Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.034381
Article

Federated Blockchain Model for Cyber Intrusion Analysis in Smart Grid Networks

N. Sundareswaran* **and S. Sasirekha**

Department of Information Technology, Sri Sivasubramaniya Nadar College of Engineering, Chennai, 603110, India
*Corresponding Author: N. Sundareswaran. Email: sundareswarann@ssn.edu.in
Received: 15 July 2022; Accepted: 22 September 2022

Abstract: Smart internet of things (IoT) devices are used to manage domestic and industrial energy needs using sustainable and renewable energy sources. Due to cyber infiltration and a lack of transparency, the traditional transaction process is inefficient, unsafe and expensive. Smart grid systems are now efficient, safe and transparent owing to the development of blockchain (BC) technology and its smart contract (SC) solution. In this study, federated learning extreme gradient boosting (FL-XGB) framework has been developed along with BC to learn the intrusion inside the smart energy system. FL is best suited for a decentralized BC-enabled system to adapt learning models for trustworthy and reliable transactions. Many features and attributes of the Third International Knowledge Discovery and Data mining Tools Competition (KDD Cup 1999) dataset have been used in this study to perform experimental analysis. The likelihood of intrusions in the network is mathematically stated. The participant nodes run the BC based FL-Smart Contract (SC) algorithms to detect network intrusions. FL provided aggregated learning results from the experiment that was 99% accurate in predicting network intrusion. The experimentally determined block storage gain and retrieval gain were 97.5% and 95.4% respectively. The intrusion in the smart grid network was evaluated, and the data indicated that there was 1.2% illegal access. Moreover, the learning system's accuracy, retrieval and storage intrusions, legal access and transaction processing times were considered for comparison. The proposed system outperformed contemporary research-developed systems targeted for the same application. Therefore, this study provides a guaranteed intrusion learning system and secure transaction system for smart grids.

Keywords: Blockchain; federated learning system; intrusion detection; internet of things; smart grids

1 Introduction

Energy systems include various sources of energy, such as solar systems, wind energy farms, and traditional sources that are extracted and managed by government and private energy sectors in adherence to energy policies. Nowadays, the energy sector is deploying IoT smart meters and sensors to simplify its operations management [1]. However, smart meter measurement information is prone to cyber-attacks because the information is shared through the internet to a remote site. Smart IoT sensors monitor the

functioning of transformers, and the status of power lines, in real-time [2]. The existing energy systems meet the need for IoT enabled smart systems for monitoring and controlling energy sources distributed throughout the country. Smart energy data storage systems such as IoT smart meters, big data in smart grids and cloud based active power-data-storage systems are storing energy consumption data, sensors data from multisensory systems in the smart grid, data from smart grid heat-control systems. All these storage systems are subject to various intrusion cyber-attacks, such as hijacking authentication and authorization, distributed denial of service (DDos), malicious bot injection, and byzantine attacks [3]. In 2018, the French smart energy network reported that malware affected power distribution to nearly 50000 consumers. In Tasnee, Saudi Arabia, a malware attack stopped many generators of the smart grid system, and in Ivano-Frankivsk, Ukraine, a Trojan named "*Black Energy"* took control of the data server of the national smart grid system. Critical information on systems such as power plants, power distribution stations, power lines, and smart meters needs to be kept secret [4,5]. The serious issue with existing smart energy systems is that it is possible to hijack the command operation of the server, which leads to total system collapse and inaccessibility. Therefore, control system servers need to be secured against intrusion. Numerous cyber security systems exist, however, blockchain (BC) technology is a potentially strong cyber security system because it employs distributed computing and facilitates resource-sharing between multiple destinations or servers. BC provides a permanent solution against cyber threats by implementing a distributed consensus. The BC distributed consensus algorithm executes smart grid transactions/operations through multiple authenticated peers/nodes in the smart grid network [6,7]. The centralized cloud solution for sensitive data storage is replaced by a distributed multiple-data-server storage system. The communications between servers/nodes are stored as communication history in the distributed ledger of BC. This makes it easy to detect intrusive behavior in networks of nodes. BC technology implements an inbuilt digital signature to authenticate all communications between service providers and consumers throughout the network. Smart contract (SC) replaces third parties by establishing trustful communication between the energy stockholders. In this study, BC and Federated Learning (FL) systems have been combined to effectively detect and erase intrusions in the smart grid network. FL is a mixture of distributed and machine learning models introduced by Google. It is meant for high-level predictions to support any real-time application. In our proposed smart grid system, FL has been introduced to learn the intrusion or malicious contents in the system. FL is best suited for adapting machine learning models to a decentralized BC-enabled system to enable secure, reliable, and transparent transactions for any system domain [8,9]. BC SCs are used to implement FL algorithms between service providers and consumers in distributed smart grids for secure communication. FL has been applied to predict intrusion based on payload signatures. The BC SC verifies the signatures and reports to the server in case a cyber-intrusion signature is found [10]. The SC also maintains a history of the results using distributed ledger technology (DLT). The history serves as a reference to all participant nodes in the smart grid network and ensures transparency for all its stakeholders. Federated Machine Learning (ML) models are written as SCs, and they are implemented in the cluster of participant nodes of the smart energy network. In this study, the XGBoost ML algorithm is used to learn intrusions because it provides a strong prediction model for the participant nodes. The prediction results from these nodes are aggregated on the server. The aggregated result would be recorded using DLT, and necessary actions to eradicate intrusions would be taken in a timely manner by the BC smart energy network. FL enables global discovery of intrusion from the learning results of local participant nodes without affecting the functionality of the network [11]. FL provides promising results in detecting intrusion signatures within the network from scattered data samples in local nodes. A smart grid network trained using an FL algorithm reduces the risk of intrusion in all possible ways in the distributed environment by invoking the local training models to build a global model by proving and confirming the hidden intrusion payload signatures. The following paragraph summarizes the key contributions of this study.

