International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2, 2025

Dynamic Data Storage And Verifications In Cloud Environments

[1]Mr. K. Anil Kumar, [2]Kethavath Pavan Kalyan, [3]Mudigonda Sai Abhinav, [4]Mamidikayala Sudheer

[1]Assistanat Professor, Department of Information Technology, Guru Nanak Institutions Technical Campus, India.

[2,3,4]B.Tech Students, Department of Information Technology, Guru Nanak Institutions Technical Campus, India.

pavankethavath9010@gmail.com

ABSTRACT:

This project explores a decentralized data management system involving data owners, data processors, fog servers, cloud service providers, and third-party verifiers. The primary goal is to ensure secure data access, authorization, and verification in a distributed environment. The data owner shares sensitive data with a data processor, who uses fog servers for processing. To achieve decentralization, two fog servers are implemented to distribute the workload and enhance system reliability. These fog servers are responsible for managing the data between the data owner and the data processor. The project we have implanting a data share we have used a Homomorphic Encryption The cloud service provider plays a crucial role by maintaining the infrastructure and providing the necessary resources for the servers. Additionally, a third-party verifier is introduced to oversee the integrity and validity of the data access requests. The third-party verifier requests permission from the data owner to view a particular file. If the data owner grants permission, the file is made accessible to the verifier; otherwise, access is denied. This project aims to provide a robust, decentralized architecture that secures data access, ensures privacy, and allows for transparent verification of access requests. By leveraging fog computing, decentralization, and third-party verification, the system ensures that data owners retain control over their files while facilitating secure and authorized data interactions.

1-INTRODUCTION

INDUSTRIAL Internet platforms (IIP), also called industrial IoT (IIoT) platforms (e.g., GE Predix, Siemens Mind- Sphere, PTC ThingWorx, etc.) [1], are rapidly emerging worldwide and have received extensive attention from both academia and industry. The IIP integrates a variety of manufacturing resources or capabilities of different enterprises to provide services for platform users. Conversely, the IIP can also analyse the data from connected products, plants and systems to improve the reliability, flexibility, and efficiency of the industrial systems in different sectors, such as manufacturing, energy, logistics, etc.

Existing notable IIPs including Mind Sphere and Predix are built based on an edge-cloud infrastructure [4], in which fog nodes collect an enormous amount of industrial data from IIoT devices whilst the cloud stores these data for further optimizing operations, creating better quality products, or deploying new business models, etc [5]. In this infrastructure, the cloud service plays an essential role in the IIP that stores all collected data for data analysis. However, it also brings security challenges to the industrial data and system. On one hand, as a centralized infrastructure, cloud storage suffers from a single point of failure [6] due to many uncertainties, e.g., system breakdowns, power outages, etc. This gives rise to a growing number of data loss cases in many notable cloud service providers such as Google and Tencent. As a consequence, the cloud service provider (CSP) may store corrupted input/output of analysis tasks, and finally forward inaccurate analysis results to cloud users or send incorrect control decisions to devices. On the other hand, when the industrial cloud is externally attacked, the attackers may maliciously modify the collected running data (to camouflage equipment breakdowns), forge the operation/configuration data, or manipulate the results and decisions of industrial applications, which may result in the manipulation or destruction of the entire industrial system [7]. For instance, Stuxnet virus destroyed a large number of centrifuges in the Natanz power station by tampering with their operation data [8]. Therefore, cloud data integrity guarantee becomes one of the most concerning issues in cloud-based IIP.

EXISTING SYSTEM:

This gives rise to a growing number of data loss cases in many notable cloud service providers such as cloud data.

On the other hand, when the industrial cloud is externally attacked, the attackers may maliciously modify the collected running data forget the operation/configuration data, or manipulate the results and decisions of industrial applications, which may result in the manipulation or destruction of the entire industrial system.

The security of the approach Ekavic is analyzed in the random. A prototype of Ekavic is implemented and extensive experiments demonstrate that Edasvic incurs fewer computational costs than the state-of-the-art approaches.

