S_{ES} = h(PPR_{new}^* || CRID || PSID_{UAV} || TSTP_2)$ and creates reply message $M_1 = \{PPR_{new}^*, CS_{i}, CRID, PSID_{CS}, TSTP_2\}$ and transmit to $UAV_i$ through open channel.

Step-4: after successful receive of reply message ($M_1$) from $CS$ , timestamp gets validated $TSTP_2^*$ by $UAV_i$, i.e., $TSTP_2^* - TSTP_2 \leq \Delta T$ is denoting valid timestamp or invalid timestamp. if matches successfully, then $UAV_i$ checks for certificate using $CR_{TSTP_2}$. $B = PB_{IA} + h(PB_{IA} || PB_{IA})$. $UAV_i$ uses decryption $CS_i = D_{PB_{IA}}(c_{AVT})$. Furthermore, $UAV_i$ evaluates $CS_i = h(PSID_{UAV} || CRID || PSID_{CS} || TSTP_2)$ and verify if $CS_i = CS_i$ then $UAV_i$ evaluates $PPR_{new}^* = h(PSID_{UAV} \oplus h(PSID_{CS} || PPR_{new} || TSTP_2))$ generates a session key $S_{ES} = h(PSID_{UAV} || CRID || PSID_{CS} || TSTP_2)$ and disseminates to $CS$. Furthermore, $UAV_i$ chooses valid timestamp $TSTP_2$ and verify session key $S_{ES}$ by $CS$, $UAV_i = h(SID_{UAV} || TSTP_2)$ and makes changes with $PPR_{new}$ and $PPR_{new}^*$ in open channel.

Step-5: after successful receive of acknowledgement message $M_i$ gets validated $TSTP_2^*$ by $CS$ i.e., $TSTP_2^* - TSTP_2 \leq \Delta T$ is denoting valid timestamp or invalid timestamp. if matches successfully, then $UAV_i$ checks certificate by $CR_{TSTP_2}$. $B = PB_{IA} + h(PB_{IA} || PB_{IA})$. $UAV_i$ uses decryption $CS_i = D_{PB_{IA}}(c_{AVT})$. $CS_i$ generates a session key $S_{ES} = h(PSID_{UAV} || CRID || PSID_{CS} || TSTP_2)$ and creates reply message $M_1 = \{PPR_{new}^*, CS_{i}, CRID, PSID_{CS}, TSTP_2\}$ and transmit to $UAV_i$ through open channel.

Algorithm 1 Proof-of-Authority (Aura Algorithm) for Block Verification and Addition

1: $State$: $CS \in D_b$. Set of miners.
2: $f_i = (A_i, f_i)$. $A_i$, local blockchain of node $f_i$ is a DAG of block $A_i$ and pointer $f_i$.
3: $b$ denotes Block records
4: parent, preceeding node of $b$
5: miners, who mines and sign block $b$
6: step, new block added to the network
7: $duration$, each step takes time to validate and added
8: function $PROPOSE()$
9: $while$ True do
10: $step ← CT / duration$, CT → clock time
11: $if k ∈ CS_i ∧ step mod |CS_i| == k$ then
12: $b.parent ← b(C_i)$, lb → last block
13: $b.CS ← f_i$
14: $b.step ← step$
15: $CS_i ← (A_i, b, f_i, b.parent)$
16: disseminate ($C_i$)
17: sleep($duration$)
18: $end while$
19: $end function$
20: function $SCORE(A_i, f_i)$
21: $return UNIT256-MAX * height($A_i, f_i$) - step-num($A_i, f_i$)$
22: $end function$
23: function $DELIVER(A_i, f_i)$
24: $if Score($A_i, f_i$) > Score($A_i, f_i$) then
25: $Score($A_i, f_i$) ← Score($A_i, f_i$)$
26: $end if$
27: $end function$
28: function $isDECIDED(b_k)$
29: $if b_k.UAV \in A_i ∪ b_i.step > = b.step$ return $(|V| * 2 > |CS_i|)$
30: $end function$
TABLE I: Class-wise performance analysis of proposed IDS

