frontiers in Blockchain

ORIGINAL RESEARCH
published: 02 August 2021
doi: 10.3389/fbloc.2021.607320

Vind: A Blockchain-Enabled Supply Chain Provenance Framework for Energy Delivery Systems

Eranga Bandara[1], Sachin Shetty[1], Deepak Tosh[2] and Xueping Liang[3]*

[1]VMASC, Old Dominion University, Norfolk, VA, United States, [2]Department of Computer Science, University of Texas at El Paso, El Paso, TX, United States, [3]Department of Information Systems and Supply Chain Management, University of North Carolina at Greensboro, Greensboro, NC, United States

Enterprise-level energy delivery systems (EDSs) depend on different software or hardware vendors to achieve operational efficiency. Critical components of these systems are typically manufactured and integrated by overseas suppliers, which expands the attack surface to adversaries with additional opportunities to infiltrate into EDSs. Due to this reason, the risk management of the EDS supply chain is crucial to ensure that we are knowledgeable about the vulnerabilities in software and hardware components that comprise any critical part, quantifiable risk metrics to assess the severity and exploitability of the attack, and provide remediation solutions that can influence a prioritized mitigation plan. There is a need to realize cyber supply chain risk management for industrial control systems' hardware, software, and computing and networking services associated with bulk electric system (BES) operations. This article proposes a blockchain-based cyber supply chain provenance platform ("Vind") for EDSs to realize data provenance in a cyber supply chain ecosystem.

Keywords: blockchain, smart contract, big data, energy delivery systems, microservice architecture, supply chain, supply chain and risk

OPEN ACCESS

Edited by:
René Hüsler, Lucerne University of Applied Sciences and Arts, Switzerland

Reviewed by:
Remo Pareschi, University of Molise, Italy
Andrea Vitaletti, Sapienza University of Rome, Italy

****Correspondence:***
Eranga Bandara
cmedawer@odu.edu

Specialty section:
This article was submitted to Non-Financial Blockchain, a section of the journal Frontiers in Blockchain

Received: *16 September 2020*
Accepted: *23 June 2021*
Published: *02 August 2021*

Citation:
Bandara E, Shetty S, Tosh D and Liang X (2021) Vind: A Blockchain-Enabled Supply Chain Provenance Framework for Energy Delivery Systems. Front. Blockchain 4:607320. doi: 10.3389/fbloc.2021.607320

1 INTRODUCTION

To achieve reliable and efficient energy delivery system (EDS) operations, utility companies typically adopt various software/hardware products and solutions developed by third-party vendors. The enterprises have realized significant advantages due to such outsourcing activities in terms of continuous support services, just-in-time deliveries, elimination of inventories, globalization, and reduced operational cost. Although these proprietary commercial-off-the-shelf solutions offer high interoperability characteristics with a large variety of product features which the consumer enterprise may not need, integration of such products and services in the critical EDS infrastructures significantly increases the risk of supply chain–related threats, which could adversely impact the reliability of bulk electric systems (BESs). In the past, several attempts (Ellison and Woody, 2010; Du et al., 2013; CSRC, 2018) have shown how adversaries use strategic attack campaigns for extortion, malicious data alterations, and exfiltrations. Attack *via* a manufacturer source code or product and attack *via* vendor remote access are two such attack scenarios (Liang et al., 2018). The software supply chain attacks are becoming prevalent and disrupting the inherent trust between the software provider and the consumer. Thus, the involvement of multitude vendors, suppliers, distributors, and integrators in EDSs must be well handled, and the supply chain risks introduced in the BES should be mitigated through cyber-secure approaches.

There is a need to manage the third-party and supply chain risks for BESs. The plan will involve processes and actions for the utility companies, vendors, and suppliers to strengthen the supply chain risk management. However, the successful realization of this standard hinges on the users, owners, and operators of BESs and the third-party software/hardware vendors as well as suppliers who are engaged in implementing various processes related to BESs. Since these responsible entities must develop one or more documented risk management plan by identifying and assessing the cyber risks involved in BESs, it is important to build a collaborative security ecosystem where multiple stakeholders can seamlessly handle the vendor-identified incidents through effective notification, coordination, disclosure, and validation mechanisms. In this way, the enterprises will have increased visibility into the methods, applications, and services to ensure integrity and authenticity of all software provided by the vendors, thus providing a viable way to manage the evolving cyber risks on the BES.

Typically, current state-of-the-art provenance systems can be deployed through logging and auditing technologies to manage the cyber supply chain risk (Ivanov, 2018). However, these technologies are not effective in the cyber supply chain infrastructure, which are complex in nature, due to several layers of interoperating suppliers and vendors across geographical and organizational boundaries. To identify the origin, cause, and impact of violations in the cyber supply chain infrastructures, the collection of forensics and logs from diverse and disparate sources is required, which is an insurmountable task. At the same time, logs only provide a sequential history of actions related to every application.

We address the challenge of providing a resilient mechanism to monitor and audit the processes involved in the vendor's software and hardware components of BESs. There is an increased responsibility on the vendors to adopt software integrity criteria and controls. The challenges for meeting supply chain risk management requirements (Goff et al., 2014) (Contract, 2020) involve a) timely notification, coordination, and disclosure of vendor-identified incidents; b) software integrity and authenticity; c) vendor remote access; and d) information system planning. Realizing these functionalities is critical for BESs, which would allow devising quantifiable risk metrics to assess the severity and exploitability of BESs and develop prioritized risk remediation solutions. While it is of utmost importance to develop a framework of cyber-secure supply chain management, the multi-ownership environment introduces additional challenges to establish inherent trust among the involved entities. Provisioning transparency and provenance-tracking services are therefore critical in order to build trust and automate the supply chain risk management process. The recent emergence of Blockchain technology has shown its true potential in offering a tamper-resistant distributed ledger platform that can be used to maintain data provenance in an effective and secure manner. However, the practical deployment of Blockchain for better management of the supply chain risks in BESs has not been well studied. Further, the common issues existing in the cloud-based data storage, such as the lack of data privacy, lack of data immutability, lack of traceability, and lack of data provenance, are still hindering around in the integration process of Blockchain technology.

Therefore, in this article, we propose a Blockchain-based cyber supply chain provenance system ("Vind") for EDSs. At a macro level, the approach followed to build Vind involves 1) the customer outlining the performance and security requirements, 2) the vendor identifying the appropriate hardware and software suppliers that will meet the supply chain provenance requirements, and 3) the suppliers ensuring that they report the desired information that allows the customers to improve auditability, attribution, and provenance of their critical assets. The system also involves the integration of vulnerability databases that would allow reporting to the stakeholders about potential vulnerability risks associated with any component, and the supplier will be notified to provide remediation plans.

The Vind platform is built on top of Rahasak blockchain, which is highly scalable in terms of storage capability (Bandara et al., 2018; 2021). This blockchain can be used to share the asset supply chain information between multiple parties in EDSs. All the physical/software asset information in EDSs and their supply chain management information are stored in the blockchain. The supply chain management functions are implemented as blockchain smart contracts in the Vind platform (Bandara et al., 2019; 2020). In this article, a performance evaluation of the underlying blockchain storage in the Vind platform is presented. The evaluation shows the scalability and transaction throughput features in the Vind blockchain system. Following are the main contributions from Vind.