The KDD dataset and SC modules are used to identify malicious intrusions in the network of participating nodes. Using mathematical models in the sample space, the network's numerous intrusion cases are defined. The implementation includes scanning the vulnerabilities of the network using security tools, establishing FL-SC modules to detect intrusion, testing FL-accuracy (%), installing Ethereum BC tools on participating nodes and servers, and analyzing storage and retrieval operations by setting threshold times to spot intrusions. Finally, to support the effectiveness of the proposed system, the findings of this study are compared with those of current works.

2 Related Works

Hassan et al. [12] proposed the BC solution analysis of smart grid systems for various platforms, such as Ethereum, Hyperledger, Tendermint, and the Energy Web Foundation with respect to several important parameters of smart grid networks, such as data, automation, consensus, and networks. BC customization for grid monitoring, security certificates, wholesale, and incentives is also considered for real-time smart energy networks. Andoni et al. [13] proposed the study of 100 BC research projects for integration with smart energy networks. They discussed the commercial market challenges for the mapping of BC technology with smart grid networks, applied BC system taxonomies such as proof-of-work, distributed consensus, and practical byzantine fault tolerance (PBFT) to various use cases of energy sectors. They also discussed an IoT smart network and its automation with systematic energy asset management. Kong et al. [14] proposed a linear network coding for FL systems to train image classifiers as shared learning models from participant nodes to secure the private data of the system. This work compared the accuracy of the proposed learning model with respect to FL clients and servers. The system was found to be best suited for decentralized network systems and privacy-preserving network applications. Xia et al. [15] proposed the federated XGBoost learning model to achieve security and privacy in network systems. They compared the accuracy, number of communication exchanges among participants and throughput of their method to those of standard machine learning approaches. The proposed model accepted the possible dataset and features such as Acrene, Biodes, Credit and German to apply learning models and analyze them with the fairness, accountability, transparency and ethics (FATE) model. Their study also considered the security analysis of the system against passive attacks and private data attacks on central servers and owners, and proved from the fixed threshold value of the proposed model that data was not leaked. It also proposed a key exchange scheme for establishing trust with the users of neural networks. The authors used FL to aggregate the results on the server and verified the correctness and efficiency of the results produced by the server by leveraging a deep security analysis and verification scheme on federated training sets. A novel optimization method for distributed multi-task learning (MTL) was employed to learn models for related tasks [16]. Federated results were produced from k distributed nodes $\{k_1, k_2,\ldots, k_m\}$. The structures, modeling and relationships between generated learning results by nodes were optimized with the help of the proposed MTL framework. Simulations were performed using the Google glass dataset, vehicle sensors, and human activity recognition to prove the optimization results. Table 1 illustrates whether the following parameters were considered (✓) or not considered (✗) in the studies mentioned in the literature survey (1) application, (2) system security, (3) mathematical model, (4) proof of efficiency and (5) analysis of privacy. The proposed federated XGBoost BC authentication promises to fulfill all of these parameters.

Luo et al. (S. No 1) in Table 1 considered the application of a smart energy storage system, constructed a mathematical model to determine the electrical energy storage potential of the system, and proved the energy storage efficiency of the real-time system. This work did not consider the security of the system or analyze its privacy. The actual security issues of smart grids are listed and described in Kim et al. [17]. The remaining articles (S. No 2–6) in Table 1 are analyzed with respect to the analysis parameters. The proposed work is an

experimental study that is discussed in Section 5 in which all analysis parameters are considered and shown as the results of the experiment. The following are the key advantages of this work.

Table 1: Literature survey

S. no	Authors	Journal & year	1	2	3	4	5
1	X. Luo et al.,	Journal of Applied Energy, 2015	✓	✗	✓	✓	✗
2	M. Li et al.,	Journal of Pattern Recognition Letters, 2020	✓	✓	✗	✓	✓
3	N. U. Hassan et al.,	IEEE Industrial Electronics Magazine, 2019	✓	✓	✗	✓	✓
4	M. Andoni et al.,	Journal of Renewable and Sustainable energy, 2019	✓	✓	✓	✗	✓
5	Y. Chen et al.,	Information Sciences, 2020	✗	✓	✗	✓	✗
6	Proposed work	-	✓	✓	✓	✓	✓

To combat malicious intrusion cyber-attacks on the crucial power grid system, the BC and FL systems are combined. By combining the outcomes of several participant nodes in the distributed environment, FL offers a robust prediction model. Energy data is stored and retrieved by the BC as blocks. We noticed that the history of transactions for both store and retrieve operations was automatically documented when we used Ethereum BC and its SC modules to secure sensitive data. Additionally, this document served as a log for the analysis of storage and retrieval operations. Intrusions were purposefully initiated in the target system after analysis of the documents in the BC logs that were used for storing and retrieving data. FL aggregation correctly delivered all intrusions within the allotted period. The accuracy, intrusion rates, and transaction processing times of the proposed system were compared to those published in contemporary studies. The proposed system produced the best outcomes.