2-LITERATURE SURVEY

TITLE: FairShare: Blockchain enabled fair, accountable and secure data sharing for industrial IoT
AUTHOR: J. Sengupta, S. Ruj, and S. D. Bit,
YEAR: 2023
DESCRIPTION:
Industrial Internet of Things (IIoT) opens up a challenging research area towards improving secure data sharing which currently has several limitations. Primarily, the lack of inbuilt guarantees of honest behavior of participating, such as end-users or cloud behaving maliciously may result in disputes. Given such challenges, we propose a fair, accountable, and secure data sharing scheme, Fairshare for IIoT. In this scheme, data collected from IoT devices are processed and stored in cloud servers with intermediate fog nodes facilitating computation. Authorized clients can access this data against some fee to make strategic decisions for improving the operational services of the IIoT system. By enabling blockchain, Fairshare prevents fraudulent activities and thereby achieves fairness such that each party gets their rightful outcome in terms of data or penalty/rewards while simultaneously ensuring accountability of the services provided by the parties. Additionally, smart contracts are designed to act as a mediator during any dispute by enforcing payment settlement. Further, security and privacy of data are ensured by suitably applying cryptographic techniques like proxy re-encryption. We prove Fairshare to be secure as long as at least one of the parties is honest. We validate Fairshare with a theoretical overhead analysis. We also build a prototype in Ethereum to estimate performance and justify comparable results with a state-of-the-art scheme both via simulation and a realistic testbed setup. We observe an additional communication overhead of 256 bytes and a cost of deployment of 1.01 USD in Ethereum which are constant irrespective of file size..

TITLE: Trust management in industrial Internet of Things
AUTHOR: Y. Wan, X. Lin, K. Xu, F. Wang, and G. Xue,
YEAR: 2023
DESCRIPTION:
Automobile manufacturers around the world are increasingly deploying Industrial Internet of Things (IIoT) devices in their factories to accompany the Industrial Revolution 4.0. Security and privacy are the main limitations to the integration of Internet of Things (IoT) into industrial processes. Therefore, it is necessary to protect industrial data contained in IIoT devices and keep them confidential. As a step towards this direction, in this paper, we propose a dynamic trust management model suitable for industrial environments. We propose also to change the traditional centralized architecture of IIoT networks in automotive plants into a hybrid architecture based on a set of new industrial relationship rules. The performance evaluation in this work is done in two parts. In the first part, we compare our proposed architecture with the traditional architecture of the plant's IIoT network. The results of this comparison show that our architecture is more suitable to simplify trust management of IIoT devices. In the second part, we demonstrated the ability, the adaptiveness and the resiliency of our proposed trust model against behavioral changes of IIoT nodes in malicious environments.

TITLE: Blockchain-based offline auditing for the cloud in vehicular networks
AUTHOR: H. Yu, Z. Yang, S. Tu, M. Waqas, and H. Liu,
YEAR: 2022
DESCRIPTION:
The rapid growth of various vehicular apps such as automotive navigation and in-car entertainment has brought the explosion of vehicular data. Such a growth has given rise to a huge challenge of maintaining the quality of cloud storage services for the whole period of storage in vehicular networks. As a result, poor quality of services easily causes data corruption problems and thereby threats vehicular data integrity. Blockchain, a tamper-proofing technique, is considered a promising approach for mitigating data integrity risks in cloud storage. However, existing blockchain-based schemes for auditing long-term cloud data integrity suffer from poor communication performance in a vehicular network. In this study, a blockchain-based offline auditing scheme for cloud storage in the vehicular network is proposed to improve auditing performance. Inspired by the data structure of blockchain, we design an evidence chain to achieve offline auditing, which allows the cloud to spontaneously generate data integrity evidence without communicating with auditors during the evidence generation phase. Furthermore, we extend our scheme to support public and automatic validation based on the smart contract. We prove the security of the proposed scheme under the random oracle model and further provide the performance evaluation by comparing with the state-of-the-art approaches.