1. A blockchain-based cyber supply chain provenance platform is proposed to address the requirements of the supply chain risk management.
2. Permissioned blockchain has been used to store all the physical/software asset information in EDSs and their supply chain provenance information.
3. Blockchain-based audit storage has been introduced to store the audit logs of software/hardware assets and entities in the Vind platform.
4. To address the privacy concerns in the blockchain, off-chain storage has been integrated into the blockchain.

The rest of the article is organized as follows. **Sections 2** and **3** discuss the architecture and implementation details of the Vind platform. **Section 4** presents the performance evaluation of Vind. Related works are discussed in **Section 5**. Finally, **Section 6** concludes the article.

2 VIND ARCHITECTURE

2.1 Overview

Figure 1 depicts the architecture of the Vind platform. Vendors, companies, and service providers and administrators are the main stakeholders in the Vind platform. The service provider is the host of the Vind platform. The service provider's main function is to on-board vendors and companies into the platform without the central authority. Besides, the service providers will offer services to various system entities but are not trusted from the architecture perspective. All the registration information will

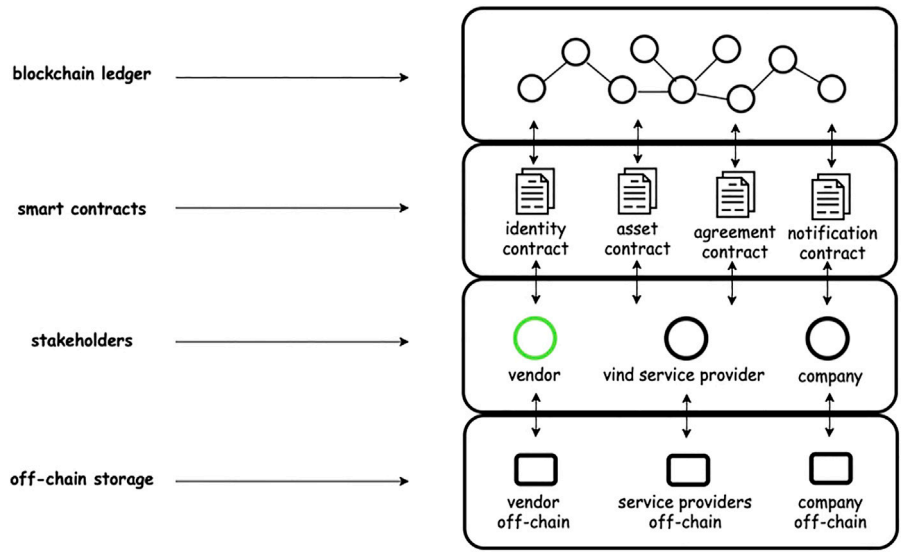

FIGURE 1 | Overview of the Vind architecture. Smart contract defines the application functions in the Vind platform. Blockchain clients (stakeholders) interact with smart contract functions.

be verified by the blockchain node for authenticity and future validation. The service provider cannot modify the registration information on the blockchain ledger, which is immutable and tamper-resistant. Vendors produce hardware/software products, and companies purchase these software/hardware products from the vendors, which can be identified as assets. Each asset in the Vind platform is represented with a unique digital identity (e.g., UUID). Blockchain provides a tamper-evident, shared digital ledger that records data in a public or private peer-to-peer network. Blockchain can be used to share the asset supply chain information between multiple parties in EDSs. All the physical/software asset information in EDSs and their supply chain provenance information are stored in the blockchain. All the blockchain functions of the Vind platform are implemented with smart contracts. More information about these smart contracts is discussed in Section 2.3. Companies, vendors, and service providers are the blockchain clients of the application. They will provide web/mobile applications to interact with blockchain functions. Each blockchain client has its own off-chain storage. Off-chain storage can store confidential data in blockchain peers. The hash of the data stored in off-chain storage can be published in the blockchain. In this way, Vind has addressed the common issues in cloud-based data storage, lack of data privacy, lack of data immutability, lack of traceability, and lack of data provenance.

2.2 Functionality

Vind platform facilitates the following features related to vendor/company/asset life cycle management and supply chain management. There are separate smart contracts to handle each of these functions in the Vind blockchain, as shown in **Figure 1**. There are three main stakeholders in the Vind platform: vendors, companies, and service provider/administrator. An identity contract handles the identity registrations of each of these entities. We have used a self-sovereign–based approach when registering the identities in the Vind platform (Mühle et al., 2018; Baars, 2016). Once identity is registered, the entity will be onboard into the Vind platform. Basically, stakeholder's life cycle management of the Vind platform will be handled by the Identity contract. Service providers who are the maintainer of the Vind platform have the privilege to manage the life cycle of the vendors and companies (basically edit, delete vendor/company profiles). The software/hardware products in the EDS will be supplied by the vendors. Companies (e.g., energy company/power plant) purchase the products from vendors (purchased software/hardware products identified as assets in the energy company). When new assets are purchased in an energy company, their information needs to be saved in the blockchain. This asset life cycle management is handled by Asset contract 13. Theasset contract supports functionality to create and search assets in the blockchain. Once an asset is onboarded into the energy company, there is an agreement formed between the vendor and the company. This agreement defines the supply chain risk management agreement terms (Goff et al., 2014; Contract, 2020). This agreement negotiation and the establishment is handled by the Agreement contract 14.

Various notifications (which relate to assets in the energy companies) will be generated and sent to a responsible person in the energy company and other organizations. All these notification generation and sending functionalities will be handled by the Notification contract 15. There are four main notification scenarios in the Vind platform: a) notification about vendor-identified attack incidents related to the products or services (coordination of responses to vendor-identified incidents related to the products or services), b) notification about vendor-identified issues related to the products or services, c) notification by vendors when remote or onsite access should no longer be granted to vendor representatives, and d) disclosure by vendors of known