3 Federated XGB

The performance of various learning solutions has been expressed using various metrics. For example, classification and regression tree, AdaBoost, random forest, and XGB have exhibited 96.74% accuracy, 97.5% specificity, 97.3% sensitivity, and 98.77% area under curve respectively [18,19]. Therefore, in this work, the federated XGBoost system has been selected to model intrusion detection. Federated XGB generates a strong prediction model by linearly combining small classifiers. It works under the federated gradient boosting framework and creates a final prediction model tree through a set of earlier individual prediction models obtained from the participant nodes. Consider that a set of input variables (p) and output variables (q) are distributed uniformly but randomly to the scattered participant nodes in the smart grid network. The set of inputs $P = \{p_1, p_2, p_3,\ldots, p_n\}$, and the outputs are of the form $\{q, p_i\}$, where $i = 1, 2,\ldots, n$. The outputs have been obtained by mapping the training sample results to find intrusion signatures from p to q. The set of input criteria such as $p_1, p_2, p_3,\ldots, p_n$ is analogous to a set of decision trees used to determine whether a signature is found. The bagging process involves bootstrap aggregation to combine all sets of input decision trees. Each step of the bootstrap aggregation in the server and the votes are combined to form the final decision tree [20]. Meanwhile, the predictions in each round are tested with the expected outcome. The difference between the predicted and expected outcomes represents the error or loss of the training model. Thereafter, the gradient is calculated to describe the depth of the error model through the loss function. The mathematical partial derivation is formed by the loss function to minimize the error rate in the training model. The regression tree is a model for prediction.

The model is represented as a function f(x) as shown in Eq. (1) to minimize the depth of error or loss function.

$$f(x)= \min\left(\gamma \sum_{i=1}^{n} L(x,\ \gamma)\right) \tag{1}$$

where γ-error factor, L(x, y)–Loss function, min–minimum

The Federated gradient boosting model is represented by

$$F(x) = \frac{\sum_{i=1}^{n} x}{n} \tag{2}$$

where F(x) is the set of predictions of the first iteration as shown in Eq. (2).

The boost function B(x) is obtained by

$B(x)= f(x) + g(x)$ with error factor γ at each step, it can be represented as shown in Eq. (3)

$$B_k(x) = f_{k-1}(x) + \gamma g_k(x) \tag{3}$$

Here, the residual error is calculated as (x-f(x)) to form a regression tree from the previous iteration. It can be represented as g(x). g(x) computes the mean of each decision tree from the previous model. It optimizes the prediction models for malicious content by fitting the decision tree.

4 Proposed Federated BC Model

In general, the collaborative responses from multiple participant nodes or edge devices are collected and aggregated on the BC server. As shown in Fig. 1, the requests to set up intrusion detection learning models are distributed to multiple participant nodes (A, B, C, etc.) in the network from the BC server. The responses or results from the participant nodes are optimized in the BC server by aggregation. The predictions of intrusion or intrusions confirmed by signatures are sent as responses from the participant nodes. Though the BC server is not a central server, it performs the optimization based on the results produced by the participant nodes. The optimized or aggregated responses are observed as global model responses for making further decisions. The participant nodes may be at geographically different locations in the distributed environment, but all of them work on the same dataset signatures and divide problems assigned by the server, according to the divide and conquer principle. In the BC server, FL divides the problem (intrusion detection) and requests participant nodes to work on their own learning models and expects the responses within stipulated times. As a result, the secured and shared prediction models, as the aggregated outcome from the BC server, identify the malicious content in the smart grid network. All the training and signature datasets are stored in the edge devices. The secured environments in participant nodes are achieved by their local data storages. There is no need to exchange the dataset to the BC server; instead, each node shares only its own responses through end-to-end encryption. The predictions are in real-time within the specified time interval. The BC stores all the requests, responses and summarized results as history of transactions using DLT. The BC DLT transaction is immutable because any update or change requires approval from the distributed BC nodes. The actual steps of federated intrusion detection operations are discussed in Algorithm 1.

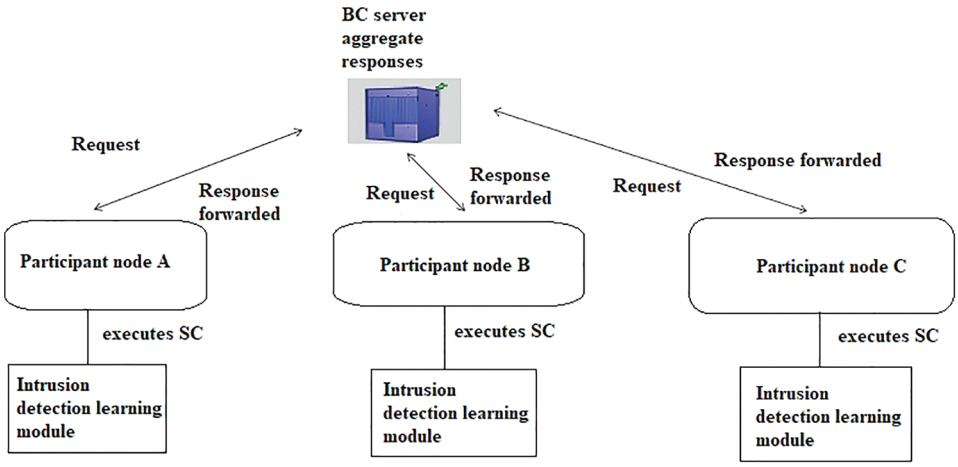

Figure 1: System architecture

Algorithm 1: Federated intrusion detection

INPUT: Intrusion problem P(i), Signature dataset sig(i)

OUTPUT: List of malicious intrusions (M)

1: **for each** round i = 1, 2, 3, …, n **do**
2: c(i) = clients set
3: **for each** client c(i) **do**
4: c(i) = split(P(i))
5: c(i) = train_model(sig(i))
6: **if** match(sig(i)=P(i)) **then**
7: F = F U {sig, P(sig))
8: T = T U {t, P(t)}
9: **end if**
10: client_update(c, i)
11: **done**
12: $M^2 = M^2$ U {F,F(t)}
13: **done**
14: **for** local step j = 1, 2, 3, …, n **do**
15: M = M – Grad{F(P, t) for P
16: return M to server
17: **done**
18: **END**