TITLE: Privacy-preserving proof of storage for the pay-as-you-go business model
AUTHOR: T. Wu, G. Yang, Y. Mu, F. Guo, and R. H. Deng,
YEAR: 2022
DESCRIPTION:
Proof of Storage (PoS) enables a cloud storage provider to prove that a client's data is intact. However, existing PoS protocols are not designed for the pay-as-you-go business model in which payment is made based on both storage volume and

duration. In this paper, we propose two PoS protocols suitable for the pay-as-you-go storage business model. The first is a time encapsulated Proof of Retrievability (PoR) protocol that ensures retrievability of the original file upon successful auditing by a client. Considering the large size of outsourced data, we then extend the protocol to a privacy-preserving public auditing protocol which allows a third party auditor to audit outsourced data on behalf of its clients without sacrificing the privacy of the data or the timestamp (i.e., time of storage). We formalize the definition, system model and security model of the proposed PoS system and prove the security of the proposed protocols by a sequence of games in the algebraic group model with a random oracle. We analyze the performance of the protocols both theoretically and experimentally and show that the protocols are practical.

TITLE: Transparent ciphertext retrieval system supporting integration of encrypted heterogeneous database in cloud-assisted IoT
AUTHOR: X. Feng, J. Ma, S. Liu, Y. Miao, X. Liu, and K. R. Choo,
YEAR: 2023
DESCRIPTION:
Ciphertext retrieval under heterogeneous data sets is a critical concern in the cloud-assisted Internet of Things (IoT). Previous ciphertext retrieval schemes eliminate computation burden and security concerns to some extent, but cannot support the encrypted heterogeneous data sets. To solve this problem, in this article we propose a transparent ciphertext retrieval system regardless of programming language, accessible platforms, and heterogeneity of data sets. The retrieval system is a novel combination of the proposed access and integration middleware with a current encrypted cloud database system. The middleware enables cross-language and cross-platform queries over various database systems; besides, it schedules the designed data integration method in heterogeneous databases to realize cross-database queries over the encrypted cloud data sets. The proposed ciphertext retrieval system achieves efficient data sharing among different IoT applications with various database systems. We prove that the system guarantees data security by avoiding malicious hackers. Extensive experiments show that our design is efficient and feasible in practice..

TITLE: secure fog-based architecture for industrial Internet of Things and Industry
AUTHOR: J. Sengupta, S. Ruj, and S. D. Bit,
YEAR: 2023
DESCRIPTION:
The advent of Industrial Internet of Things (IIoT) along with cloud computing has brought a huge paradigm shift in manufacturing industries resulting in yet another industrial revolution, Industry 4.0. Huge amounts of delay-sensitive data of diverse nature are being generated, which need to be locally processed and secured because of their sensitivity. However, the low-end Internet of Things devices are unable to handle huge computational overheads. In addition, the semi-trusted nature of cloud introduces several security concerns. To address these issues, this article proposes a secure fog-based IIoT architecture by suitably plugging a number of security features into it and by offloading some of the tasks judiciously to fog nodes. These features secure the system alongside reducing the trust and burden on the cloud and resource-constrained devices, respectively. We validate our proposed architecture through both theoretical overhead analysis and practical experimentation, including simulation study and testbed implementation.

3-PROJECT DESCRIPTION

The proposed Edasvic mainly focuses on the protection of data integrity. The challenge-response process in Edasvic can also protect the privacy of the stored data. When considering the availability of the stored cloud data, since Edasvic periodically checks the integrity of cloud data, it partially guarantees the availability of data during storage verification with high probability. That is, the stored data is intact and can be accessed from the cloud during each round of verification. Nonetheless, since the proposed scheme is principally dedicated to maintaining the integrity of cloud data, there still requires more direct measurements such as data backup and recovery load balancing and data monitoring to fully guarantee the availability of cloud data in IIP.