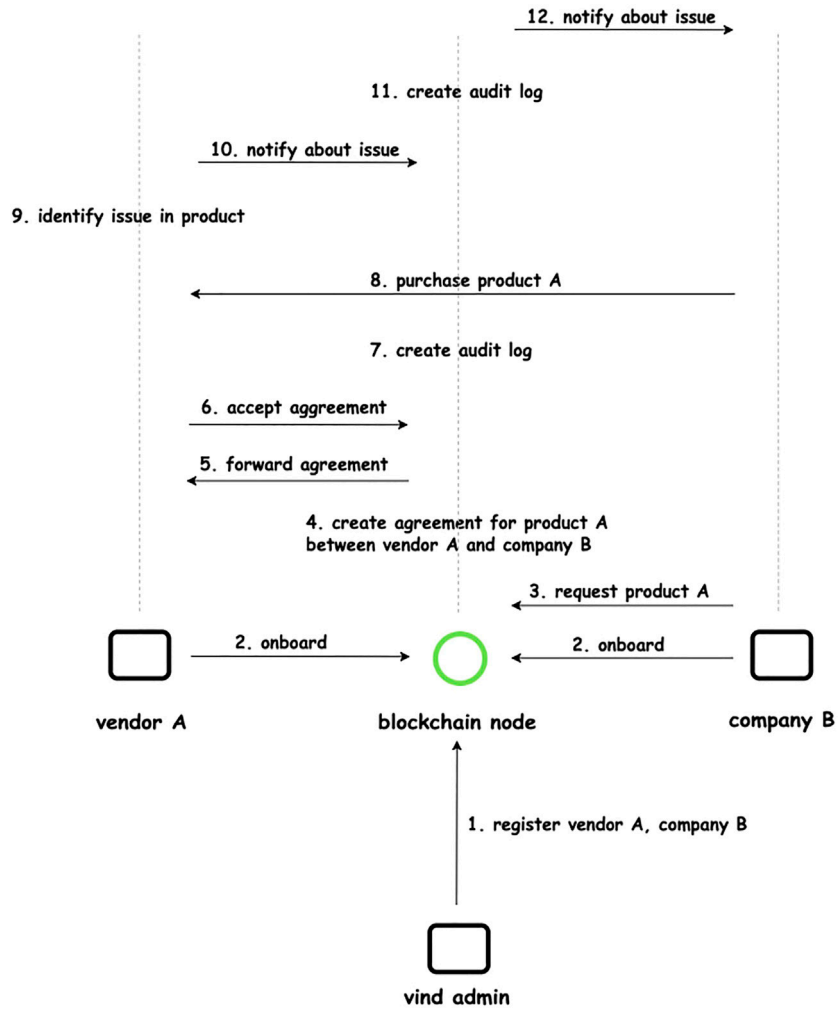

FIGURE 2 | Interaction diagram of a use case in Vind. Vind admin onboard vendors and companies into the platform. When a company purchases a product from the vendor, agreement will be formed. The notifications regarding the products will be generated by vendors.

vulnerabilities related to the products or services. We have integrated the automatic attack discovery and notification function of assets in the Vind platform by using the common vulnerability exposure search API (CVE search API) (Pham and Dang, 2018). CVE search API is a service that contains a list of CVE security vulnerabilities of common software products. There is a public CVE database available with all known vulnerabilities of software products. We have incorporated a CVE search API to find known vulnerabilities of software products/assets in the Vind platform and notify them to the corresponding vendors and companies. In the Vind platform, we have run our own CVE search API service. It periodically fetches the vulnerabilities published in the public CVE search API and generates notifications based on the vulnerability.

There are various patches that can be sent for the software products which are used by the energy companies. The integrity of these patches needs to be verified by using a cryptographic method [e.g., SHA256 hashing (Gueron et al., 2011)]. This integrity checking and verification functionality are implemented with the Verification contract. There should be coordination of controls for vendor-initiated interactive remote access and system-to-system remote access with a Vendor. For example, when a third-party support person gets access (login) to troubleshoot a software product at the energy company, their access information should be traced. These access information tracing will be implemented with the Trace contract. The audit log of the assets, notifications, and identities is maintained in the Vind platform's blockchain ledger. This audit log contains all the events that happened (e.g., create, edit, and delete) to the assets, identities, and notifications in the Vind platform. It provides the data provenance of the Vind platform on top of the blockchain ledger.

2.3 Vind Use Case

Consider a scenario where Vendor A, Company B, and the Vind service provider (Vind administrator) want to procure products and services in the Vind platform. The necessary interactions occurring among these entities can be depicted as presented in **Figure 2**. The administrator can log in to the system and manage

the entities in the platform, **Supplementary Appendix Figures 16, 17**. In this scenario, first, the administrator will onboard Company A and Vendor A into the Vind platform. When onboarding, it registers basic information of the entities such as email, name, and other information in the blockchain, **Supplementary Appendix Figures 18, 19**. These onboarding functions of the blockchain are implemented in the Identity smart contract. Once onboarded, vendors and companies can register into the platform by using the Vind mobile app/Web application with the given e-mail address as the user identifier (e.g., username). Once logged-in, vendors can view the products they have and the companies using those products from the portal, **Supplementary Appendix Figures 20–22**. Vendors need to onboard their products into the platform *via* the Web application, **Supplementary Appendix Figure 23**. Companies can view the purchased products (e.g., assets) and their information from the company portal, **Supplementary Appendix Figures 24, 25**. These product and asset life cycle management functions are implemented with Asset contracts in the blockchain.

When a company is purchasing a product from a vendor, that product can be onboarded to the company through the company web portal, **Supplementary Appendix Figure 26**. Assume Company A is going to buy the product of Vendor A. When buying a product (asset), the company needs to specify the asset information and cyber supply chain agreement terms, **Supplementary Appendix Figure 27**. At this time, the vendor and company execute a supply chain agreement contract. The agreement contract in Vind will implement the functionalities needed to automate the supply chain risk management requirements and notify Vendor A. Then, Vendor A fetches the agreement terms from the blockchain and decides whether to accept or reject these terms, **Supplementary Appendix Figure 28**. If vendors accept the terms, Product A will be linked to Company A in the Vind platform as an asset.

If, for instance, the vendor identifies an issue in Product A, it must follow the agreement terms and is required to notify the issue to Company A. This functionality is integrated with Vind to enable the seamless release of software patches, notification of vulnerabilities, and software updates for the products to the companies. In order to ensure the integrity of the patches and updates, they will be signed by the PGP key of the vendors. Company users can enter the serial # of the patch and check the integrity. Vind will allow the release of patches and updates to the companies only when the integrity is verified. In each of the above steps, audit logs will be created and stored in the blockchain, **Supplementary Appendix Figures 29, 30**.

3 VIND IMPLEMENTATION

We have built the production version of the Vind platform to support high scalability and high transaction load. To cope with the high transaction load and back pressure (Destounis et al., 2016) operations, we have adopted a reactive stream-based approach using Akka streams (Davis, 2019). All the services in the Vind platform are implemented as small services (micro-services) with the single responsibility principle. These services are dockerized (Merkel, 2014) and deployed using the Kubernetes (Burns et al., 2016) container orchestration system. All the microservice communications are handled *via* the Apache Kafka (Kreps et al., 2011) message broker. We run 3 Kafka broker nodes with 3 Zookeeper nodes in Vind. The platform is running as a permissioned blockchain system in a private cloud. **Figure 3** shows the architecture of the Vind platform.

Rahasak blockchain (Bandara et al., 2018; 2021) has been used to implement the functionalities of the Vind platform. Rahasak is a big data-friendly enterprise-level, a scalable blockchain platform. The back-pressure operation of the scalable application is handled with the reactive streaming-based approach in the Rahasak blockchain. It comes with concurrency-enabled Aplos smart contracts (Bandara et al., 2019; 2020) which are written with the Scala functional programming language (Odersky et al., 2004) and Akka actors (Hewitt, 2010; Gupta, 2012). This smart contract platform supports high transaction throughput with supporting concurrent transaction execution on the blockchain. Rahasak blockchain facilitates full-text with the blockchain asset using Elasticsearch-based search API. We have selected the Rahasak blockchain to implement the Vind platform due to these enterprise-level features on it. We were able to support high transaction load, handle large volumes of back-pressure operations, and support full-text search on Vind using the Rahasak blockchain.

