Journal of Engineering Science and Technology
Special Issue on ACSAT, October (2021) 120 - 136

BLOCKCHAIN LETTER OF CREDIT: A TRANSACTION-LEVEL ANALYSIS

ASIF BHAT[1,*], RIZAL MOHD NOR[1], MD AMIRUZZAMAN[2]

[1]Kulliyah of Information and Communication Technology, International Islamic University Malaysia, Jalan Gombak, 53100, Selangor, Malaysia
[2]Department of Computer Science, West Chester University of Pennsylvania West Chester, PA 19383, USA
Corresponding Author: Brno University of Defence

Abstract

This study discusses how blockchain technology can be utilized to construct an effective model in trade finance to optimize the process. The study delves into Accepire-BT, a collaboration of trade finance components and blockchain technology. We explore the existing trade finance paradigm and explore how Accepire-BT can help alleviate some of its drawbacks. The events of transactions driven by Accepire-BT are described in detail. The following key characteristics are highlighted: immutability, decentralization, authentication, and data structure. Additionally, we highlight Accepire-BT's benefits and capabilities, including efficiency, transparency, cooperation, and auditability, as well as how they may be accomplished without jeopardizing security, confidentiality, or interoperability. Finally, this study analyzes global trade system's vulnerabilities in terms of trust and security. It proposes a solution using blockchain technology in conjunction with the Standard Letter of Credit as a form of trade finance.

Keywords: Blockchain, Contracts, Distributed, Ethereum, Ledger technology, Smart trade finance.

1. Introduction

Since the introduction of Bitcoin, many have viewed this new cryptocurrency with much skepticism [1-3]. However, lately, the underlying technology – the Blockchain – has had increased interest from the industry [4, 5]. The application of a decentralized ledger is viewed by many as a new source of future innovations. These innovations will arise from using a peer-to-peer transaction-based technology like this because it can cut out the intermediary in various industries. In essence, the Blockchain is a public digital ledger with built-in cryptographic mechanisms to ensure that the ledger cannot be maliciously altered.

Furthermore, the peer-to-peer-based blockchain technology means no central server architecture governing the Blockchain [6]. Instead, it is a decentralized system, where each user of the technology can opt-in to have the entire ledger stored on their computer, and in that way contributing to its decentralization and reliability. One of the crucial parts of the Blockchain is the concept and utilization of smart contracts. Smart contracts are self-organized and autonomous contracts that can self-govern their state [7]. This means that the smart contract can decide to see if a contract is fulfilled and, based on the contract rules, manage payments or digital tokens between parties. All of these are essential parts of blockchain technology and are governed by smart contracts. It is a method of creating a trustless and intricate system that coordinates data between several parties. It is trustless in the sense that there is no necessity in trusting a human entity, but instead, it is the technology that can be trusted. What this offer is that it essentially nullifies any hassle there could be in a multiparty agreement. The first Blockchain was defined by Satoshi Nakamoto in 2008 and used as the primary technology under Bitcoin [8]. This kind of distributed database has since then been the inspiration for others of its kind. One of these blockchain platforms with very high popularity is Ethereum, a generalized blockchain optimized for decentralized applications and not explicitly aimed at virtual currency, such as Bitcoin. Ethereum was first proposed in 2013 and later went live with its first beta version in 2015. This platform offers "Turing-complete" logic that can be used inside of its smart contracts and has the potential of being the template of a new web 3.0.

2. Ethereum's Data Structure

The Ethereum blockchain is a state machine that operates on transactions. A state machine reads a sequence of inputs and then transitions to a new state depending on those inputs. We begin with a "genesis state" in the state machine of Ethereum.

This is equivalent to a clean slate before any network transactions occurring. This genesis state is transformed into a final state when transactions are performed. At any given moment, the current state of Ethereum is represented by this final state, as seen in Fig. 1.

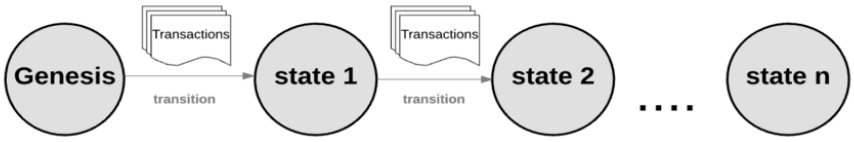

Fig. 1. State transition in Ethereum.

There are millions of transactions in Ethereum's current state referred to as "blocks." Each block consists of a transaction sequence, and each block is linked to the preceding block, as illustrated in Fig 2 only a legitimate transaction shifts from one state to another. A transaction must pass a validation process called as mining to be declared authentic.

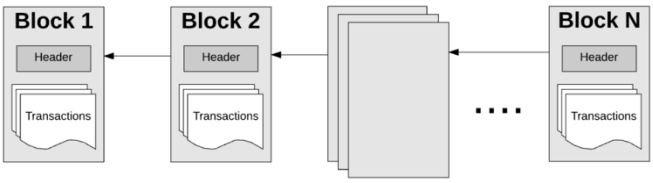

Fig. 2. Blocks and transactions in Ethereum.

Mining occurs if a collection of computers (nodes) pools their computing capacities to construct a block of legitimate transactions. A miner node can attempt to build and verify a block various miner throughout the globe attempt to develop and authorize blocks concurrently. When a miner submits a block to the Blockchain, they give a mathematical "proof" of its validity, proving the block's legitimacy. A miner must verify a block quicker than any other miner to be included on the main Blockchain. The process of authenticating each block through the submission of mathematical proof by a miner is referred to as "proof of work."

A miner receives a certain amount of currency in exchange for their efforts to verify a new block. The Ethereum blockchain operates based on an intrinsic digital coin known as "Ether." Fresh ether tokens are produced and dispersed each time a miner generates a block.