Notations: P: Problem, Sig: Intrusion Signatures, F: File contains malicious content, T: Time at which malicious content found, M^2: Malicious intrusions include File and Time, Grad: Gradient of intrusion

As described in Algorithm 1, the set of clients (participant nodes) c(i) are selected at random for each round of iteration in steps 1–2 and responses from the clients are sent back to the server in step 10. The intrusion detection problem P(i) is split equally and assigned to each client c(i) in step 4. The intrusion signature dataset sig(i) is used by prediction training models to identify the malicious signature present in the network. The problem P(i) is considered as suspected content and compared with the signature dataset sig(i). If a match is found, the confirmed problem signature P(sig) is recorded into a file F and the time at which the match is found is recorded into a time file T, and these two files are updated to the client node for each iteration in steps 5–9. Malicious intrusion (M^2) is identified through the training dataset and recorded as a signature file F and time T, as mentioned in step 12. The cumulative federated gradient results for problem P are aggregated for all clients by the server and returned to the network administrator in steps 14–16.

4.1 Smart Contract Execution

The SC was established for executing intrusion-learning code in response to the BC server's request. The SC made the execution transparent and provided immediate responses to the server. It also maintained logs using DLT as transactions to store and retrieve all requests/responses. In this study, the FL-XGB modules were written as an SC and executed in participant nodes as shown in Fig. 1. The federated intrusion detection operation shown in Algorithm 1 was fully implemented as an SC in which train_model( ), client_update( ), and the cumulative federated gradient function Grad {F( )} were sub modules of SC, as depicted in Fig. 2.

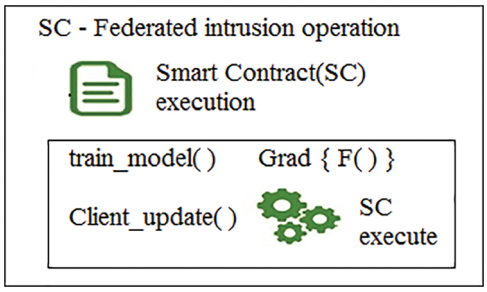

Figure 2: Smart contract operation

4.2 Mathematical Model of Malicious Intrusion Behavior in the Smart Grid Network

Assume that the sample space is comprised of a set of nodes B, M, and N. If B has been identified for its malicious intrusion behaviour, then operations associated with B, for example B1 + B2 are also malicious. When B is malicious, the probability distribution associated with B is P(B).

Lemma 1: A node is malicious when all possible values lie within $0 \leq P(B) \leq 1$. A random experiment to find intrusion on two independent sets can be obtained by a joint distribution function. B_i is the collection of discrete malicious events identified by the system that can be stated as

$$\sum P(B_i) \text{ where } i = 1,\ 2,\ 3,\ \ldots,\ n$$

The mathematical experiment for the sample space is taken as $S = \{a_1, a_2, a_3, \ldots, a_n\}$. The distribution function for the continuous events is $F = \int P(B_i)$ where $i = 1, 2, 3, \ldots, n$. In the distributed environment the likelihood of intrusion occurring can be expressed as shown in Eq. (4)

$$P(B) = \frac{n(B)}{n(S)} \tag{4}$$

where P(B) is the probability distribution of node B, n(B) is the number of malicious nodes, and n(S) is the total number of nodes in the distributed system.

Lemma 2: In a vector space, distributed nodes are both active and malicious. The subsets of malicious nodes from the set of nodes span to a set of all linear combinations of r nodes.

When malicious node (α) exists in a vector space that can be defined as $N = (n1, n2, n3, \ldots, n_p)$, the set becomes as shown in Eq. (5)

$$\alpha N = \alpha(n1,\ n2,\ n3,\ \ldots,\ n_p) = (\alpha n1,\ \alpha n2,\ \alpha n3,\ \ldots,\ \alpha n_p) \tag{5}$$

where N is the set of distributed nodes (n1, n2, n3, …, n_p), and α is a malicious node in the network.

Therefore, the malicious node can exploit and obtain the authorization of each node in the network. While rectifying the negative effect of malicious nodes, the network shall switch back to normal mode. This can be stated as,

$$N + \boldsymbol{\alpha} + (-\boldsymbol{\alpha}) = N \tag{6}$$

where N is a set of active nodes in the network, $\boldsymbol{\alpha}$ is a malicious node, and $(-\boldsymbol{\alpha})$ is the deactivation of the malicious node.

When a network contains two malicious nodes $(\alpha,\ \beta)$, the intrusion of these nodes can affect all active nodes in the distributed network. Eq. (7) illustrates the distributed nature of mathematics.

$$(\alpha + \beta)n_p = \alpha n_p + \beta n_p \tag{7}$$

In this case, n_p is a set of active nodes, and $(\alpha,\ \beta)$ are malicious nodes in the network.

Eq. (6) is a closure property of a malicious node. While a closure property exists in a network, the nodes in the network N are present in the complex network space. The trivial intrusion vector space B = [p] is the property required to function and perform transactions in the network. If f(x) is the function of a valid or legitimate node in the network g(x) is the function of an invalid or malicious node in the network and they are in the same distributed vector space network, then, working on the specified time interval $[t_1, t_2]$ such that the time is a relatively non-zero constant ρ, while f(x) and g(x) are a part of same BC network, then it can be defined as $f(t_1) = g(t_2) = \rho$ since $f(t) + g(t) = 2\ \rho$, whereas while f(t) + g(t) is not present in the same network, $\rho = 0$.

