Une approche cryptographique pour l'authentification de données multimédia dans l'Internet des Objets basée sur les blockchains

Bismark Tei Asare

DOCTORAT BRETAGNE LOIRE MATHSTIC

Université de Bretagne Occidentale

THESE DE DOCTORAT EN COTUTELLE INTERNATIONALE DE

L'UNIVERSITE
DE BRETAGNE OCCIDENTALE, FRANCE
ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

ET

GHANA COMMUNICATION TECHNOLOGY UNIVERSITY, GHANA
GCTU GRADUATE SCHOOL
Computer Science Department

Par
Bismark Tei ASARE

A cryptographic technique for authentication of multimedia data in internet-of-things using blockchain

Thèse présentée et soutenue à Brest, France, le 25 November 2022
Unité de recherche : Lab-STICC (UMR, CNRS 6285), UBO, Brest - France
Préparée au CRITAC, GCTU, Ghana

Rapporteurs avant soutenance :

Ismail BISKRI	Professeur, Université du Québec à Trois Rivières
Koudjo Mawuefam KOUMADI	Professeur, Prince George's Community College

Composition du Jury :

Ismaïl BISKRI	Professeur, Université du Québec à Trois Rivières / Rapporteur
Koudjo Mawuefam KOUMADI	Professeur, Prince George's Community College / Rapporteur
Gouenou COATRIEUX	Professeur, IMT-Atlantique / Président du jury
Anca Christine PASCU	Maître de Conférences HDR Emérite, Université de Bretagne Occidentale / Examinatrice
Laurent Tchamnda NANA	Professeur, Université de Bretagne Occidentale / Directeur de thèse
Kester QUIST-APHETSI	Professeur, Ghana Communication Technology University / Co-directeur de thèse

Abstract

Authentication methods exist to allow devices and users to be validated before granting the devices and users access to available resources on a network. From classical username and password authentication credentials to the advanced forms of identification and validation of devices and users, authentication techniques continue to be made less effective as available hacking techniques and platforms become progressively complex. High profile cybersecurity attacks on networks and the occurrences of critical security breaches on existing wireless sensor networks grow by the day. From small-office-home-office network to Government databases, the incidences of cybersecurity breaches on these networks have risen sharply in recent times. The distributed ledger technology provided for by blockchain with all its benefits has shown great industrial prospects in the face of the rampant occurrences of cyber thefts of critical data from compromised networks. In the past decades, the severity of cyber-attacks on databases ranging from small to large networks for individual, corporate and Governmental databases have taken a worrying trend. A classical IoT network involves multiple subsystems that heterogeneously connect with each other providing an interface for integrating with the other subsystem. In such cases, interactions such as machine-to-machine, machine-to-environment, and machine-to-human communications may exist. A crucial challenge in such communication is a security mechanism that assures integrity and confidentiality of the devices as well as data in the IoT network. The advent of stable connectivity, advanced artificial intelligence approaches and expanded access for digital inclusion have led to the development of disruptive technologies that required individual, and corporate users of these disruptive technologies to have a digital identity to enable them use and explore the full potentials of these technologies that have become a commonplace where users alike engage to share data. Since the last decade, the pace of the development of these

disruptive technological platforms and services have made it possible for existing classical technologies to improve their applications by adapting different mechanisms geared towards making them relevant and attractive for a wider coverage. The inception of the internet of things have made interesting use cases for the internet as an enabler for the collection and processing of data from environments that were difficult and unsuitable for data collection and processing using traditional means. The internet of things allows subsystems to connect, collect and share data across nodes using the internet. Database and network identity models for connecting end devices towards the collection and aggregation of critical data within internet of things architectures have been in use since its inception. The classical centralized identity model allowed entities and identity fragments across multiple subsystems to be authenticated from a central location. This centralized identity management model allows user identities and data collection using a central dedicated system. The centralized identity model relies on centralized authentication mechanisms that creates a single point of failure for such database systems. Again, the centralized data is a honeypot for cyber-attacks. Although there exist security solutions for authenticating node data within internet of things systems, these security mechanisms are not as efficient and appropriate for deployment in environments where the devices have memory, space, and computational processing constraints. Blockchain as a database structure adopts an authentication mechanism that depend on cryptographic primitives and the distributed ledger technology to eliminate the incidences of single point of failure. Although Blockchain holds security benefits for ensuring enhanced privacy and confidentiality for stored data, the computational overhead in classical blockchain technology makes it a challenge for full use in a network where its devices are resource constrained. In recent times, the level of data breaches and cyber-attacks on wireless sensor networks have heightened the need for improved security mechanism for improving existing solutions to assure data integrity and availability. Additionally, the