2.1. Accounts

Ethereum's global "shared state" is comprised of numerous tiny objects called accounts that communicate with one another via a message passing mechanism. An account is identified by a state and a twenty-byte address. An address comprises of a 160-bit unique identifier to uniquely identify an account. Accounts are classified into two types:

i. Externally owned accounts: they are secured by private keys and do not include any code.
ii. Contract accounts: they contain associated code and are governed by the contract code.

2.2. Account state

An account state is composed of four components that are common to all accounts:

i. Nonce: If the account is externally owned, it reflects the number of transactions sent from the account's address, but if the account is a contract account, it indicates the number of contracts created by the contract account.
ii. Balance: The amount of Wei that each address possessed. Each Ether is composed of 1e+18 Wei. This statistic indicates the total number of transactions sent from the externally owned account address, whereas the nonce indicates the total number of contracts held by the contract account.

iii. StorageRoot: A hash of the Merkle Patricia tree's root node. This tree initially contains only the hash of the account's storage contents.
iv. CodeHash: An account's Ethereum Virtual Machine (EVM) code's hash. The code is hashed and stored for contract accounts, and it contains the empty string's hash for externally owned accounts.

2.3. World state

The global state of Ethereum is composed of a mapping of account addresses to states. A data structure called a Merkle Patricia tree is used to preserve the mapping. Merkle trees ("Merkle Trie") are a special type of binary tree comprised of:

i. A collection of nodes containing the underlying data, with a substantial number of leaf nodes at the root of the tree.
ii. A set of intermediary nodes, each of which contains a hash of its two child nodes.
iii. A single root node representing the tree's top, generated identical to the hash of its two child nodes.

The following steps generate the data at the root of the tree: i) chunking the intended data, ii) bucketing the chunks, and iii) computing the hash of each bucket. The same practice will continue until just one hash is left: the root hash, as illustrated in Fig. 3.

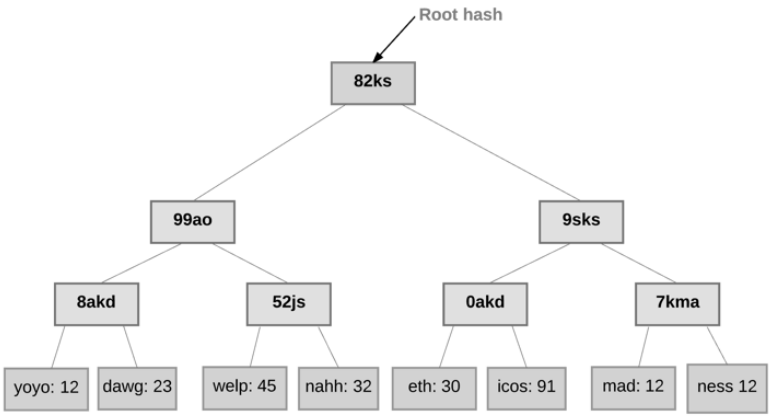

Fig. 3 Merkle trie structure.

This tree has a key for each value. To reach the associated value kept in the leaf nodes, the key should indicate the child node to follow, starting with the tree's root node. For example, the state tree mapping between the related accounts and addresses provides each account's balance, nonce, codeHash, and storageRoot for Ethereum (storageRoot is a tree itself).

This same trie structure is utilized to hold transaction and receipt data. More precisely, each block includes a "header," as seen in Fig. 4, which contains the root node hashes of three distinct Merkle Trie structures, including:

i. State Trie.
ii. Transactions Trie.
iii. Receipts.

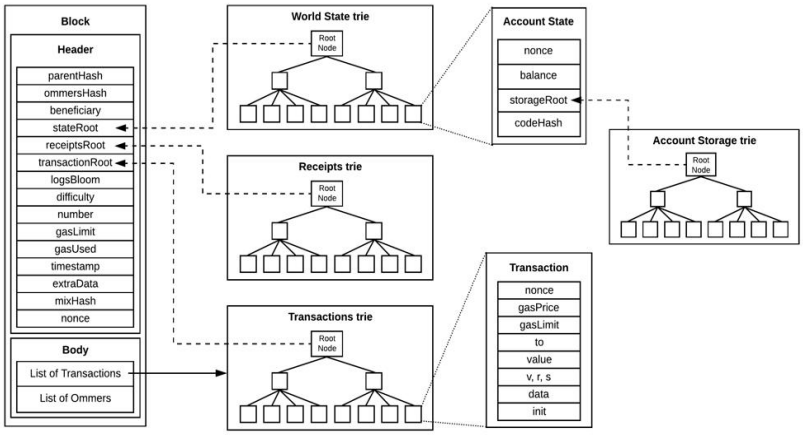

Fig. 4. Ethereum's data structure.

A Merkle Patricia tree is advantageous since the root node depends on the information in the tree, and the root node hash may thus be used for data protection. Additionally, because the block header contains Ethereum state's root hash, receipts trees and transactions, any node can verify a subset of the Ethereum state without keeping the entire state, that might theoretically be infinite in size. A block header consists of the following, as shown in Fig. 4.

i. Parenthash: the parent block's header's hash.
ii. Ommershash: hash of the current block ommer list.
iii. Beneficiary: the rewards for mining a block will be credited to this account address.
iv. Stateroot: the state trie's root node's hash.
v. Transactionsroot: the trie's root node's hash, which contains all block transactions.
vi. Receiptsroot: the hash of the root node of the trie that includes the receipts for all block transactions.
vii. Logsbloom: a global event indicator for the block header.
viii. Difficulty: the degree of block difficulty.
ix. Number: the current block's count.
x. Gaslimit: the present gas quota for each block.
xi. Gasused: the amount of gas consumed in total by block transactions.
xii. Timestamp: the unix block creation timestamp.
xiii. Extradata: further block information.
xiv. Mixhash: a hash used in conjunction with the nonce to establish that the block has performed sufficient calculation.