METHODOLOGIES

User Interface Design
In this module we design the windows for the project. These windows are used for secure login for all users. To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page.
Owner
The Owner has a register with a user id and password. The Owner has login. It generate a private keys. The Data owner has a upload a data to store in a database. The Owner has a data request to to view a data. The Owner2 has login with a user id and password. The owner 2 has a data recovery to recover a data. The owner has a view a owner data. The data has a download a data from a database.

4-REQUIREMENTS ENGINEERING

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 13, Issue 2, 2025

We have conducted experiments on our collected dataset and extensive results have demonstrated that our model outperforms all other existing models. In the future, we will investigate more tasks under this framework, such as event summarization and event attribute mining in social media.

HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system does and not how it should be implemented.

- PROCESSOR : DUAL CORE 2 DUOS.
- RAM : 2GB DD RAM
- HARD DISK : 250 GB

SOFTWARE REQUIREMENTS:

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team's progress throughout the development activity.

5-DESIGN ENGINEERING

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

1. **User Authentication:**
   - The system must allow users (Owners, Owner2, TPA) to log in using a user ID and password.
   - The system must generate private keys for data owners upon successful login.
2. **Data Management:**
   - The data owner must be able to upload data to the database.
   - The data owner must be able to request and view their data.
   - Owner2 must be able to recover data from the database.
   - The system must allow data owners to download data from the database.
3. **Fog Server Management:**
   - The system must support multiple fog servers (fog server 1 and fog server 2).
   - The system must allow the addition of new owners (Owner2) to fog server 2.
   - The system must store owner data on fog server 1 and fog server 2.

3. Data Processor

With a user ID and password, the Data Processor may be accessed. In a database, there are additional owners for fog server 1. The owner data is on the fog server 1. The login credentials for the fog server 2 are a user ID and password. To add members, the fog server2 includes an add owner2 feature. There is a fog server2 added to owner data2.

4. Cloud Service Provider

A user ID and password are required to access the cloud service provider's login. The CSP has the authority to authorize data. The CSP can see the data. The CSP displays all of the information for a fog server1. The CSP's fog server2 allows users to view all of the server information.

5. Third Party Auditor

The tpa has login with a user id and password. The TPA has a view a data. The TPA has a approves a download data.

6-DEVELOPMENT TOOLS

This chapter is about the software language and the tools used in the development of the project. The platform used here is JAVA. The Primary languages are JAVA,J2EE and J2ME. In this project J2EE is chosen for implementation.

FEATURES OF JAVA THE JAVA FRAMEWORK

Java is a programming language originally developed by James Gosling at Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere".

Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applications the java framework is a new platform independent that simplifies application development internet. Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game

consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

EVOLUTION OF COLLECTION FRAMEWORK

Almost all collections in Java are derived from the java.util.Collection interface. Collection defines the basic parts of all collections. The interface states the add() and remove() methods for adding to and removing from a collection respectively. Also required is the toArray() method, which converts the collection into a simple array of all the elements in the collection. Finally, the contains() method checks if a specified element is in the collection. The Collection interface is a subinterface of java.util.Iterable, so the iterator() method is also provided. All collections have an iterator that goes through all of the elements in the collection. Additionally, Collection is a generic. Any collection can be written to store any class. For example, Collection<String> can hold strings, and the elements from the collection can be used as strings without any casting required.

MAP:

Maps are defined by the java.util.Map interface in Java. Maps are simple data structures that associate a key with a value. The element is the value. This lets the map be very flexible. If the key is the hash code of the element, the map is essentially a set. If it's just an increasing number, it becomes a list. Maps are implemented by java.util.HashMap, java.util.LinkedHashMap, and java.util.TreeMap. HashMap uses a hash table. The hashes of the keys are used to find the values in various buckets. LinkedHashMap extends this by creating a doubly linked list between the elements. This allows the elements to be accessed in the order in which they were inserted into the map. TreeMap, in contrast to HashMap and LinkedHashMap, uses a red-black tree. The keys are used as the values for the nodes in the tree, and the nodes point to the values in the map

THREAD:

Simply put, a *thread* is a program's path of execution. Most programs written today run as a single thread, causing problems when multiple events or actions need to occur at the same time. Let's say, for example, a program is not capable of drawing pictures while reading keystrokes. The program must give its full attention to the keyboard input lacking the ability to handle more than one event at a time. The ideal solution to this problem is the seamless execution of two or more sections of a program at the same time.