All the functionalities of the Vind platform are implemented with Aplos smart contracts in the Rahasak blockchain. There are four main smart contracts: a) identity contract, b) asset contract, c) notification contract, and d) verification contract. Web/mobile apps are the client applications on the Vind platform, invoking various functions implemented in smart contracts on the blockchain. The main functionalities include onboarding vendors, asset management in blockchain, report incidents, patch management, and view audit log. The requests generated from Web/mobile apps will be directed to blockchain smart contracts *via* the Vind gateway service which is HTTP REST API built with Golang (Schmager et al., 2010). Mobile/web client authentication/authorization will be handled by JWT-based (Jones, 2011) auth service in the Vind platform. Web/mobile client credential information will be stored in auth-storage (database) in the auth service. The asset in the Vind platform has assigned a unique digital identity. This identity is embedded in the QR code. The Vind mobile application comes with a QR code scanner. It can scan the QR code and fetch all the audit log information and attack information which are related to the software/hardware asset from the blockchain and visualize it.

The off-chain storage can store confidential data in the Vind platform. Each peer in the network has its own off-chain storage. They store confidential data on it and publish the hash of those data into the blockchain ledger. We have used Apache Cassandra (Lakshman and Malik, 2010) distributed storage to build the off-chain storage in the Vind platform. The off-chain storage is connected with Kibana and Grafana (Reelsen, 2014) analytics' dashboard to do the data visualizations. The CVE search API is a service that contains a list of all related CVE security vulnerabilities of common software products. There is a public

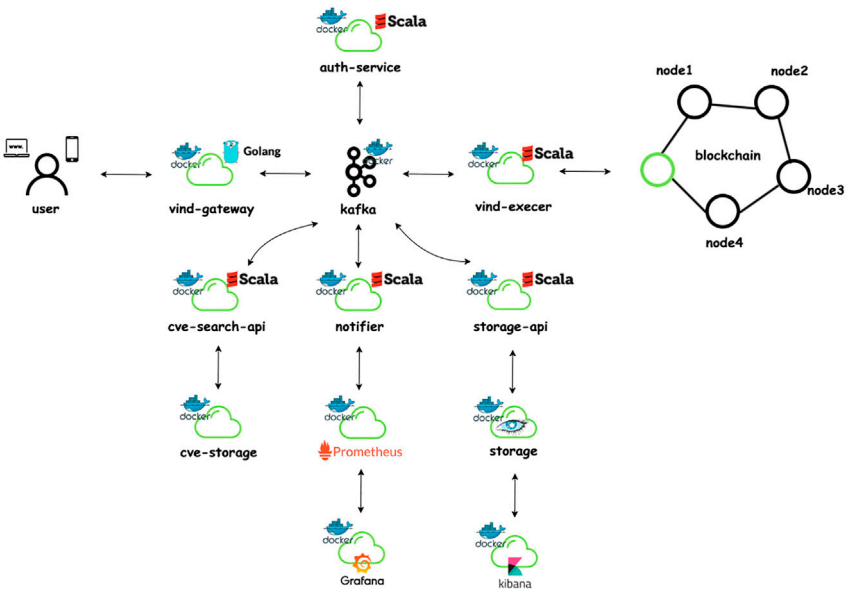

FIGURE 3 | Microservices-based architecture in the Vind platform. The platform is running as a permissioned blockchain system in a private cloud.

CVE database available with the known vulnerability of software products. In the Vind platform, we have run our own CVE search API service. It periodically fetches the vulnerabilities published in the public CVE search API and generates notifications based on the vulnerability. By using CVE search API, we have integrated automatic attack discovery and notification functions of assets in the Vind platform.

4 VIND PERFORMANCE EVALUATION

The performance evaluation of Vind was completed and is discussed. To evaluate the performance of the underlying blockchain ledger of the Vind platform, we have developed the Vind ledger service with three different blockchain systems [Rahasak Bandara et al. (2021), BigchaindDB McConaghy et al. (2016), Hyperledger Fabric Androulaki et al. (2018)] and compared the results. To obtain the results, we deployed the Vind platform with a multi-peer blockchain cluster in AWS 2xlarge instances (16 GB RAM and 8 CPUs). Rahasak blockchain runs with 4 Kafka nodes, 3 Zookeeper nodes, and Apache Cassandra (Lakshman and Malik, 2010) as the state database. The smart contracts on the Rahasak blockchain implemented with the Scala functional programming and an Akka actor-based Aplos (Bandara et al., 2019, 2020) smart contract platform. Hyperledger Fabric is set up to run with a Kafka-based consensus utilizing 3 Orderer nodes, 4 Kafka nodes, 3 Zookeeper nodes and LevelDB Androulaki et al. (2018) as the state database. BigchainDB blockchain is set up to run with Tendermint consensus Kwon (2014) and MongoDB Abramova and Bernardino (2013) as the state database. The evaluation results are obtained for the following, with the varying number of blockchain peers (1–15 peers) used in different evaluations. Rahasak blockchain currently does not support Byzantine failures, so in this evaluation, we did not consider the Byzantine behaviors.

1. Transaction throughput
2. Transaction scalability
3. Transaction execution rate
4. Search performance
5. Block generate time

4.1 Transaction Throughput

For this evaluation, we recorded the number of invoke transactions and query transactions that can be executed in each peer in the Vind platform. Invoke transaction creates a record in the ledger and updates the status of the assets in the blockchain. Query searches the status of the underlying blockchain ledger. They neither create transactions in the ledger nor update the asset status. We flooded concurrent transactions for each peer and recorded the number of completed results. As shown in **Figure 4**, we have obtained consistent throughput in each peer on the Vind platform. Since queries are not updating the ledger status, it has high throughput (2 times) compared to invoke transactions. Then the invoke transaction throughput of Vind is tested with different blockchain ledgers. We have implemented the Vind blockchain ledger service with Rahasak, BigchainDB, and Hyperledger Fabric platforms, and compared the transaction throughput results. As shown in **Figure 5**, the Vind ledger with Rahasak performs with a higher transaction throughput than BigchainDB and Hyperledger Fabric. The main reason for these observations is the enterprise-level features available in the Rahasak blockchain. Similar to invoke transactions, the Query operation throughput of the Vind

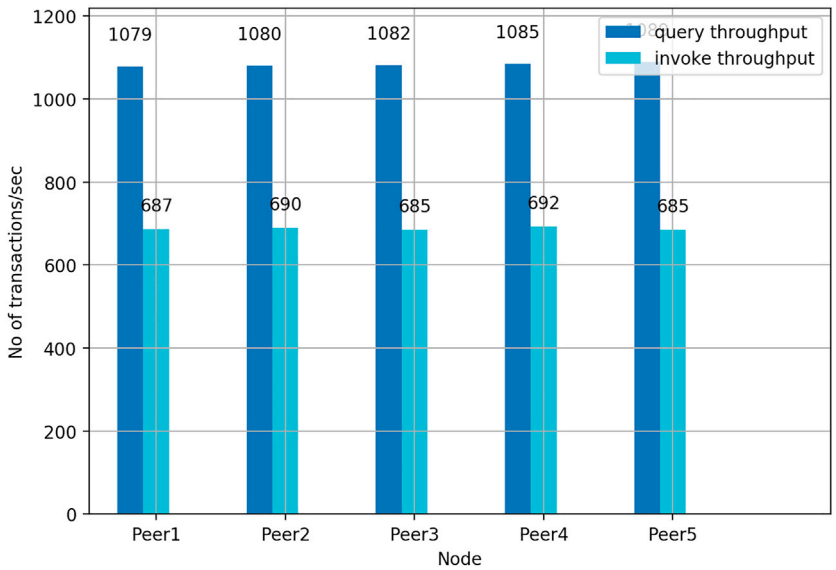

FIGURE 4 | Invoke and query transaction throughput of single blockchain peer in the Vind platform.