Notec: it is a hash used in conjunction with the mixHash to verify that the block has performed sufficient computation.

3. Smart Contracts

Smart contracts are self-executing after specific predefined requirements are met and are designed to allow (digital) considerations and artifacts to be exchanged in the real world. In other words, after the conditions agreed by the parties and enforced in the code have been fulfilled, this code will automatically execute those acts without any human intervention necessary or tolerated.

Ethereum uses the Solidity programming language to create smart contracts. Fig. 5 illustrates the process of deploying solidity code to the Blockchain. The solidity source code is converted to EVM bytecode by the solidity compiler "solc." This is referred to as the contract creation code. This acts similarly to a constructor in that it places the contract's bytecode on the Blockchain and can be performed just once by the EVM to place the run-time bytecode on the chain. The byte code is executed by the EVM when the contract is invoked.

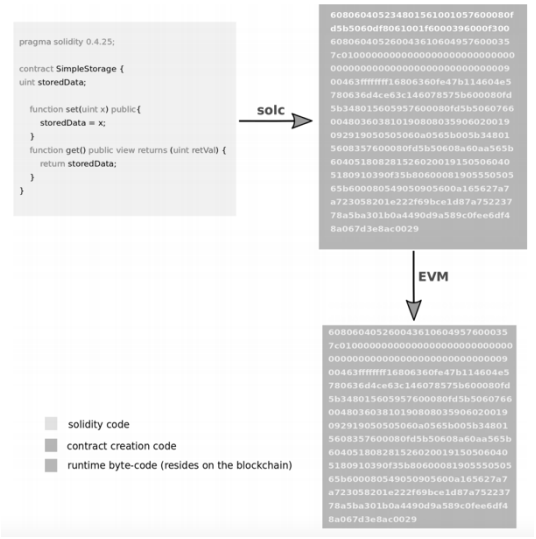

Fig. 5. How Solidity code is put on blockchain.

4. Background

Historically, paper-based trade presented unique problems with forgeries and other security concerns. Banks adopted SWIFT communications and began digitizing paper-based records, but these innovations had little impact on the game. [9]. Security concerns such as document privacy and confidentiality, malicious modification, and trust difficulties persist in contemporary systems [10]. The alternative method of obtaining a secure system is relatively expensive and requires the presence of several third parties. [11] mentions that the laws and regulations governing cross-border commerce, i.e., customs procedures, are extremely intricate, characterized by stringent compliance requirements, corruption, and security violations and breaches. Indeed, trade facilitation is required to streamline this process. The worldwide shipping sector transports 90% of the world's products and continues to rely heavily on paperwork [12].

Blockchain seems to solve these challenges by having an immutable ledger for the storage of digital data and assurances through smart contracts. The use of Blockchain in security is supposed to simplify and increase the efficiency of the sector, making it easier and safer for enterprises of all sizes to trade across borders, thereby contributing to global economic development. Recent progress in Blockchain-based payment systems to be utilized in International Trade Finance is mentioned in [11, 13-16]. Several critical aspects, both from a financial application standpoint and from the perspective of significant regulatory requirements

pertaining to account provisioning for financial asset reporting, are discussed in [17]. Jain and Sedamkar [18] demonstrated that blockchain technology can be used to facilitate asset transfers, Anti-Money Laundering (AML) initiatives and Know Your Customer (KYC) procedures. Du et al. [19] have established supply chain finance, a subset of trade finance, using blockchain technology. While the current financial landscape may not be entirely replaced by technology, its proportional influence maybe both transformative and destructive. The goal of this study is to identify the system's pain points and to offer a solution based on blockchain technology and Standard Letters of Credit as a method of trade finance.

We suggest a technique based on blockchain smart contracts for ensuring the trading ecosystem's trust and security. Juma et al.[20] proposes that the deployment take place on a private blockchain because we aim to construct important business activities like issuing letters of credit. Still, we're implementing Ethereum, a public blockchain platform, because i) it assures the protection and legitimacy of all users, ii) a private blockchain network has fewer nodes or users, a security breach is higher and iii) private blockchains are constrained to the necessity for a central Identity and Access Management (IAM) system to run properly. It has all the administrative and control rights, and it enables the inclusion of a new node to the network or access to the blockchain information. This entire system is against the principle of decentralization, one of the foundations of blockchain technology. Our approach will address most of the pain points we discovered. While dominant platforms such as Corda [11] give nearly identical features, Ethereum also does.

5. Traditional trade finance model

As seen in Fig. 6, the procedure and pain points associated with the traditional trade financing model are as follows:

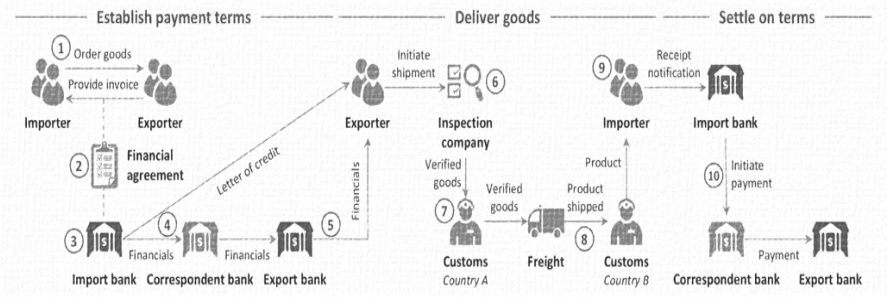

Fig. 6. Traditional trade finance process [21].