5 Results and Discussion

5.1 Implementation

Tensorflow was developed by Google to support FL and machine learning computations. Keras a python library, is used to define and train learning models. The Python distribution Anaconda was used to execute program code and the CUDA Toolkit 10.0 by NVIDIA was used as the API to work with python to accelerate the compilation process. Ethereum BC was installed to create BC user accounts, and Solidity was used to write SC to be executed on top of Ethereum BC. Ganache-CLI was used as the command line interface to execute all types of Ethereum BC transactions and the SC was deployed on Ethereum to implement the FL algorithm for intrusion detection in the smart grid network.

5.2 Network Scanning

The network scanning tool Network Mapper (NMAP) was used to scan network ports, internet protocol (IP) addresses and applications running inside the network. NMAP finds and reports vulnerabilities in network devices and applications. NMAP detects malware applications, hijacking vulnerabilities, blocks trojan execution within the smart grid network, and safeguards smart meters from bot injection attacks. It scans and protects the web servers, network applications, antivirus systems, domain name system (DNS) servers, and operating systems through its timely reports to the administrator.

5.3 KDD Cup' 99

The KDD Cup' 99 dataset was used for the intrusion detection operation in our experiment. The dataset has many attributes with associated features. The important attributes contained in KDD are transmission control protocol (TCP) connections, traffic, attack vectors, etc. [21]. The KDD dataset contains 4898431 records out of which 93% are attack signatures. The proposed federated prediction models were applied on this dataset to analyze the accuracy, prediction time, etc. The following Table 2 shows the experimental support provided by the KDD dataset, the sample list of cyber intrusion classes, and their corresponding cyber-attacks.

Table 2: Attack vectors provided by the KDD dataset

Class	Sample attacks
Denial of Service (DoS)	UDP, Storm, Worm, Back, Smurf
Remote to Local (R2L)	Guess_Password, Spy, Imap, Xlock
Probe	Satan, Saint, Mscan
User to Root (U2R)	Buffer overflow, LoadModule, PS, Xterm

When U2R attacks are considered, the major cyber-attacks such as buffer overflow, LoadModule, PS attack, and Xterm can be identified by the KDD dataset. A simple definition of these attacks is as follows. The buffer overflow attack copies large volumes of data into program buffers causing system overflow. The loadmodule attack dynamically loads multiple kernel modules for execution which kills the system. The PS attack executes race conditions in programs, putting the system in one or more infinite loops. The Xterm attack enables the attacker to execute commands in the server, providing complete control of the system to the attacker.

5.4 Experimental Study

The problem P was divided into three parts that were assigned to Participants A, B, and C, and the results were sent back to the server for aggregation. Tables 3.1 and 3.2 list the results from Participant nodes A, B, and C. The extracted number of features and accuracy have been listed for the given record set (A–D) containing between 1000 and 8000 records. The tables also list the aggregated or federated results of the participant nodes (clients). It is clear from Table 3.2 that the accuracy (%) achieved by the federated results is higher than that achieved by the results produced by Participant nodes A, B, and C.

5.5 Block Storage and Retrieval in BC

The BC contains a chain of blocks (Block 1, Block 2, Block 3, …, Block n) as shown in Fig. 3. A block stores information such as energy information that includes source and units, transaction information, that includes trades, and timestamp information that conveys the time at which the block was created or added

to the chain. Block hashes start at 0000 for Block 1, and the remaining blocks store the hash of the previous block to increase the difficulty of opening the block for an attacker/intruder.

Table 3.1: Learning results of participant nodes A, B

Record set	Features	Participant node-A accuracy (%)	Participant node-B accuracy (%)
A–1000	20	96.20	95.50
B–3000	30	95.30	96.40
C–5000	35	97.10	97.90
D–8000	40	94.30	95.50

Table 3.2: Learning results of participant node C and server

Record set	Features	Participant node-C accuracy (%)	Server aggregated accuracy (%)
A–1000	20	97.70	98.50
B–3000	30	95.30	97.50
C–5000	35	98.20	99.00
D–8000	40	96.70	98.80

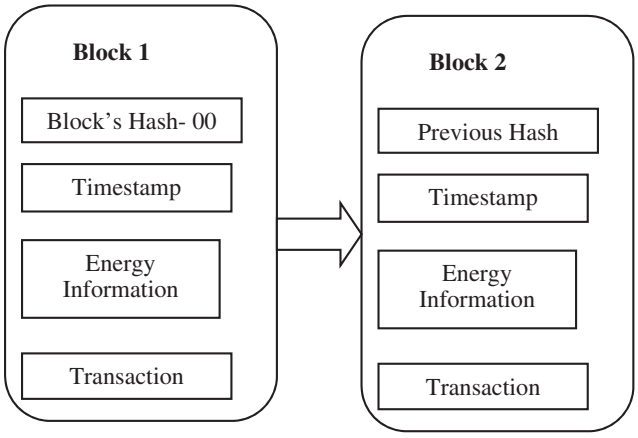

Figure 3: Sample BC model

The BC stores and retrieves energy information from the blocks. The blocks residing in the BC are secured by its own secure hash algorithm (SHA-256) cryptographic hashing function. The transactions such as retrieval and update are protected by its digital signature operation. Table 4 shows the storage time and retrieval time of the respective number of blocks. The average time required to store 10 blocks was 180 s, and the average time required to retrieve 10 blocks was 2.16 s according to the results obtained from the simulation study.

Block storage time and retrieval time analyses were performed with respect to block size (number of blocks) and time in seconds, as shown in Figs. 4a and 4b. As the block size increased, the storage and retrieval times also increased. The average retrieval time of the blocks was lower than their average

storage time. This is because the storage operation involves block creation, transaction and cryptographic operation, whereas the retrieval operation simply verifies and validates the credentials and keys, respectively.