7-TESTING

UNIT TESTING:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

FUNCTIONAL TEST:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

SYSTEM TEST:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

PERFORMANCE TEST:

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

INTEGRATION TESTING:

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up –

software applications at the company level – interact without error.

8- CONCLUSION

The integrity of industrial data in the cloud can directly influence the performance of intelligent applications and services provided by industrial internet platforms, and thus should be guaranteed for both data owners and platform users. In this work, we propose an efficient and dynamic storage verification scheme for cloud-based IIP. Thanks to the design of a light-weight homomorphic authenticator and an authenticator accumulator, the computational overheads of pre-processing the data can be minimized.

REFERENCES:

[1] J. Sengupta, S. Ruj, and S. Das Bit, "A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT," J. Netw. Comput. Appl., vol. 149, Jan. 2020, Art. no. 102481.
[2] L. D. Xu, W. He, and S. Li, "Internet of Things in industries: A survey," IEEE Trans. Ind. Informat., vol. 10, no. 4, pp. 2233–2243, Nov. 2014.
[3] Y. Zhang and H.-Y. Wei, "Risk-aware cloud-edge computing framework for delay-sensitive industrial IoTs," IEEE Trans. Netw. Service Manag., vol. 18, no. 3, pp. 2659–2671, Sep. 2021.
[4] J. Sengupta, S. Ruj, and S. D. Bit, "FairShare: Blockchain enabled fair, accountable and secure data sharing for industrial IoT," IEEE Trans. Netw. Service Manag., vol. 20, no. 3, pp. 2929–2941, Sep. 2023.
[5] M. Aazam, S. Zeadally, and K. A. Harras, "Fog computing architecture, evaluation, and future research directions," IEEE Commun. Mag., vol. 56, no. 5, pp. 46–52, May 2018.
[6] K. A. Nuaimi, N. Mohamed, M. A. Nuaimi, and J. Al-Jaroodi, "A survey of load balancing in cloud computing: Challenges and algorithms," in Proc. 2nd Symp. Netw. Cloud Comput. Appl., Dec. 2012, pp. 137–142.
[7] C. Boudagdigue, A. Benslimane, A. Kobbane, and J. Liu, "Trust management in industrial Internet of Things," IEEE Trans. Inf. Forensics Security, vol. 15, pp. 3667–3682, 2020.
[8] S. Weinberger, "Is this the start of cyberwarfare? Last year's Stuxnet virus attack represented a new kind of threat to critical infrastructure,"Nature, vol. 474, no. 7350, pp. 142–146, 2011.
[9] H. Wang, D. He, J. Yu, and Z. Wang, "Incentive and unconditionally anonymous identity-based public provable data possession," IEEE Trans. Services Comput., vol. 12, no. 5, pp. 824–835, Sep./Oct. 2019.
[10] H. Jin, H. Jiang, and K. Zhou, "Dynamic and public auditing with fair arbitration for cloud data," IEEE Trans. Cloud Comput., vol. 6, no. 3, pp. 680–693, Jul./Sep. 2018.
[11] B. Sengupta and S. Ruj, "Efficient proofs of retrievability with public verifiability for dynamic cloud storage," IEEE Trans. Cloud Comput., vol. 8, no. 1, pp. 138–151, Jan. 2020.
[12] H. Yu, Z. Yang, S. Tu, M. Waqas, and H. Liu, "Blockchain-based offline auditing for the cloud in vehicular networks," IEEE Trans. Netw. Service Manag., vol. 19, no. 3, pp. 2944–2956, Sep. 2022.