1) Process

i. The importer and exporter agree to sell the products on a future date and time.
ii. The financial agreement is represented in an invoice showing the number of goods sold, the rates, and the delivery date.
iii. The importer supplies a copy of the financial arrangement to a bank for inspection.
iv. The import bank checks the financial arrangement and provides the corresponding bank with financial details in its relationship with the export bank on behalf of the importer.

v. The export bank offers financial information to the exporter to start the shipment.
vi. A trusted third-party company inspects the products for invoice alignment.
vii. Local customs officers inspect the goods based on the country code within the exporting country.
viii. The goods are brought from Country A to Country B by cargo, and the local customs agents inspect the goods in compliance with the country code in the importing country.
ix. After inspection, the goods are sent to the importer, who notifies the import bank of receipt.
x. Upon receipt of the notice, the import bank starts payment by the corresponding bank to the export bank. Then, via smart contract, the funds are transferred to the exporter, and the exporter posts the acknowledgment.

2) Pain Points

i. Creating Manual Contracts: The import bank manually checks the importer's financial arrangement and sends the financial details to the corresponding bank.
ii. Invoice Factoring: Invoices are used by exporters for short-term multi-bank funding, adding risk if goods fail to be shipped.
iii. Delayed timeline: The arrival of the goods is delayed by several intermediaries' inspections and various contact points.
iv. Manual AML Review: Export banks must perform AML checks manually using import bank financials.
v. Multiple Channels: Since every group operates on various platforms across nations, misunderstandings are widespread, and there is a high propensity to fraud.
vi. Duplicative Bills of Lading: They are often funded due to banks' failure to check their legitimacy.
vii. Multiple Versions of the Truth: As financial services are sent from one entity to another, there are significant obstacles to version control as changes are made.

Delayed payment: several intermediaries must check that the importer has received goods, as decided before the exporting bank receives money.

6. Accepire-BT

The purpose of this study is to demonstrate that Accepire-BT has the potential to transform the trade finance business. To demonstrate this possibility, we produced a simple system to grasp and utilize for our assessors. Our target audience of assessors includes individuals with experience in trade finance but no prior understanding of smart contracts or blockchain technology. As a result, our solution does not depart much from existing solutions regarding usability and platform for interaction.

Figure 7 depicts an overview of Accepire-BT, a blockchain-based trade finance system. Trade Finance may be defined as the financial transactions between a seller and a buyer in both local and international trade enabled by intermediaries like banks, customs, inspection, and shipping businesses.

A decentralized digital ledger (Ethereum) is utilized to provide concurrent access to information by all stakeholders. When a transaction occurs, data is added concurrently across all nodes, and a new block is created. Each transaction is irreversible and has a unique nonce value trail. Existing data cannot be updated;

new data may only be inserted. This renders Accepire-BT impervious to tampering and secure. In addition, numerous consensus techniques are employed to guarantee that secrecy is maintained and that only trustworthy parties participate in transaction operations. This ensures efficient validation and verification even without a centralized authority.

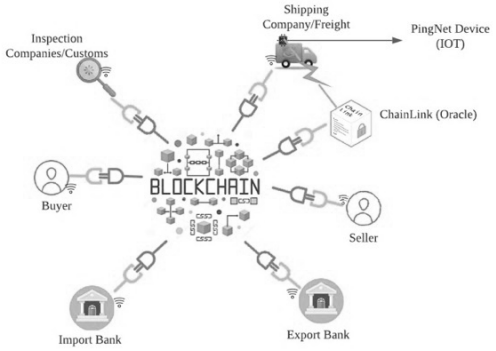

Fig. 7. Overview of Accepire-BT trade platform.

Smart Contracts guarantee that regulatory requirements and transaction terms are satisfied. These contracts contain the rules, and transactions are completed only once all contract criteria and terms have been satisfied.

The application of smart contracts in Accepire-BT is illustrated in Fig. 7. Smart contracts operate as an intermediary between the Blockchain and its users. A smart contract will facilitate all transactions created in the prototype. This avoids centralized participation and maintains the notion of a facilitating trade finance business model compared to existing alternatives.

Figure 8 accepire-BT enables organizations to function more efficiently by serving as a one-stop shop for all financial processes that can be trusted, owing to the smart contract capabilities and transparent distributed ledger. Individuals may interact and examine data in real time. Additionally, this assists in mitigating risk and eliminating wait times.

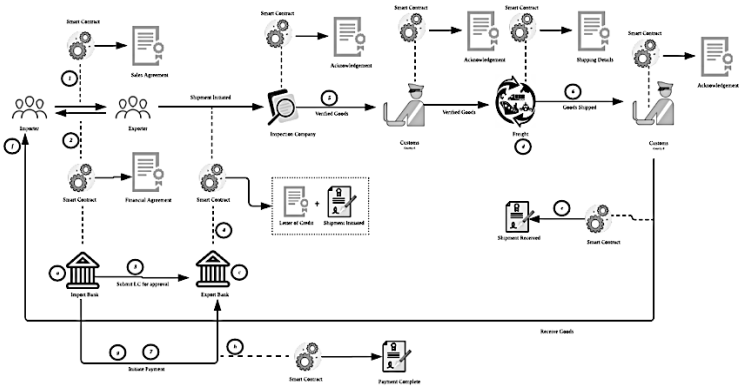

Fig. 8. Overview of the trade finance process in accepire-BT.

1) Process

i. The importer and exporter collaborate to create a basic agreement that satisfies both parties' needs. If they agree on the transaction conditions, a formal agreement is established and communicated through smart contract with the importing bank.
ii. Through a smart contract, the importer bank will authorize and communicate with the exporter bank on the purchase deal. In addition, the importer bank is authorized to establish accountability conditions, evaluate purchase papers, and submit responsibilities payable to the exporter bank.
iii. The draft agreement is reviewed and validated by the exporting bank upon receipt. The smart contracts are then verified. These contracts include all information chains associated with trading. After implementing a smart contract, the data stored within it becomes immutable. As a result, the order and payment responsibilities are irreversibly locked in.
iv. The exporter will digitally sign a smart contract-generated letter of credit upon receipt of the smart contract. This initiates the transaction.
v. The exporter makes available the products ordered by the importer for shipment. Inspections are conducted before to shipping by third-party vendors and customs agents in the exporter's country.
vi. These entities sign the Blockchain's smart contract after being inspected. After then, the importer receives the products.
vii. Upon receipt of the goods, the importer acknowledges the receipt and initiates the payment procedure.
viii. Payments are made to the exporter using smart contracts, and the exporter posts an acknowledgement.