absence of a standardization for communication, architecture, and security mechanism for blockchain-based security approach in the context of internet of things have created a gap in practice as well as theory for a security framework that adopted blockchain tailored for authenticating node data in IoT. This thesis seeks to adopt a blockchain-based cryptographic mechanism for authentication of node data for internet of things. The security solution proposed adopted the use of digital signature schemes appropriate for ensuring integrity for node data that involved an architecture for a cyber-physical node-data communication. To achieve security and integrity for transmission of data across nodes within internet of things environment the Edwards-curve digital Signature Algorithm (EdDSA) is employed to provide high-speed transaction in processing hashing and digital schemes for verifying messages among communicating nodes. The use of the distributed ledger technology ensured a distributed storage of node data between IoT gateway and the cloud. The system ensured decentralized authentication, portable identity across different subsystems, and the provision of a mechanism that assured enhanced security, privacy, and availability of node data. The Colored Petri Net (CPN) was employed for the modelling, simulation, analysis, and verification of properties of the proposed system.

Key words: Internet of Things, Blockchain, Public-Key Cryptographic, Authentication.

Résumé

Les cyberattaques hautement sophistiquées et les occurrences de failles de sécurité sur l'IoT (Internet des Objets) croissent de jour en jour. La technologie des blockchains de par ses atouts, a suscité un grand intérêt de la part des industriels face aux cyberattaques visant ces systèmes. Contrairement au modèle classique d'identité basé sur un système central dédié qui a l'inconvénient d'être un point singulier de défaillance, elle offre un mécanisme d'authentification basé sur une technologie de registres distribués évitant d'avoir un point singulier de défaillance. Malgré les atouts des blockchains, les solutions existantes ne prennent pas suffisamment en compte certaines spécificités de l'IoT telles que les contraintes en termes de capacités de stockage et de calculs. Par ailleurs, les primitives cryptographiques utilisées n'offrent pas le niveau de sécurité nécessaire. Dans cette thèse, des solutions ont été proposées pour répondre à ces problèmes. Ses principales contributions peuvent se décliner en 3 volets principaux: la proposition d'une architecture basée sur la blockchain pour la sécurité de l'IoT, la proposition d'approches cryptographiques pour le renforcement de la sécurité dans les architectures basées sur la blockchain pour l'IoT et, l'élaboration d'un modèle basé sur une approche formelle, à savoir les réseaux de Petri, pour la simulation, l'analyse et la vérification d'architectures basées sur la blockchain pour l'IoT.

Mots clés : Internet des Objets (IoT), Blockchain, Cryptographie à clé publique, Authentification, Modélisation, Réseaux de Petri.

Acknowledgement

Foremost, I thank Almighty God for giving me the grace and strength to complete this thesis. Several people have played critical roles by extending support and assistance in diverse ways towards the completion of this thesis.

My sincere and immense appreciation goes to my thesis directors Professor Laurent NANA of the UBO, Professor Kester QUIST-APHETSI of CRITAC - GCTU for their unique patience and academic tutelage throughout the research work which spans from the PhD title selection to the conclusion of the work. Their immense knowledge, motivation and forbearance have given me the best PhD experience in conducting my research. I would like to equally thank the members of my dissertation monitoring committee for their time, research direction and contributions towards the thesis.

I would like also to thank the Lab-STICC council, UBO Computer Science department, and the management and leadership of CRITAC - GCTU for granting me access to the laboratory and research facilities within their research laboratories. Without their support, it would not have been possible to conduct this research.

Last but not the least, I would like to thank my family, friends, and lab colleagues for their diverse support especially during the difficult times of the PhD journey.

Dedication

This work is dedicated to my Parents: Mr. Daniel Asare & Mrs. Victoria Asare in honor of their continued sacrifices and investments into my education and career.

List of Figures

List of Tables

List of Algorithms

Table of Contents

Chapter 1: Introduction

1.1 General Context

The world over, organisations and Governments have been increasingly digitizing and automating the manual ways of doing things. In recent times the use of disruptive technologies towards aggregating information and transmitting same across devices, platforms and networks have become a common place in academic, business, and Government databases systems [1]. The size of computerized systems keeps expanding rapidly to generate and store high volume of data. The cloud technology holds a lot of prospects for the remote storage of critical data [2]. Different sectors of a country's economy ranging from agriculture, healthcare, sports and entertainment, education, business enterprises, and commerce have all resorted to creating a strong digital presence for their products and services. This digitization has demanded that traditional network architectures and their underlying technologies be expanded to include innovative ways of connecting to collect critical data for processing towards helping the organization make relevant decision that will make them have the competitive edge to differentiate them form their peers. The internet of things holds the potential of connecting ordinary devices and objects together to aggregate data about the physical measurements of the environments and machinery particularly in confined spaces to help manage and maintain these environments.