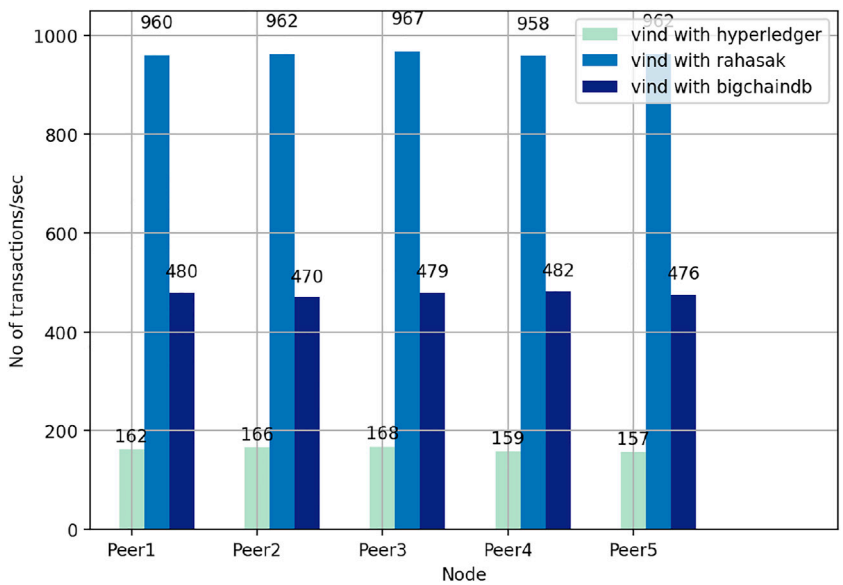

FIGURE 5 | Invoke transaction result of the Vind platform with different blockchain ledgers.

blockchain ledger tested with different blockchain platforms. Rahasak serves its query API *via* Apache Lucene index-based search API. BigchainDB facilitates the query API with MongoDB and Hyperledger Fabric with CouchDB. Query transaction throughput of Rahasak is higher than that of both BigchainDB and Hyperledger Fabric, as shown in **Figure 6**.

4.2 Transaction Scalability

For this evaluation, we recorded the number of executed invoke/query transactions (per second) over the number of blockchain peers in the network. We issued concurrent transactions in each blockchain peer and recorded the number of executed transactions. **Figure 7** shows the transaction scalability comparison of the Vind platforms' blockchain. When adding a node to the cluster, it nearly linearly increases the transaction throughput. This means the transaction latency will be decreased when adding blockchain peers to the cluster. Query transactions have high scalability when comparing to invoke transactions. The main reason is query transactions are not updating the ledger status like invoke transactions. Then we have tested the Vind

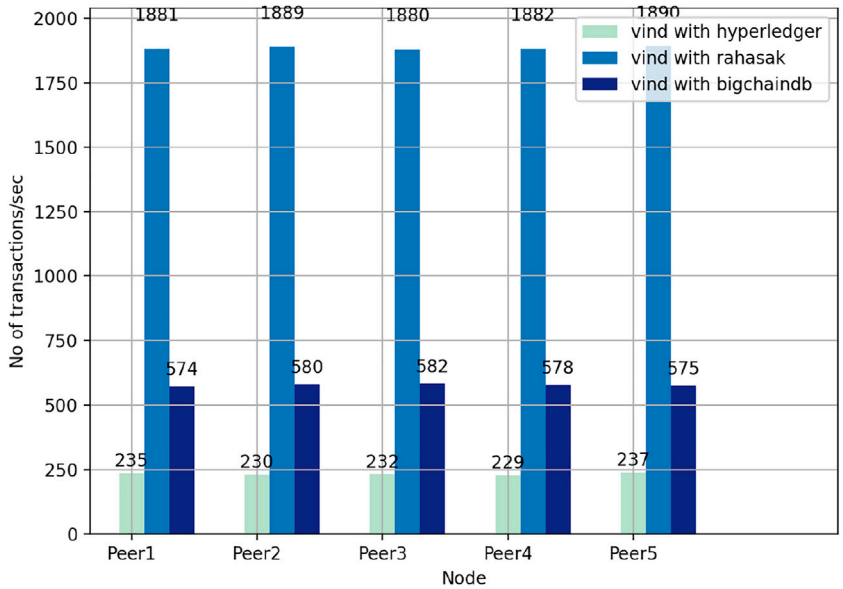

FIGURE 6 | Query transaction result of the Vind platform with different blockchain ledgers.

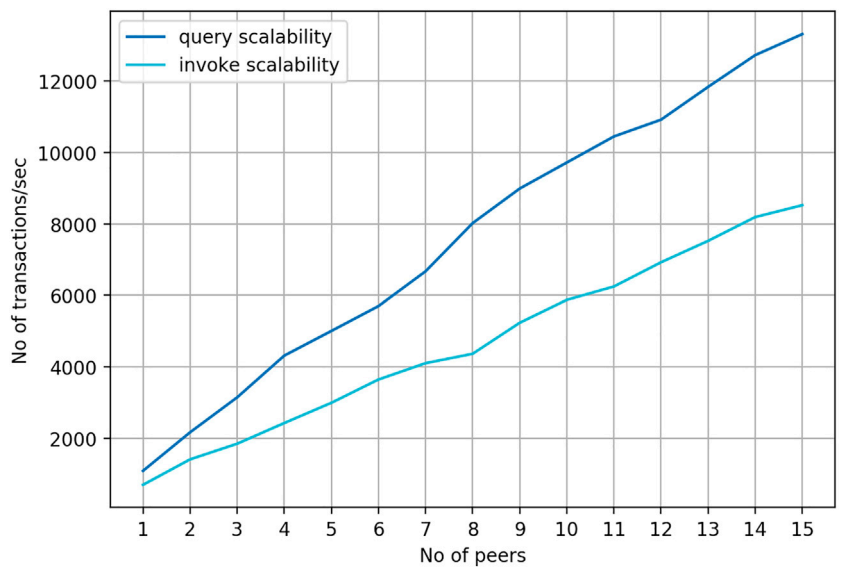

FIGURE 7 | Invoke/query transaction scalability (per second) over the number of blockchain peers in the network.

platforms' blockchain ledger scalability results with the different blockchain systems. As shown in **Figure 8**, the transaction scalability of the Vind ledger is compared in Rahasak, BigchainDB, and Hyperledger Fabric. Vind blockchain ledger with Rahasak has a higher scalability than that with BigchainDB and Hyperledger Fabric blockchains. Both Hyperledger Fabric and BigchainDB blockchains do not support the real-time transaction-enabled blockchain architecture. Rahasak blockchain supports the real-time transaction-enabled "Validate-Execute-Group" blockchain architecture. Also, Rahasak blockchain supports the concurrent transaction execution–enabled smart contract platform which does not exist in BigchainDB or Hyperledger fabric. Due to these reasons, the Vind ledger with the Rahasak blockchain has higher scalability than BigchaindDB and Hyperledger Fabric.