2) Benefits

i. Real-Time Review: Financial records connected and available via accepire-BT are checked and accepted in real-time, minimizing shipping time.
ii. Transparent Factoring: Invoices generated by Accepire-BT provide a real-time, transparent picture of subsequent short-term funding.
iii. Disintermediation: Banks that facilitate trade using Accepire-BT do not require a reputable broker to assume the risk, therefore eliminating the requirement for correspondent banks between import and export banks.
iv. Reduced counterparty risk: Accepire-BT monitors bills of lading, preventing double-spending capacity.
v. Decentralized contract execution: Real-time status update as contract terms are met, minimizing the time and the headcount needed to track the supply of products.
vi. Evidence of ownership: Accepire-BT maintains complete transparency on the ownership and placement of items.
vii. Automated payment and lowered processing fees: Smart contracts execute contractual conditions without the requirement for corresponding banks or additional transaction expenses.
viii. Regulatory transparency: Authorities have a real-time view of the vital records that sustain compliance and AML operations.

Ping Asset devices are utilized to collect IoT data in Accepire-BT, while ChainLink functions as the Oracle that talks with the smart contracts. Despite the fact that these devices are responsive to physical circumstances like temperature,

shock and movement, events are produced only when numerous additional business-defined factors occur. Among these occurrences are the following: Departure and arrival zones, Geo-fences and locations, Temperature events, Shock events, UV index, Humidity, Radiation, Dwell time, Altitude, Distance, Inventory networks, Business and workflow rules, AI-determined hotspots, and User-definable custom data [22].

6.1. Ping asset

Ping Asset devices transmit data into smart contracts automatically, where IoT event data may be used to execute code and initiate numerous operations, including payment for products and services [22].

6.2. Chain link

ChainLink is the most extensively used and safe technology for executing universal smart contracts. ChainLink, administered by a decentralized global community of hundreds of thousands of individuals, is providing a more equitable style of contracting. Its network presently secures billion-dollar worth of smart contracts in many ecosystems, including gaming, insurance and Decentralized Finance (DeFi). In addition, hundreds of companies rely on ChainLink to give indisputable truth and provide continuous, trustworthy data flows. By integrating ChainLink and Ping, the IoT-focused network will gain a variety of functionalities [23].

7. Results and Discussion

We can resolve the fundamental concerns indicated in Section 5 with Accepire-BT. We've built a smart contract as the sole means through which all users may access the blockchain ledger. After deploying the smart contract, all peers join the network and gain access to the methods (Calls and Transactions) specified in Table 1. The addresses of the parties in the transaction are listed in Table 2.and Fig. 9 summarizes the output of the functions utilized in Accepire-BT. When a peer proposes a transaction, it is verified by other network peers, and after all nodes respond positively, committers add the transaction to the ledger. Additionally, information cannot be changed or deleted by any authority once it is recorded owing to the immutability of the ledger, which establishes trust.

Malevolent actors cannot change the prior transaction. This results in change to the hash of the previous block's field in the following block, resulting in inconsistency and therefore being ignored by the network. Additionally, since data is distributed among all peers, the likelihood of a malicious change in one place influencing the ledgers of all peers is nil. Therefore, the risk of data manipulation in centralized systems is decreased, while ensuring data integrity.

Peers must sign the transaction using their private keys, which enables others to determine which peer initiated the transaction, thus ensuring the transaction's nonrepudiation. Smart contracts expedite the administrative process by automating payment and delivery of goods. Eliminating manual tasks via process atomization saves money and minimizes human error, thus increasing efficiency and reducing delays. Because the system is event-driven when a contract's condition is met, an event is sent, and the job specified is done. Each peer has perpetual access to the ledger, which enables regulators and compliance to real time auditing. Due to the

absence of inspection and paper document exchange, the time between shipping and payment is significantly reduced.

Table 1. Transactions and Calls/ Queries.

Participants	Transactions	Queries
Buyer	setSalesContract() addOrder() confirmInvoice() cancelOrder() setFinancialAgreementParties() confirmShipment()	orderExists() viewInvoices() viewOrders() checkOrder()
Seller	createInvoice() confirmOrder() cancelOrder() setShippingAgreement() paymentReceived()	orderExists() viewInvoices() viewOrders() checkOrder()
Issuing/Import Bank	confirmFinancialAgreement() setLCAgreement() addDocument() setPaymentAgreement() initiatePayment()	getNumberOfDocuments() getDocumentID() IsDocumentValid()
Corresponding Bank	validateDocument() processpayment()	getNumberOfDocuments() getDocumentID() IsDocumentValid()
Shipping Company	initiateShipment()	
Inspection Company	verifyGoods()	
Buyer's Country Customs	verifyGoods()	
Seller's Country Customs	verifyGoods()	

Table 2. Participants and addresses.