[13] T. Wu, G. Yang, Y. Mu, F. Guo, and R. H. Deng, "Privacy-preserving proof of storage for the pay-as-you-go business model," IEEE Trans. Dependable Secure Comput., vol. 18, no. 2, pp. 563–575, Mar. 2021.
[14] J. Sengupta, S. Ruj, and S. D. Bit, "A secure fog-based architecture for industrial Internet of Things and Industry 4.0," IEEE Trans. Ind. Informat., vol. 17, no. 4, pp. 2316–2324, Apr. 2021.
[15] Y. Wu, Z. Wang, Y. Ma, and V. C. M. Leung, "Deep reinforcement learning for blockchain in industrial IoT: A survey," Comput. Netw., vol. 191, May 2021, Art. no. 108004.
[16] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, opportunities, and directions," IEEE Trans. Ind. Informat., vol. 14, no. 11, pp. 4724–4734, Nov. 2018.
[17] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial Internet of Things and Industry 4.0," IEEE Trans. Ind. Informat., vol. 14, no. 10, pp. 4674–4682, Oct. 2018.
[18] J. Shen, J. Shen, X. Chen, X. Huang, and W. Susilo, "An efficient public auditing protocol with novel dynamic ructure for cloud data," IEEE Trans. Inf. Forensics Security, vol. 12, no. 10, pp. 2402–2415, Oct. 2017.
[19] R. Chen, Y. Li, Y. Yu, H. Li, X. Chen, and W. Susilo, "Blockchainbased dynamic provable data possession for smart cities," IEEE Internet Things J., vol. 7, no. 5, pp. 4143–4154, May 2020.
[20] Y. Du, H. Duan, A. Zhou, C. Wang, M. H. Au, and Q. Wang, "Towards privacy-assured and lightweight on-chain auditing of decentralized storage," in Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS), Nov. 2020, pp. 201–211.
[21] Y. Du, H. Duan, A. Zhou, C. Wang, M. H. Au, and Q. Wang, "Enabling secure and efficient decentralized storage auditing with blockchain," IEEE Trans. Dependable Secure Comput., vol. 19, no. 5, pp. 3038–3054, Sep. 2022.
[22] P. Li, Y. Cheng, and F. Tao, "Failures detection and cascading analysis of manufacturing services collaboration toward industrial internet platforms," J. Manuf. Syst., vol. 57, pp. 169–181, Oct. 2020.
[23] F. Adamsky et al., "Integrated protection of industrial control systems from cyber-attacks: The ATENA approach," Int. J. Crit. Infrastruct. Protection, vol. 21, pp. 72–82, Jun. 2018.
[24] C. Adaros Boye, P. Kearney, and M. Josephs, "Cyber-risks in the industrial Internet of Things (IIoT): Towards a method for continuous

ISSN 2347–3657
Volume 13, Issue 2, 2025

assessment," in Proc. Int. Conf. Inf. Secur. Guildford, U.K.: Springer, 2018, pp. 502–519.
[25] J. Wu, J. Chen, W. Liu, Y. Liu, C. Liang, and M. Cao, "A calibrated individual semantic based failure mode and effect analysis and its application in industrial internet platform," Mathematics, vol. 10, no. 14, p. 2492, Jul. 2022.
[26] W. Chonghua, L. Jun, and C. Xuehong, "Research on industrial internet platform security protection," Netinfo Secur., no. 9, pp. 6–10, 2019.
[27] A. Liu, Q. Zhang, Z. Li, Y.-J. Choi, J. Li, and N. Komuro, "A green and reliable communication modeling for industrial Internet of Things," Comput. Electr. Eng., vol. 58, pp. 364–381, Feb. 2017.
[28] M. Ma, D. He, N. Kumar, K. R. Choo, and J. Chen, "Certificateless searchable public key encryption scheme for industrial Internet of Things," IEEE Trans. Ind. Informat., vol. 14, no. 2, pp. 759–767, Feb. 2018.