Table 4: Block storage and retrieval time

No. of blocks	Storage time (s)	Retrieval time (s)
1	10	0.3
5	100	1.05
10	180	2.16

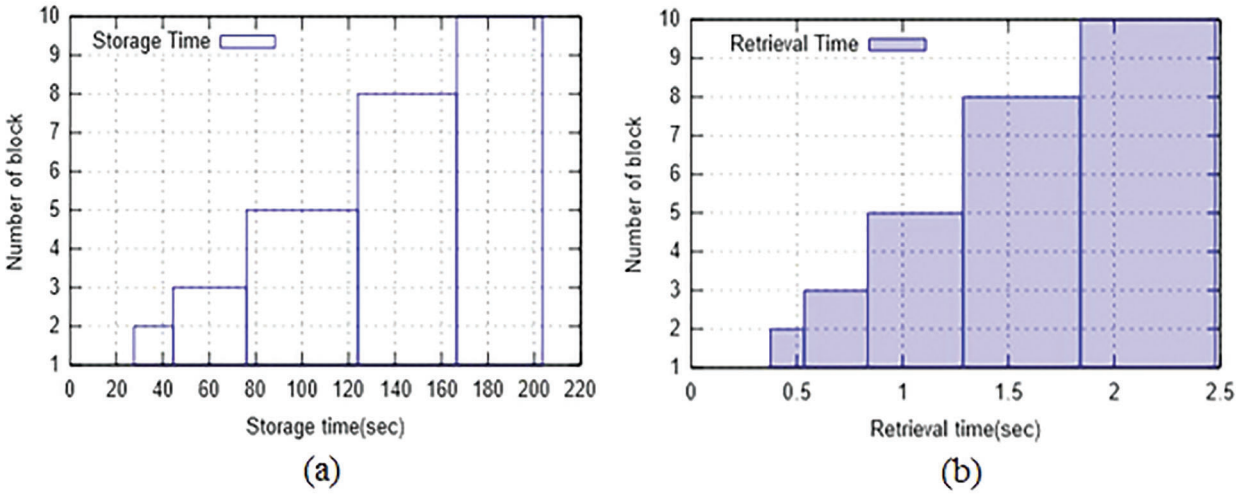

Figure 4: Block time analysis (a) storage time (b) retrieval time

As shown in Table 4, the average storage time for 10 blocks is 180 s and the average retrieval time is 2.16 s. The threshold time was set as per the results listed in Table 5 to perform intrusion analysis in the network. The storage and retrieval threshold times for a block were 18 s and 0.216 s respectively. Table 6 compares the actual time taken and the corresponding threshold time counterpart for varying block sizes of 15–350. Table 6 shows the confirmed intrusion by including time variation between the actual and threshold times. The intrusion was confirmed for the block sizes of 25, 100, 200 and 350 respectively. The actual storage time taken for the block of size 25 was 470.2 s which exceeded the threshold time of 450 s by +20.2 s. Therefore, storage intrusion (S.I) is confirmed. The actual retrieval time taken for the block of size 25 was 6.2 s which exceeded the threshold time of 5.4 s by +0.8 s. Therefore, retrieval intrusion (R.I) is confirmed. The existences of S.I and R.I and the corresponding time differences from the threshold times are stated in Table 6.

The S.I and R.I analyses for the data presented in Table 5 are depicted in Figs. 5a and 5b. The threshold time for each block size is depicted by a point and the actual times taken for storage and retrieval operations for varying block sizes are depicted by lines. It is clear from the graphs for the storage and retrieval operations that the actual time (line) above the threshold time (point) has been identified as the intrusion, as listed in Table 6. Upon considering storage intrusion in the block of size 200 and retrieval intrusion in the block of size 350, the actual time taken (line) is found to lie clearly higher than the threshold time (point). Therefore, intrusion is verified.

Table 5: Block storage and retrieval threshold time analysis

Block size	Storage time (s)	Storage threshold time (s)	Retrieval time (s)	Retrieval threshold time (s)
15	260.4	270	3.20	3.24
25	470.2	450	6.2	5.4
50	880.6	900	9.8	10.8
100	1950.4	1800	23.0	21.6
200	3900.5	3600	45.0	46.8
350	6460.3	6300	79.2	75.6

Table 6: Storage and retrieval intrusion analysis

Block size	Storage time (s)	Storage intrusion (S.I) (s)	Retrieval time (s)	Retrieval intrusion (R.I) (s)
15	260.4	0 (−9.6)	3.20	0 (−0.04)
25	470.2	S.I (+20.2)	6.2	R.I (+0.8)
50	880.6	0 (−19.4)	9.8	0 (−1.0)
100	1950.4	S.I (+150.4)	23.0	R.I (+1.4)
200	3900.5	S.I (+300.5)	45.0	R.I (+1.8)
350	6460.3	S.I (+160.3)	79.2	R.I (+3.6)

The S.I and R.I analyses for the data presented in Table 5 are depicted in Figs. 5a and 5b. The threshold time for each block size is depicted by a point and the actual times taken for storage and retrieval operations for varying block sizes are depicted by lines. It is clear from the graphs for the storage and retrieval operations that the actual time (line) above the threshold time (point) has been identified as the intrusion, as listed in Table 6. Upon considering storage intrusion in the block of size 200 and retrieval intrusion in the block of size 350, the actual time taken (line) is found to lie clearly higher than the threshold time (point). Therefore, intrusion is verified.