Participant	Address
Buyer	0x227694770889EcE3024A310Ccf8BF58FE91DB723
Seller	0x0086dDd79062ABD1f7DC704019Ae516967fa2Ff6
Issuing Bank	0x9aC6Ec5Fb03e4663E1EE86A262Dd688F7BD1DEC5
Corresponding Bank	0x27E4534B96D4ECb66E03da16B9F2Ae22820Cc430
Shipping Company	0xe1E6e5182a65306C286F92Ee678e612699d5a8C0
Inspection Company	0x265DA25263054c7b3236Ee5fbC36966963d01fA4
Buyer's Country Customs	0x83fE6DA87Bcd46679E1a64962DAE7E0923773809
Seller's Country Customs	0xc519a1723D8c5646F833F238449C863873088CbB

7.1. Transaction execution

To be executed, all transactions must first satisfy a predefined set of conditions. These include:

i. Properly formatted RLP. The term "RLP" stands for "Recursive Length Prefix" and refers to a data format for encoding binary data in nested arrays. RLP is the serialization format used by Ethereum.

ii. Signed transaction that is valid.

iii. Nonce of valid transaction. The nonce represents the total number of transactions sent from the account. The transaction's nonce must match the sender account's nonce for it to be legitimate.

iv. The transaction's gas limit should be equal to or higher than the transaction's intrinsic gas consumption. The intrinsic gas costs are as follows: a preset transaction cost of 21,000 gas; a transaction gas cost for the data sent (4 gas for each byte of data or code that equals zero; 68 gas for each byte of data or code that is not zero); and an additional 32,000 gas if the transaction creates the contract.

$$i_g=p_c+s_f+c_c \quad (1)$$

where i_g = Intrinsic gas, p_c= Predefined gas (21000), s_f= Storage fee (4(x) 68(y)), c_c = Contract Creation (32000).

v. The sender's balance in Ether must be sufficient to cover the sender's "upfront" gas costs. Calculating the upfront cost of gas is simple: The maximum gas cost is calculated by multiplying the transaction's gas limit by the transaction's gas price. Then this maximum cost increases the overall value sent from the transmitter to the receiver.

$$u_c=g_l*g_p+v \quad (2)$$

where u_c = upfront cost, g_l = gas limit (50000), g_p = gas price (20 gwei), v = value (0.05 Ether).

If the transaction meets all of the previous validity requirements, we go on to the next step. To account for the current transaction, we subtract the transaction's upfront cost from the sender's balance and increase the sender's account's nonce by one. At this stage, the remaining gas may be calculated by subtracting the transaction's total gas limit from the intrinsic gas used.

$$g_r = g_l - i_g \quad (3)$$

where g_r = gas remaining (50000), g_l = gas limit, i_g = intrinsic gas

Following that, the transaction begins to execute. Ethereum maintains track of the "substate" during the transaction's execution. This substate is used to store information gathered throughout the transaction required shortly following its conclusion. It comprises the following:

i. Self-destruct set: The collection of accounts (if any) that is deleted upon completion of the transaction.

ii. Log series: In virtual machine execution, log series are checkpoints archived and indexable.

iii. Following the transaction, the money is to be refunded to the sender's account.

Following that, the transaction's numerous computations are processed. After all important phases of the transaction are finished, the condition is reached by calculating the quantity of excess gas to be reimbursed to the sender, provided that no wrong state occurs. The sender gets together with the unused gas part of the allocation from the "refund balance" previously defined. When the sender is reimbursed, the miner gets the Ether equivalent of the gas spent in the transaction, and the gas is added to the block gas counter. The gas counter keeps track of all transactions that contribute to a block's total gas consumption and aids in block verification. All accounts that were part of the self-destruct set are deleted (if any). Finally, a new state is created along with a collection of the transaction's logs.

7.2. Calls vs. transactions

A call is a local invocation of a contract function without any broadcast or publication on the Blockchain. It is a read-only procedure that uses no Ether.

A transaction is sent to the network, validated by miners, and then published on the Blockchain if it is legitimate. It is a write operation that will influence other accounts, alter the Blockchain's state, and use Ether (unless a miner accepts it with a gas price of zero). Due to the asynchronous nature of the transaction, the immediate return value is always the transaction's hash, as indicated in Fig. 9. A transaction is a series of cryptographically signed instructions provided by a third-party account. They are serialized and published to the Blockchain. Regardless of their nature, they include the following components:

iv. Nonce: a tally of the sender's transactions.

v. Gasprice: the value in Wei that the sender is willing to pay for each unit of gas required to complete the transaction.

vi. Gaslimit: the maximum amount of gas that the sender is willing to pay on the transaction. The sum is fixed and prepaid prior to any computation being performed.

vii. To: the recipient's address.

viii. Value: the value is the amount of Wei transmitted from the sender to the receiver. In a contract-creating operation, this value is utilized as the starting balance for the newly created contract account.

ix. V, R, And S: utilized to produce the signature that identifies the transaction's sender.

x. Init (Exists just for contract creation): A segment of EVM code is needed to initialize the new contract account. init is just executed once and then discarded. When init is invoked for the first time, it returns the body of the account code, which is the code that is permanently connected with the contract account.

xi. Data (A field that is optional and exists only for message calls): the message call's data input parameters. For example, if a domain registration service function is performed by a smart contract, a call may need input data like as the IP address and domain name Fig. 9 shows the output.