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PR</td>
<td>93.95%</td>
<td>89.30%</td>
<td>83.98%</td>
<td>38.88%</td>
<td>98.23%</td>
<td>97.37%</td>
<td>96.25%</td>
<td>96.97%</td>
<td>90.01%</td>
</tr>
<tr>
<td>DR</td>
<td>88.90%</td>
<td>79.80%</td>
<td>88.56%</td>
<td>98.39%</td>
<td>90.39%</td>
<td>90.23%</td>
<td>96.25%</td>
<td>94.23%</td>
<td>98.88%</td>
</tr>
<tr>
<td>F1</td>
<td>90.97%</td>
<td>77.77%</td>
<td>90.19%</td>
<td>86.38%</td>
<td>96.21%</td>
<td>95.98%</td>
<td>96.25%</td>
<td>97.87%</td>
<td>89.84%</td>
</tr>
<tr>
<td>FAR</td>
<td>0.00005</td>
<td>0.000038</td>
<td>0.0000125</td>
<td>0.00008</td>
<td>0.000018</td>
<td>0.000024</td>
<td>0.00008</td>
<td>0.000038</td>
<td>0.00005</td>
</tr>
</tbody>
</table>

TABLE II: Comparison of multi-vector DR (%) with some commonly used baseline techniques

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PR</td>
<td>100.00%</td>
<td>95.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
</tr>
<tr>
<td>DR</td>
<td>100.00%</td>
<td>95.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
</tr>
<tr>
<td>F1</td>
<td>100.00%</td>
<td>95.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
<td>90.00%</td>
</tr>
<tr>
<td>FAR</td>
<td>0.00005</td>
<td>0.000038</td>
<td>0.0000125</td>
<td>0.00008</td>
<td>0.000018</td>
<td>0.000024</td>
<td>0.00008</td>
<td>0.000038</td>
<td>0.00005</td>
</tr>
</tbody>
</table>

V. PERFORMANCE ANALYSIS

The experiments were executed on a Tyrope PC with two 2.20GHz Intel CPUs and 128 GB of RAM. The intrusion detection system was developed using the TensorFlow package Keras. The Ethereum Rinkey network was used to create the smart contract module. The CSAE layer was trained for 10 epochs using two hidden layers containing (64,32) neurons, whereas GRU used two hidden layers with (64,32) neurons, MLP used two hidden layers with (16,8) and last layer has softmax classifier, Adam Optimizer, ReLU activation, categorical cross-entropy as loss function, and 100 batch size for 10 epochs. The intrusion performance was evaluated using the CICIDS-2017 dataset, which contains 390222 attack and 2035505 normal instances [17]. We pre-processed both datasets using the techniques outlined in [18] with 70% training and 30% testing sets. This paper employs a variety of performance metrics such as, accuracy, detection rate, precision score, F1 score and false alarm rate. However, to calculate these values, various parameters are used such as, True Positive ($\alpha$), True Negative ($\gamma$), False Positive ($\beta$), and False Negative ($\delta$) determines correct classified attack instances, correct classified normal instances, normal observations classified as attack instances, and attack observations classified as normal instances, respectively [1]; Accuracy (AC): The percentage of all correctly identified regular and attack instances is determined by AC that is $AC = \frac{\alpha + \beta}{\alpha + \beta + \gamma + \delta}$. Detection Rate (DR): The appropriate proportion of attacks identified is determined by DR or Recall (RC) that is, $DR = \frac{\alpha}{\alpha + \beta}$. Precision (PR): PR is calculated by dividing the number of attack behaviors observed by the total number of observations classified as an attack, $PR = \frac{\alpha}{\alpha + \beta}$. F1 Score: The weighted average of PR and DR/RC is determined by the F1 score, that is, $F1 = 2 \times \frac{PR \times RC}{PR + RC}$. False Alarm Rate (FAR): FAR identifies cases of attack that were incorrectly identified, that is, $FAR = \frac{\beta}{\alpha + \beta}$. 