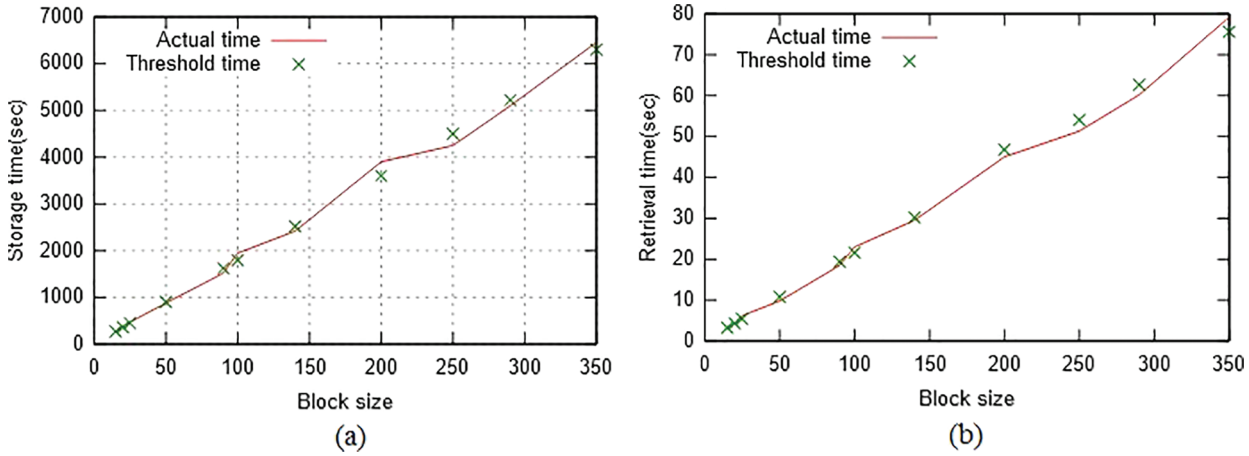

Figure 5: Intrusion analysis (a) storage intrusion (b) retrieval intrusion

6 Result Analysis

A list of the tests conducted and their analysis metrics and results is provided in Table 7. The individual tests and their results are discussed in this section. According to the literature survey, the proposed experimental work and its concept have improved the accuracy and gain. A possible intrusion of 1.2% was found for every 350 blocks of storage and retrieval operations in the application system. Therefore, this study has provided a clear vision of both vulnerability and gain in the smart grid system. Furthermore, the experimental study has provided insight into possible intrusions into the smart energy system. Table 8 compares the results of recent studies associated with smart grids, intrusion detection operations, and BC.

Table 7: Result analysis of proposed work

Test	Analysis metrics	Results
Vulnerability test	940 Ports analyzed	3 Ports are vulnerable
FL-Accuracy test	5000 Record set analyzed	FL-Aggregation accuracy is 99%
Storage gain test	Total 350 blocks are analyzed	Storage gain is 97.5% where actual storage time (6460.3 s) from the threshold time (6300 s)
Retrieval gain test	Total 350 blocks are analyzed	Retrieval gain is 95.4% where actual retrieval time (79.2 s) from the threshold time (75.6 s)
Intrusion test	Total 350 blocks are analyzed	Legitimate action: 98.8% Intrusion action: 1.2% Storage intrusion: 4 out of 350 Retrieval intrusion: 4 out of 350

Table 8: Comparison of results of contemporary studies

Author	Literature study	Proposed system
Shanmugapriya et al. [22]	Deep learning precision and accuracy of intrusion detection are high, but intrusion test is not performed.	FL tested intrusion both at storage and retrieval operations.
Yu et al. [23]	Accuracy of ANN training is 98% Intrusion test on AI based Gray-Wolf optimization Intrusion on data injection: maximum 90% legitimate action.	FL-aggregation accuracy is 99%. Maximum 98.8% accuracy achieved.
Mishra [24]	BC based bifold intrusion detection system obtained storage and retrieval intrusion results for 100 blocks as 200 s and 5 s respectively.	Storage and retrieval intrusions for 100 blocks found at 150.4 s and 1.4 s respectively.
Agung et al. [25]	The processing time for 100 transactions was approximately 12 s.	Required approximately 9 s for 100 transactions.

The proposed system has comparatively given best performance with the current works associated with the same domain and application.

7 Conclusion

The system developed in this study ensures the security of energy data, identifies hostile intrusion, and verifies its functionality in the smart grid network. The IP addresses, network ports, and active applications of the smart grid network were scanned using the network scanning tool NMAP. With the support of the BC server, the proposed federated XGB architecture was able to detect network intrusion with an accuracy of nearly 99%. In our investigation, intrusion detection was conducted using the KDD Cup'99 dataset. The proposed system examined storage and retrieval intrusions by setting a threshold time that validated harmful intrusions into the network. Energy information was built using the Ethereum BC as a series of blocks. In the proposed design, a BC server was utilized to manage all operations, including information block storage, retrieval operations, distribution and aggregation of federated models, history maintenance in ledger, network and stakeholder operations safety. The findings of this experimental study are presented in Sections 5 and 6, together with considerations for all analytical factors, including system security, mathematical modeling, proof of efficiency by gain, and analysis of privacy by authentication.

8 Future Work

In our future studies, key-exchange mechanisms will be used for secure end-to-end communication, and the responses from the edge devices will be further improved. An effective systematic technique can be presented to quickly eliminate the intrusion. When the FL participant nodes respond to and communicate their output results with the aggregate server, strong multi-level encryption/decryption can be used. The size of the network can also be considered while evaluating the key network quality standards, such as throughput and response time.