Function	BlockHash	Block Number	From	To	Gas Used
setSalesContract()	0x4a8f53d034c7a777447c814254f2da8954573e472952d5b7b27d08d84b646188	2	0x227694770889ece3024a310ccf8bf58fe91db723	0x8541257ce7fc58014195ce6220c73aa8926d47e7	130036
addOrder()	0x83f41eabaee29ee2cd4365601f47880dbe2158408831423802 2e03aa29620002	3	0x227694770889ece3024a310ccf8bf58fe91db723	0x8541257ce7fc58014195ce6220c73aa8926d47e7	177001
createInvoice()	0x3754c662a2f4b17e67c2a8ad5dd6446830afb54a01fe78d618d4e94de0b538f5	4	0x0086ddd79062abd1f7dc704019ae516967fa2ff6	0x8541257ce7fc58014195ce6220c73aa8926d47e7	118864
confirmInvoice()	0x7d64d50ec5bff2e7abc5fe4a13acd666803be8f2b5e490049ac51461dff84069	5	0x227694770889ece3024a310ccf8bf58fe91db723	0x8541257ce7fc58014195ce6220c73aa8926d47e7	43724
confirmOrder()	0x02e43d7b653e27702d075858c95f4df85ec0abcb4e3deec9f7df985508e9fc70	6	0x0086ddd79062abd1f7dc704019ae516967fa2ff6	0x8541257ce7fc58014195ce6220c73aa8926d47e7	47467
setFinancialAgreement()	0x6b502b9e7f321ea0b37ad67170daac3c5e311d7a75f643cbfcb8a6b2531d6183	9	0x227694770889ece3024a310ccf8bf58fe91db723	0x8541257ce7fc58014195ce6220c73aa8926d47e7	131226
confirmFinancialAgreement()	0xe5cfaff65b2db8e86790e23c5b4a88f8d9aea7dd3ef523081a1a2b1616bd357a	10	0x9ac6ec5fb03e4663e1ee86a262dd688f7bd1dec5	0x8541257ce7fc58014195ce6220c73aa8926d47e7	23649
setLCAgreement()	0x1044b807d0bb66f5216b8ec769cf1e1e3a49b5adc0c2699499b6afb06fe313a7	11	0x9ac6ec5fb03e4663e1ee86a262dd688f7bd1dec5	0x8541257ce7fc58014195ce6220c73aa8926d47e7	128983
addDocument()	0x94d2a6d7b576207ca45692aca75236d29c86b6a59559bb43e0af58cab25ce97d	12	0x9ac6ec5fb03e4663e1ee86a262dd688f7bd1dec5	0x8541257ce7fc58014195ce6220c73aa8926d47e7	47274
validateDocument()	0x9de9f403da2ca9147a0841823d0b25369e827b2a587f0bf39037cdac6a76bfee	13	0x27e4534b96d4ecb66e03da16b9f2ae22820cc430	0x8541257ce7fc58014195ce6220c73aa8926d47e7	44871
setShippingAgreement()	0x3feb998cb91ed53b925b32f63937dcf4ed7ad20e7ecd45eb0382f409b53c3b24	14	0x0086ddd79062abd1f7dc704019ae516967fa2ff6	0x8541257ce7fc58014195ce6220c73aa8926d47e7	177178
verifyGoods()	0x031efd20cd6842edff14405ee081f014f09d5c9f5dfec38043783989390040c2	15	0x265da25263054c7b3236ee5fbc36966963d01fa4	0x8541257ce7fc58014195ce6220c73aa8926d47e7	28679
verifyGoods()	0x485a8ec4affc2193a112016a919cd6936e3585a1addff7d73296a99a2d2174b1	16	0xc519a1723d8c5646f833f238449c863873088cbb	0x8541257ce7fc58014195ce6220c73aa8926d47e7	30486
initiateShipment()	0xb35348439c21191f64fd2d2cff39605baac3935df88fb0a0b361c8baaf52516e	17	0xe1e6e5182a65306c286f92ee678e612699d5a8c0	0x8541257ce7fc58014195ce6220c73aa8926d47e7	31285
verifyGoods()	0x725c76c283c3fcf9c892356d7b40d9b3d8130f400790fb3579cd96d1b3a7dd3e	18	0x83fe6da87bcd46679e1a64962dae7e0923773809	0x8541257ce7fc58014195ce6220c73aa8926d47e7	30509
confirmShipment()	0x59d95550a60150cb5f5fdb59ba68856e59bcdff4f650103700313 7265e644293	19	0x227694770889ece3024a310ccf8bf58fe91db723	0x8541257ce7fc58014195ce6220c73aa8926d47e7	29674
setPayment()	0xa9350c7229ed8e2340355ee4392c8bf05233bffdaaae459ae0e3a9dd8e4d74a5	20	0x9ac6ec5fb03e4663e1ee86a262dd688f7bd1dec5	0x8541257ce7fc58014195ce6220c73aa8926d47e7	106746
initiatePayment()	0x269197b8b76c97049c97e218f729655b1bfa7054f741457cfcfb806ceea8c4b7	21	0x9ac6ec5fb03e4663e1ee86a262dd688f7bd1dec5	0x8541257ce7fc58014195ce6220c73aa8926d47e7	30436
processPayment()	0x030697626dc9404278db19fdef2cfcff2adf1b0f72a3aced814053 7b529ad405	22	0x27e4534b96d4ecb66e03da16b9f2ae22820cc430	0x8541257ce7fc58014195ce6220c73aa8926d47e7	29632
paymentRecieved()	0xe06eb082b41c8647fc8485e24c884f220021a45b566a916262a4051533b9945b	23	0x0086ddd79062abd1f7dc704019ae516967fa2ff6	0x8541257ce7fc58014195ce6220c73aa8926d47e7	30463

Fig. 9. Output of Transactions in Accepire-BT.

8. Conclusions

We have shown in this study the potential of blockchain technology to transform the landscape of trade financing. This study also discusses the traditional trade finance framework. A thorough examination of the potential advantages of incorporating blockchain technology into the current trade finance system is conducted. We also examined blockchain characteristics and how they could be used in systems to improve trade efficacy. In the context of Blockchain technology, this study aims to propose a new trade finance framework to enhance financial systems.

It may be inferred that the above-mentioned suggested system can address security concerns raised during a trading scenario. One might claim that the conventional trade process can be expedited and optimized using blockchain technology. Due to the security features inherent in blockchain technology, it can establish trust within a network. It ensures the integrity of the data being transmitted and may assist in monitoring the operation. Numerous regulations and criteria must be adhered to for international trading to be lawful. As a result, blockchain technology has the potential to significantly improve the auditability of international trade and significantly simplify society and economic preservation goals.