A. Deep Learning Module Analysis

The performance of DL approach is evaluated using a variety of assessment metrics. The proposed CSAE technique’s accuracy vs. loss is depicted in Fig. 2. Despite being employed to extract low-dimensional features, the CSAE method learned the dataset effectively, with a validation accuracy of 87.92% and a validation loss of 0.0546%. Table I shows the class-wise performance of the proposed model. It is observed that the values for PR, DR, and F1 score is high, and FAR is close to 0%. We have also compared the DR of the proposed model...
with other baseline techniques in Table II. It is seen that the proposed model outperformed these baselines for the majority of the vectors present in the dataset. Finally, as shown in Fig. 3, the overall performance of proposed model is compared with traditional approach. It is seen that the proposed model has achieved higher values and outperformed RF, DT and NB.

B. Smart Contract Module Analysis

The smart contract study shown in Fig. 4 and Fig. 5 evaluates transaction upload time, block mining, block formation, and off-chain storage. The Fig. 4a, shows the upload time over IPFS storage layer for different transactions. The Fig. 4b and Fig. 5a, shows block mining and block creation time with different number of nodes (NS) and transactions (Tx). It can be observed that the execution time linearly increasing as peers increasing in the network. The Fig. 5b, shows off-chain storage size in KB over IPFS for varying number of Tx. It can be observed that, storage size is increasing as the number of Tx increasing in the network.

VI. CONCLUSION

In this article, we designed a DL and smart contract-assisted secure data sharing framework for IoT-based intelligent agriculture. Specifically, a novel DL module was designed that combined contractive sparse autoencoder with gated recurrent unit, multi-layer perceptrons and softmax classifier to detect intrusion in the network. In smart contract module, first authentication, key management scheme was proposed. The normal transactions received from DL-based IDS were mined by cloud servers using smart contract-based PoA (aura algorithm) consensus technique. The validated transactions were added to the IPFS-based storage layer and returned cryptographic hash was stored on blockchain ledger. Experimental analysis of DL and smart contract module proves the effectiveness of the proposed framework. The future work includes the performance evaluation in terms of scalability and latency using different real-world datasets.

REFERENCES


Randhir Kumar received his Ph.D. degree in Information Technology, National Institute of Technology Raipur, Raipur, India in 2021. He is currently working as a Post-Doctoral Researcher with the Department of Electrical Engineering, Indian Institute of Technology Hyderabad, India. He has published his research article in leading journal and conferences from IEEE, Elsevier, Springer, and John Wiley. He has published more than 40 research article in the reputed journals and conferences. His paper has been published in some of the high impact factor journals such as – IEEE Internet of Things, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network Science and Engineering, IEEE Transactions on Green Communications and Networking, IEEE Transactions on Industrial Informatics, IEEE COMSNETs, IEEE ICC, Computer Networks, JPDC, and Transactions on Emerging Telecommunications Technologies (ETT Wiley). His research interest includes cryptographic techniques, information security, blockchain technology, and web mining. He is also an IEEE Member.

Prabhat Kumar (S’20, M’22) received his Ph.D. degree in Information Technology, National Institute of Technology Raipur, Raipur, India under the prestigious fellowship of Ministry of Human Resource and Development (MHRD) funded by the Government of India in 2022. Thereafter, he worked with Indian Institute of Technology Hyderabad, India as a Post-Doctoral Researcher under project “Development of Indian Telecommunication Security Assurance Requirements for IoT devices”. He is currently working as Post-Doctoral Researcher with the Department of Software Engineering, LUT School of Engineering Science, LUT University, Lappeenranta, Finland. He has many research contributions in the area of Machine Learning, Deep Learning, Federated Learning, Big Data Analytics, Cybersecurity, Blockchain, Cloud Computing, Internet of Things and Software Defined Networking. He has authored or co-authored over 25+ publications in high-ranked journals and conferences. One of his Ph.D. publication was recognized as a top cited article by WILEY in 2020-21.