4.3 Transaction Execution Rate

Next, we evaluate the transaction execution rate in the Vind platform. We tested the number of submitted transactions and

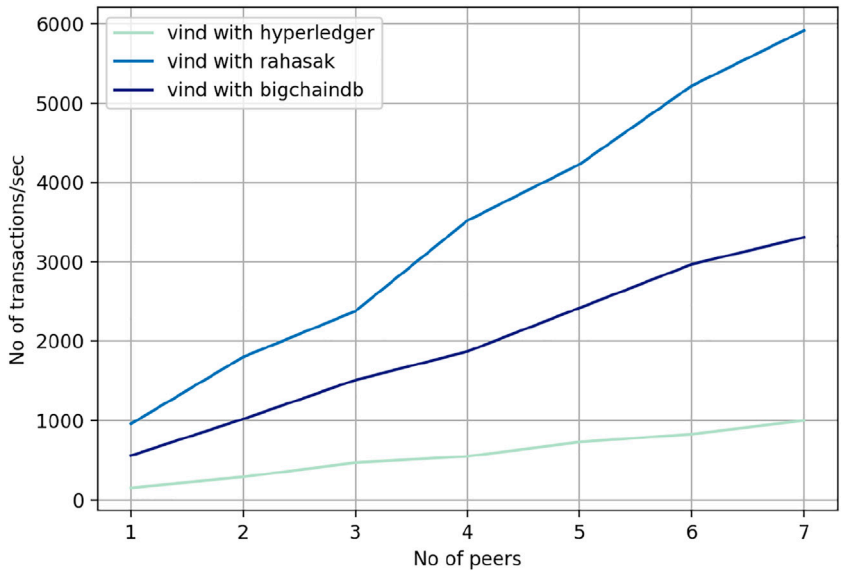

FIGURE 8 | Transaction scalability (per second) of the Vind blockchain ledger with different blockchain platforms.

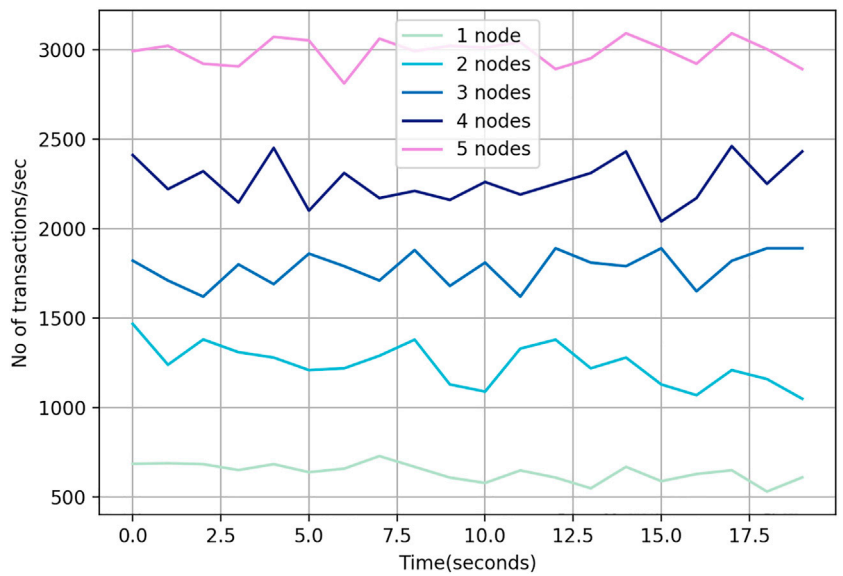

FIGURE 9 | Transaction execution rate comparison with a number of blockchain nodes in the Vind blockchain.

executed transactions in different blockchain peers recording the time. **Figure 9** shows how the transaction execution rate varies when having different numbers of blockchain peers in the Vind. When the number of peers increases, the rate of executed transactions is increased relatively. **Figure 10** shows the number of executed transactions and submitted transactions in a single blockchain peer. There is a back-pressure operation (Destounis et al., 2016; Davis, 2019) between the rates of submitted transactions and executed transactions. We have used a reactive streaming-based approach with Apache Kafka to handle the back-pressure operations in the Vind platform.

4.4 Search Performance

Vind allows searching its transaction information and asset information *via* underlying Rahasak blockchains' Lucene index-based search API (Gormley and Tong, 2015). For this evaluation, we issued concurrent search queries into Vind and computed the search time. As shown in **Figure 11**, to search 2 million records connected, it took 4 ms of time.

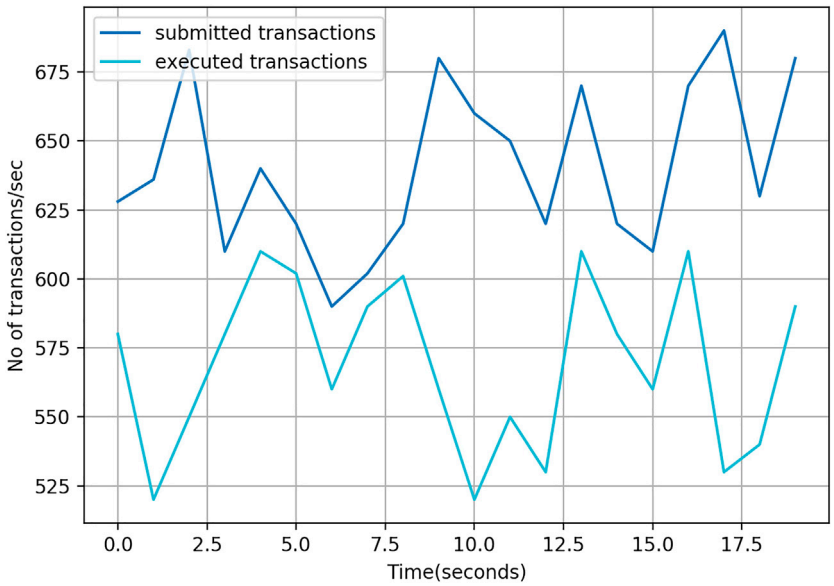

FIGURE 10 | Transaction execution rate and transaction submission rate in a single blockchain peer.

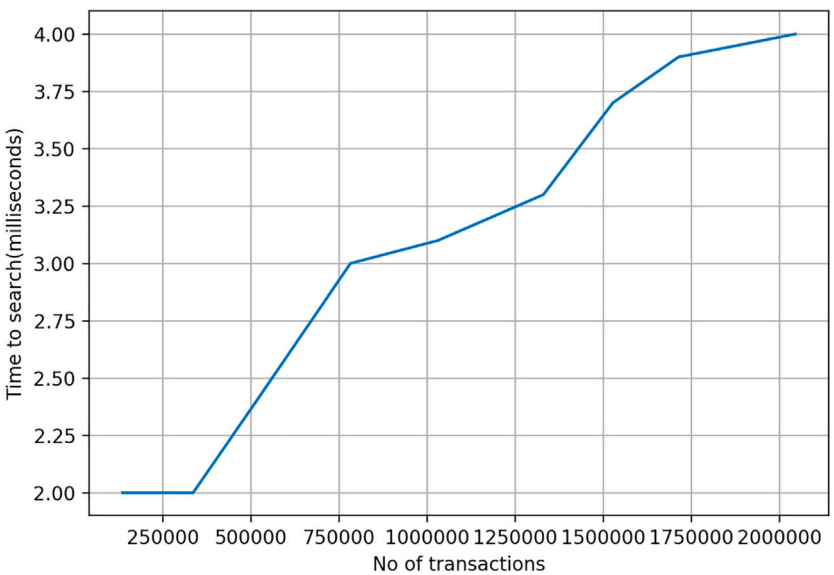

FIGURE 11 | Transaction/asset search performance in the Vind blockchain platform.