By late 2020, almost 12 companies (Minehub, Marcopolo, Contour, Infosys Finacle and, among others) aggressively pursued Blockchain to rewire commerce and trade finance. The World Trade Organization's (WTO) Emmanuelle Ganne noted that the business has made significant progress toward digitization, particularly considering the present epidemic [24]. Numerous marketplaces and trade technology businesses, including Trade Trust, Crowdz, and International Chamber of Commerce Tradeflow, have leveraged blockchain technology to narrow the trade financing gap. Multidisciplinary Digital Publishing Institute [25] included many case studies in their 2019-20 sustainability report demonstrating the

use of letter of credit and other components to link Blockchain and trade finance. The case studies include collaborations between Maersk and IBM on Hyperledger fabric advancements, Barclays and Ornua, the Spanish bank Banco Bilbao Vizcaya Argentaria (BBVA) on Ethereum Blockchain and Wave technology and HSBC on Voltron (with Corda Blockchain) trade transactions with Cargill. The research and development efforts and operations of these companies are excellent examples of the shifting trends and patterns revealing the developing trade finance and blockchain industry environments.

References

1. DeVries, P. (2016). An analysis of cryptocurrency, bitcoin, and the future. *International Journal of Business Management and Commerce*, 1(2), 1-9.
2. Parveen, P.; and Alajmi, A. (2019). An overview of Bitcoin's legal and technical challenges. *Journal of Legal, Ethical and Regulatory Issues*, 22(1), 1-8.
3. Jordi, H.J. (2015). Research and challenges on bitcoin anonymity. *Data Privacy Management, Autonomous Spontaneous Security*, 3-16.
4. Treiblmaier, H. (2019). The impact of the blockchain on the supply chain: a theory-based research framework and a call for action. SSRN *Electronic Journal*, 23(6), 545-559.
5. Miller, D. (2018). Blockchain and the internet of things in the industrial sector. *IT Professional*, 20(3), 15-18.
6. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; and Wang, H. (2017). An overview of blockchain technology: architecture, consensus, and future trends. *IEEE International Congress on Big Data (BigData Congress)*, 557-564.
7. Szabo, N. (1994). Smart Contracts. Retrieved July 5, 2021, from https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.
8. Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, 1-9.
9. BCG, SWIFT. (2019). From security baseline to best practice securing. Retrieved, October 13, 2019, from https://www.swift.com/swift-resource/222951/download?token=I8QGxxQU.
10. Schiffling, S.; Hannibal, C.; Fan, Y.; and Tickle, M. (2020) Coopetition in temporary contexts: examining swift trust and swift distrust in humanitarian operations. *International Journal of Operations and Production Management*, 40 (9).1449-1473.
11. Jessel, B.; and DiCaprio, A. (2018). Can blockchain make trade finance more inclusive? *Journal of Financial Transformation*, 47, 35-50.
12. IBM, Maersk. (2017). The paper trail of a shipping container how blockchain will help manage and track the paper trail of tens of millions of shipping containers across the world. Retrieved, October 13, 2017 from, https://www.ibm.com/downloads/cas/VOAPQGWX.
13. Coppola, F. (2019). The fast-changing world of blockchain solutions for trade finance. Retrieved, October 13, 2019, from https://www.americanexpress.com/en-gb/business/trends-and-insights/articles/blockchain-can-potentially-streamline-trade-finance/.

14. Kalra, D. (2019). Overriding FINTECH. In 2019 *International Conference on Digitization* (*ICD*), 254-259.
15. Patel, D.; Ganne, E. (2019). Blockchain & dlt in trade: a reality check. Retrieved, October 13, 2021, from https://woa.community/l/library/download/urn:uuid:591ff6e1-1b01-4a44-bae9-5fa3f4e583d6/blockchain-dlt-in-trade_a-reality-check_nov_19.pdf?format=save_to_disk&ext=.pdf
16. McNeill, W.; and Duran, O.S. (2019). Market Guide for Global Trade Management Software. *Gartner.* Retrieved, May 26, 2021, from https://www.gartner.com/en/documents/3947339/market-guide-for-global-trade-management-software.
17. Peters, G.W.; and Panayi, E. (2016). Understanding modern banking ledgers through blockchain technologies: *Future of transaction processing and smart contracts on the internet of money. New Economic Windows*, 239-278.
18. Jain, N.; and Sedamkar, R.R. (2020). A Blockchain technology approach for the security and trust in trade finance. *In 2020 14th International Conference on Innovations in Information Technology (IIT)*, 192-197.
19. Du, M.; Chen, Q.; Xiao, J.; Yang, H.; and Ma, X. (2020). Supply chain finance innovation using blockchain. *IEEE Transactions on Engineering Management,* 67(4), 1045-1058.
20. Juma, H.; Shaalan, K.; and Kamel, I. (2019). A survey on using blockchain in trade supply chain solutions. *IEEE Access*, 7.
21. World Economic Forum. (2016). The future of the financial infrastructure. world economic forum. Retrieved, October 18, 2016, from https://www3. weforum.org/docs/WEF_The_futureof_financial_infrastructure.
22. PING, (2020). Ping to integrate chainlink oracles to provide iot data to smart contracts. Retrieved, December 15, 2020, from https://medium.com/@pingnet/ping-to-integrate-chainlink-oracles-to-provide-iot-data-to-smart-contracts-2fd9d5a1abe2.
23. Hamacher, A. (2020). 29 Projects integrated with ChainLink last month. Retrieved, October 13, 2020, from https://decrypt.co/46918/29-projects-integrated-with-chainlink-last-month.
24. World Trade Organization (2020). Blockchain and DLT in Trade: Where Do We Stard? Retrieved, October 13, 2020, from https://www.wto.org/english/res_e/publications_e/blockchainanddlt_e.htm
25. Paliwal, V.; Chandra, S.; and Sharma, S. (2020). Blockchain technology for sustainable supply chain management: A systematic literature review and a classification framework. *Sustainability*, 12(18), 7638.