Ahamed Aljihani received the M.S. degree in computer science from the University of Colorado Denver, Denver, CO, USA, and the Ph.D. degree in computer science/information security track from The Catholic University of America, Washington, DC, USA. He is currently an Assistant Professor and the Chair of the Department of Computer Engineering, College of Computing and Information Technology, University of Tabuk, Saudi Arabia. His current research interests include information security, network security and privacy, secure system design, and system development.

A.K.M. Najmul Islam is an Associate Professor at Software Engineering, LUT University, Finland. He is an adjunct professor of Information Systems at Tampere University, Finland. He has received his PhD from the University of Turku, Finland, and M.Sc. from Tampere University of Technology, Finland. He has published in other highly ranked journals such as IEEE Transactions on Industrial Informatics (IITI), IEEE Transactions on Artificial Intelligence, IEEE Access, Computers in Industry, Computers & Education, Journal of Strategic Information Systems, European Journal of Information Systems and Information Systems Journal, Technological Forecasting and Social Change, International Journal of Information Management, Information Technology & People, Computers in Human Behavior, Internet Research, Communications of the AIS, among others. He is currently serving as a Senior Editor for Information Technology & People journal.

Alireza Jolfaii is an Associate Professor of Networking and Cyber Security in the College of Science and Engineering at Flinders University, Adelaide, Australia. He is a Senior Member of the IEEE and a Distinguished Speaker of the ACM. His main research interest is in Cyber-Physical Systems Security. He has published over 100 papers, which appeared in peer-reviewed journals, conference proceedings, and books. Before Flinders University, he has been a faculty member with Macquarie University, Federation University, and Temple University in Philadelphia, PA, USA. He received the prestigious IEEE Australian council award for his research paper published in the IEEE Transactions on Information Forensics and Security. Dr. Jolfaee is the IEEE Consumer Technology Publication Board member and the Editor-in-Chief of the Consumer Technology Society World Newsletter. He has served as the Regional Chair of the IEEE Technology and Engineering Management Society’s Membership Development and Activities for Australia. He has served as a program co-Chair, a track Chair, a session Chair, and a Technical Program Committee member, for major conferences, including IEEE TrustCom and IEEE ICCCN.

Sahil Garg (S’15, M’18) received the Ph.D. degree from the Thapar Institute of Engineering and Technology, Patiala, India, in 2018. He is currently working with the Resilient Machine Learning Institute (ReMI), Montreal, Canada. Prior to this, he worked as a Postdoctoral Research Fellow at ÉTS, Montreal and MITACS Researcher at Ericsson, Montreal. He has many research contributions in the area of Machine Learning, Big Data Analytics, Knowledge Discovery, Cloud Computing, Internet of Things, Software Defined Networking, and Vehicular Ad-hoc Networks. He has over 80 publications in high ranked Journals and Conferences, including 50+ top-tier journal papers and 40+ top-tier conference articles. He was the recipient of the prestigious Visvesvaraya PhD fellowship from the Ministry of Electronics & Information Technology under Government of India (2016-2018). He has been awarded the 2021 IEEE Systems Journal Best Paper Award; the 2020 IEEE TCSC Award for Excellence in Scalable Computing (Early Career Researcher) and the IEEE ICC best paper award in 2018 at Kansas City, Missouri. He is currently a Managing Editor of Springer’s Human-centric Computing and Information Sciences (HCIS) journal; and an Associate Editor of IEEE Network Magazine, IEEE Transactions on Intelligent Transportation Systems, Elsevier’s Applied Soft Computing (ASoC), and Wiley’s International Journal of Communication Systems.