4.5 Block Generate Time

Finally, we have evaluated the time taken to create blocks in the underlying blockchain storage of the Vind platform. The statistics recorded against the number of transactions in a block. Block generate time depends on a) data replication time, b) Merkle proof/block hash generate time, and c) transaction validation time. When the transaction count increases in the block, these factors will be increased. Due to this reason, when the transaction count increases, block generation time also increases correspondingly. As shown in **Figure 12**, it takes approximately 8 s to increase a block when creating blocks of 10 k transactions.

5 RELATED WORKS

Much research has been conducted to integrate blockchain with the supply chain management (Korpela et al., 2017). In this section, we briefly discuss the main features and architecture of these research projects.

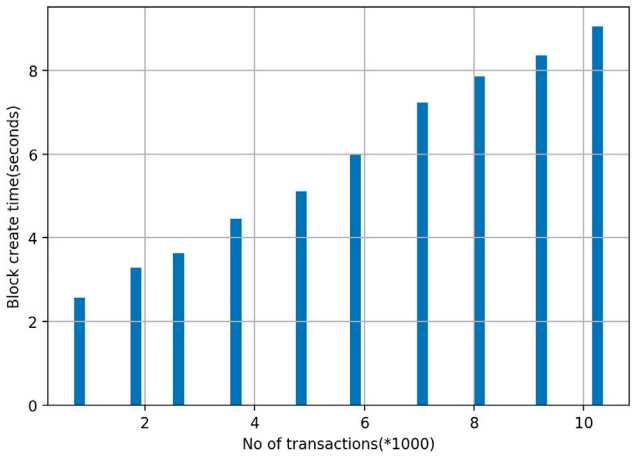

FIGURE 12 | Block creation time against the number of transactions in the block.

"*Supply chain risk*" (Mylrea and Gourisetti, 2018) and "*EDS supply chain*" (Liang et al., 2018) works proposed blockchain-based supply chain management systems for energy delivery systems. The supply chain risk effort mainly focused on utilizing proof of authority (PoA)-based permissioned blockchain to manage the supply chain life cycle in the EDSs. The supply chain of the EDS implemented the supply chain functions in energy delivery systems using hyperledger fabric blockchain smart contracts. These functions include identity management, product verification, access control, contract execution, and monitoring.

"*HACCP supply chain*" (Tian, 2017), "*AgriBlockIoT*" (Caro et al., 2018), "*Multichain WSC*" (Biswas et al., 2017), and "*Agrifood Supply Chain*" (Tian, 2016) works proposed blockchain-based food supply chain management and traceability systems. The HACCP supply chain system has given an example scenario to demonstrate how it works in the food supply chain with HACCP. This system is built on top of the BigchainDB scalable blockchain system to support highly scalable applications. The AgriBlockIOT system can rely either on the Ethereum or the Hyperledger Sawtooth publicly available blockchain implementations, while it is able to integrate various IoT sensor devices. As a use case, they have implemented and discussed traditional food supply chain scenarios from agricultural farms to customer consumption on top of the blockchain. Multichain WS proposed a blockchain-based wine supply chain traceability system which is built over a multichain-permissioned blockchain platform (Greenspan, 2015). The proposed solution is designed and implemented with five entities in the wine supply chain, namely, grape grower, wine producer, bulk distributor, filler/packer, and retailer. Agrifood Supply Chain is built to help Chinese agri-food markets to enhance their food safety and quality. They mainly rely on the RFID technology to implement data acquisition, circulation, and sharing in production, processing, warehousing, distribution, and sales links of the agrifood supply chain.

"*Manufacture supply chain*" (Abeyratne and Monfared, 2016) work proposed the supply chain traceability system for manufacturing domain. The manufacturing of cardboard boxes is used as an example to demonstrate how such technology can be used in a global supply chain network. The proposed approach comprises a decentralized distributed system that uses blockchain(s) to collect, store, and manage key information of each product throughout its life cycle. When a product moves its life cycle, each actor (e.g., producers, suppliers, manufacturers, distributors, retailers, and finally the end consumer) logs information about the product and its current status on the blockchain network. Each product would be attached with an information tag, which could be in the form of a barcode, RFID, or QR code. The registered actors in the system can access the physical asset's virtual profile and life cycle events by scanning the tag.

The comparison summary of these supply chain management platforms and the Vind platform is presented in **Table 1**. It compares implemented blockchain platform, blockchain type, consensus, application domain, scalability, and privacy-level details.

TABLE 1 | Blockchain platform comparison.

Platform	Implemented blockchain	Blockchain type (permissioned/ public)	Blockchain consensus	Application domain	Scalability	Privacy	NERC compliance [Cip (2018), Mylrea and Gourisetti (2018)]
Vind	Rahasak	Permissioned	Paxos/ZAB	Energy	High	High	Yes
NERC CIP supply chain	N/A	Permissioned	PoA	Energy	Mid	High	Yes
EDS supply chain	Hyperledger	Permissioned	Kafka/ZAB	Energy	Mid	High	No
HACCP supply chain	Bigchaindb	Permissioned	Tendermint	Food	High	High	N/A
AgriBlockIoT	Ethereum	Public	PoS	Food	Mid	High	N/A
Multichain WSC	Multichain	Permissioned	PoW	Food(wine)	Low	Mid	N/A
Agri-food supply chain	Any	Any	Any	Food	N/A	N/A	N/A
Manufacture supply chain	Any	Any	Any	Manufacture	N/A	N/A	N/A

6 CONCLUSION AND FUTURE WORK

With Vind, we have introduced a blockchain-based cyber supply chain provenance platform for energy delivery systems. It has addressed the requirements of supply chain risk management. We have used blockchain to store all the physical/software asset information, their supply chain provenance information, and audit log information in EDSs. To address privacy concerns, off-chain storage has been integrated into the blockchain. By using the Rahasak blockchain to build the Vind platform, we were able to achieve higher transaction throughput and meet the scalability requirements. We have demonstrated the scalability and transaction throughput requirements with extensive empirical evaluations. Version 1.0 of the Vind platform as described above in this article is currently running as a prototype.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

EB: methodology, software implementation and draft preparation. DT: investigation, writing—reviewing and editing. XL: investigation, writing—reviewing and editing. SS: supervision, investigation, writing—reviewing and editing.

FUNDING

This work was funded by the Department of Energy (DOE) Office of Fossil Energy (Federal Grant No. #DE-FE0031744).

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fbloc.2021.607320/full#supplementary-material

REFERENCES

Abeyratne, S. A., and Monfared, R. P. (2016). Blockchain Ready Manufacturing Supply Chain Using Distributed Ledger.. *Ijret* 05, 1–10. doi:10.15623/ijret.2016.0509001

Abramova, V., and Bernardino, J. (2013). "Nosql Databases: Mongodb vs Cassandra," in Proceedings of the international C* conference on computer science and software engineering, 14–22.

Abeyratne, S. A., and Monfared, R. P. (2016). Blockchain Ready Manufacturing Supply Chain Using Distributed Ledger. *Ijret* 05, 1–10. doi:10.15623/ijret.2016.0509001

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., et al. (2018). "Hyperledger Fabric: a Distributed Operating System for Permissioned Blockchains," in Proceedings of the Thirteenth EuroSys Conference (ACM), 30.