Acknowledgement: We are grateful to the anonymous referees for their useful suggestions. We would like to thank Editage for English language editing.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] V. D-Gomes, J. F. Martins, C. Lima and P. N. Borza, "Smart grid security issues," in *2015 9th Int. Conf. on Compatibility and Power Electronics (CPE)*, Costa da Caparica, Portugal, pp. 534–538, 2015.

[2] A. O. Otuoze, M. W. Mustafa and R. M. Larik, "Smart grids security challenges: Classification by sources of threats," *Journal of Electrical Systems and Information Technology*, vol. 5, no. 3, pp. 468–483, 2018.

[3] I. Doh, J. Lim and K. Chae, "Secure authentication for structured smart grid system," in *2015 9th Int. Conf. on Innovative Mobile and Internet Services in Ubiquitous Computing*, Santa Catarina, Brazil, pp. 200–204, 2015.

[4] D. Faquir, N. Chouliaras, V. Sofia, K. Olga and L. Maglaras, "Cybersecurity in smart grids, challenges and solutions," *AIMS Electronics and Electrical Engineering*, vol. 5, no. 1, pp. 24–37, 2021.

[5] K. Kimani, V. Oduol and K. Langat, "Cyber security challenges for IoT-based smart grid networks," *Int. J. Crit. Infrastructure Prot.*, vol. 25, no. 4, pp. 36–49, 2019.

[6] A. Ahl, M. Yarime, K. Tanaka and D. Sagawa, "Review of BC-based distributed energy: Implications for institutional development," *Renewable and Sustainable Energy Reviews*, vol. 107, no. 2, pp. 200–211, 2019.

[7] A. S. Musleh, G. Yao and S. M. Muyeen, "BC applications in smart grid–review and frameworks," *IEEE Access*, vol. 7, no. 3, pp. 86746–86757, 2019.

[8] X. Luo, J. Wang, M. Dooner and J. Clarke, "Overview of current development in electrical energy storage technologies and the application potential in power system operation," *Applied Energy*, vol. 13, no. 7, pp. 511–536, 2015.

[9] H. Zhu, H. Zhang and Y. Jin, "From federated learning to federated neural architecture search: A survey," *Complex Intell. Syst*, vol. 7, no. 10, pp. 639–657, 2021.

[10] M. Li, K. Zhang, J. Liu, H. Gong and Z. Zhang, "BC-Based anomaly detection of electricity consumption in smart grids," *Pattern Recognition Letters*, vol. 138, no. 8, pp. 476–482, 2020.

[11] Y. Chen, F. Luo, T. Li, T. Xiang, Z. Liu *et al.,* "A Training-integrity privacy-preserving federated learning scheme with trusted execution environment," *InformationSciences*, vol. 522, no. 2, pp. 69–79, 2020.

[12] N. U. Hassan, C. Yuen and D. Niyato, "BC technologies for smart energy systems: Fundamentals, challenges, and solutions," *IEEE Industrial Electronics Magazine*, vol. 13, no. 4, pp. 106–118, 2019.

[13] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach *et al.,* "BC technology in the energy sector: A systematic review of challenges and opportunities," *Renewable and Sustainable Energy Reviews*, vol. 100, no. 1, pp. 143–174, 2019.

[14] L. Kong, H. Tao, J. Wang, Z. Huang and J. Xiao, "Network coding for federated learning systems," in *ICONIP 2020*, Bangkok, Thailand, pp. 546–557, 2020.

[15] Q. Xia, W. Ye, Z. Tao, J. Wu and Q. Li, "A survey of federated learning for edge computing: Research problems and solutions," *High-Confidence Computing*, vol. 1, no. 1, pp. 1–13, 2021.

[16] V. Smith, C. -K. Chiang, M. Sanjabi and A. S. Talwalkar, "Federated multi-task learning," *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, no. 4, pp. 4424–4434, 2017.

[17] S. M. Kim, T. Lee, S. Kim, L. W. Park and S. Park, "Security issues on smart grid and blockchain-based secure smart energy management system," in *2018 7th Int. Conf. on Power Science and Engineering (ICPSE 2018)*, pp. 260–264, 2018.

[18] H. Li, Y. Cao, S. Li, J. Zhao and Y. Sun, "XGBoost model and its application to personal credit evaluation," *IEEE Intelligent Systems*, vol. 35, no. 11, pp. 52–61, 2020.

[19] R. Shi and X. Xu, "A train arrival delay prediction model using XGBoost and Bayesian optimization," in *2020 IEEE 23rd Int. Conf. on Intelligent Transportation Systems (ITSC)*, Rhodes, Greece, pp. 1–6, 2020.

[20] P. Devan and N. Khare, "An efficient XGBoost–DNN-based classification model for network intrusion detection system," *Neural Comput. & Applic.*, vol. 32, no. 4, pp. 12499–12514, 2020.

[21] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system, association for computing machinery," in *KDD '16: Proc. of the 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, San Francisco, California, pp. 785–794, 2016.

[22] J. Shanmugapriya and K. Baskaran, "Rapid fault analysis by deep learning-based PMU for smart grid system," *Intelligent Automation & Soft Computing*, vol. 35, no. 2, pp. 1581–1594, 2023.

[23] T. Yu, K. Da, Z. Wang, Y. Ling, X. Li *et al.,* "An advanced accurate intrusion detection system for smart grid cyber security based on evolving machine learning," *Front.Energy Res.*, vol. 10, no. 4, pp. 1–13, 2022.

[24] S. Mishra, "BC-Based security in smart grid network," *Int. Journal of Communication Networks and Distributed Systems*, vol. 28, no. 4, pp. 365–388, 2022.

[25] A. A. G. Agung and R. Hindayani, "BC for smart grid," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 3, pp. 666–675, 2022.