Baars, D. (2016). "Towards Self-Sovereign Identity Using Blockchain Technology," (University of Twente). Master's thesis.

Bandara, E., Liang, X., Foytik, P., Shetty, S., Ranasinghe, N., De Zoysa, K., et al. (2020). "Saas-microservices-based Scalable Smart Contract Architecture," in Security in Computing and Communications: 8th International Symposium, Chennai, India, October 14–17, 2020 (SSCC 2020 Springer Nature), 228. Revised Selected Papers.

Bandara, E., Liang, X., Foytik, P., Shetty, S., Ranasinghe, N., and De Zoysa, K. (2021). Rahasak-scalable Blockchain Architecture for enterprise Applications. *J. Syst. Architecture* 116, 102061. doi:10.1016/j.sysarc.2021.102061

Bandara, E., Ng, W. K., de Zoysa, K., Fernando, N., Tharaka, S., Maurakirinathan, P., et al. (2018). "Mystiko–blockchain Meets Big Data," in 2018 IEEE International Conference on Big Data (Big Data) (IEEE), 3024–3032.

Bandara, E., Ng, W. K., De Zoysa, K., and Ranasinghe, N. (2019). *Aplos: Smart Contracts Made Smart*. BlockSys' 2019.

Biswas, K., Muthukkumarasamy, V., and Tan, W. L. (2017). "Blockchain Based Wine Supply Chain Traceability System," in Future Technologies Conference, Vancouver, BC, Canada.

Burns, B., Grant, B., Oppenheimer, D., Brewer, E., and Wilkes, J. (2016). Borg, omega, and Kubernetes. *Queue* 14, 70–93. doi:10.1145/2898442.2898444

Caro, M. P., Ali, M. S., Vecchio, M., and Giaffreda, R. (2018). "Blockchain-based Traceability in Agri-Food Supply Chain Management: A Practical Implementation," in 2018 IoT Vertical and Topical Summit on Agriculture-Tuscany (IOT Tuscany) (IEEE), 1–4.

Cip, N. (2018). NERC CIP 013.

Contract, M. P. (2020). *Model Procurement Contract Language*.

CSRC (2018). *Software Supply Chain Attacks*.

Davis, A. L. (2019). "Akka Streams," in *Reactive Streams in Java* (Springer), 57–70. doi:10.1007/978-1-4842-4176-9_6

Destounis, A., Paschos, G. S., and Koutsopoulos, I. (2016). "Streaming Big Data Meets Backpressure in Distributed Network Computation," in IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications (IEEE), 1–9.

Du, S., Lu, T., Zhao, L., Xu, B., Guo, X., and Yang, H. (2013). "Towards an Analysis of Software Supply Chain Risk Management," in Proceedings of the World Congress on Engineering and Computer Science.

Ellison, R. J., and Woody, C. (2010). "Supply-chain Risk Management: Incorporating Security into Software Development," in 2010 43rd Hawaii International Conference on System Sciences (IEEE), 1–10.

Goff, E., Glantz, C., and Massello, R. (2014). "Cybersecurity Procurement Language for Energy Delivery Systems," in Proceedings of the 9th Annual Cyber and Information Security Research Conference, 77–79.

Gormley, C., and Tong, Z. (2015). *Elasticsearch: The Definitive Guide: a Distributed Real-Time Search and Analytics Engine*. (O'Reilly Media, Inc.).

Greenspan, G. (2015). Multichain Private Blockchain-white Paper. Available at: http://www.multichain.com/download/MultiChain-White-Paper.pdf

Gueron, S., Johnson, S., and Walker, J. (2011). "Sha-512/256," in 2011 Eighth International Conference on Information Technology: New Generations (IEEE), 354–358.

Gupta, M. (2012). *Akka Essentials*. Packt Publishing Ltd.

Hewitt, C. (2010). Actor Model of Computation: Scalable Robust Information Systems. arXiv preprint arXiv:1008.1459

Ivanov, D. (2018). *Structural Dynamics and Resilience in Supply Chain Risk Management*. Springer.

Jones, M. B. (2011). *The Emerging Json-Based Identity Protocol Suite*. W3C workshop on identity in the browser, 1–3.

Korpela, K., Hallikas, J., and Dahlberg, T. (2017). "Digital Supply Chain Transformation toward Blockchain Integration," in proceedings of the 50th Hawaii international conference on system sciences, Big Island, Hawaii.

Kreps, J., Narkhede, N., Rao, J., et al. (2011). Kafka: A Distributed Messaging System for Log Processing. *Proc. NetDB* 1–7.

Kwon, J. (2014). *Tendermint: Consensus without Mining*, 11. Draft v. 0.6, fall 1.

Lakshman, A., and Malik, P. (2010). Cassandra. *SIGOPS Oper. Syst. Rev.* 44, 35–40. doi:10.1145/1773912.1773922

Liang, X., Shetty, S., Tosh, D., ji, Y., and Li, D. (2018). "Towards a Reliable and Accountable Cyber Supply Chain in Energy Delivery System Using Blockchain," in 14th EAI International Conference on Security and Privacy in Communications Networks (SecureComm '18), Singapore.

McConaghy, T., Marques, R., Müller, A., De Jonghe, D., McConaghy, T., McMullen, G., et al. (2016). *Bigchaindb: A Scalable Blockchain Database*. BigChainDB white paper.

Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. *Linux J.* 2014, 2.

Mühle, A., Grüner, A., Gayvoronskaya, T., and Meinel, C. (2018). A Survey on Essential Components of a Self-Sovereign Identity. *Comput. Sci. Rev.* 30, 80–86. doi:10.1016/j.cosrev.2018.10.002

Mylrea, M., and Gourisetti, S. N. G. (2018). Blockchain: Next Generation Supply Chain Security for Energy Infrastructure and Nerc Critical Infrastructure protection (Cip) Compliance. *Resilience Week* 16.

Odersky, M., Altherr, P., Cremet, V., Emir, B., Maneth, S., Micheloud, S., et al. (2004). An Overview of the Scala Programming Language. Tech. rep.

Pham, V., and Dang, T. (2018). "Cvexplorer: Multidimensional Visualization for Common Vulnerabilities and Exposures," in 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10-13 Dec. 2018 (IEEE), 1296–1301.

Reelsen, A. (2014). Using Elasticsearch, Logstash and Kibana to Create Realtime Dashboards. Dostupné z: Available at: https://secure.trifork.com/dl/goto-berlin-2014/GOTO_Night/logstash-kibanaintro.pdf.

Schmager, F., Cameron, N., and Noble, J. (2010). "Gohotdraw: Evaluating the Go Programming Language with Design Patterns," in Evaluation and Usability of Programming Languages and Tools (ACM), 10.

Tian, F. (2017). "A Supply Chain Traceability System for Food Safety Based on Haccp, Blockchain & Internet of Things," in International conference on service systems and service management, Dalian, China, 16-18 June 2017, 1–6.

Tian, F. (2016). "An Agri-Food Supply Chain Traceability System for china Based on Rfid & Blockchain Technology," in 2016 13th international conference on service systems and service management (ICSSSM), Kunming, China, 24-26 June 2016 (IEEE), 1–6.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.