A picture can summarize and represent a story that consists of several lines and pages of words. The availability of sophisticated and yet averagely priced smart phones has made the use of pictures and multimedia data communication a common place in recent years. The growth in available social media applications has played a part in the widespread use for multimedia data. The internet of things (IoT) has become a network of choice owing to the numerous benefits that it provides. The security challenges of IoT networks in recent times

have attracted significant efforts from academia, Government and the business community towards a solution that will provide overall security solution to address the security loopholes in IoT to help maintain the relevance and usefulness of this pervasive technology. About 75 billion IoT connected devices is expected to be in use by the year 2025 [3].

The spike in the number of digital presences has created more avenues for cybercriminals and their related crimes to increase. The incidences of cyber-attacks from academia, industry and Government installations have increased sharply since the last decade. Database systems from transport, healthcare, financial services, energy, manufacturing, as well as entertainment sectors have been predisposed to severe data breaches that have resulted in data theft, disruptions of business process, financial losses due to cyber-attacks on their critical information technology infrastructure. The past decade has seen great research activities from the perspectives of academic research and practitioner investments on innovative and efficient security mechanisms for ensuring a cyber-hygiene ecosystem to assure the privacy, security, and availability of node data within a wireless sensor network. Blockchain technology provides a security framework in authentication of transactions. The framework assures security, privacy, and integrity of data. These security properties of blockchain make the technology a suitable option to be included in the design and implementation of a security mechanism for securing data in internet of things [4].

This thesis seeks to design a blockchain-based cryptographic authentication mechanism for validating multimedia data in an internet of things environment.

The other subsections of this chapter are the following:

- Background of Internet of Things
- Problem Statement
- Thesis Objectives
- Contributions

- Organisation of the Thesis

1.2 Background of IoT

Internet of things end devices are designed to constantly record, collect, analyse and communicate data for ubiquitous operations for long hours notwithstanding their weak design flaws due to lack of design standardization makes these sensors and other IoT end devices susceptible for targeted cyber-attacks since these end devices act as weak entry points and honeypots for compromising the security of the large systems that they are a part of [5].

Connectivity and internet protocols enable sensors to collect and communicate information from their immediate surroundings with other sensors, actuators, and diodes through an intermediary device like the sink node. Internet of things could be defined as the phenomenon concerning extending connectivity capabilities of connecting ordinary objects with the necessary electronic capabilities to connect to the internet to facilitate communication and resource sharing of resources. Aside the numerous benefits that IoT brings, there is growing challenges with creating secure and private infrastructure to support reliable and secured communication as far as internet communication among these tiny and resource constrained devices are concerned.

Wireless sensor networks have been at the forefront of the pervasive technology innovations: The Internet of things (IoT) has supported the seamless transmission of messages across several devices that span the internet. Every communication system must ensure and support the unique qualities of confidentiality, authentication, integrity, and non-repudiation of network operations. In recent times, the spate and the volume of data security comprises in IoT systems have taking an alarming rate and that has translated into a heightened research interest and investments across academia, Government, and industry in addressing the security lapses in these networks [6]. An IoT system involves heterogeneous platforms,

operational environments, and end devices that collect critical data of physical measurements and collaborate with other end devices and platforms for the communication of the collected data using the internet as a backbone for such communication. A modern IoT architecture consists of various connection points such as the sensing and embedding components, connectivity component, IoT cloud component, IoT analytics and data management component, end-user devices and user interfaces. These connection points for the component increases the attack surface for the IoT framework. A weak subsystem within the framework does not only pose a threat to itself but the entire IoT network. A compromise on any device or element within a subsystem of the IoT architecture will result in the total compromise of the system with tendencies including data corruption, data theft, data leakages and loss of connectivity.

The proliferation of smart phones and the rapid production of social media channels in recent times have also placed additional security challenge since most of these media channels integrate and operate based on IoT concepts. The number of subscribers and active participants on these channels increases sharply by the day. There exist various cryptographic mechanisms that assure the security of messages by enforcing confidentiality, integrity, authentication, and non-repudiation of data transmission. Although there are several cryptographic encryption schemes available and in use for protecting the privacy of messages, the availability of sophisticated tools, applications, and high-end devices in the hands of hackers makes it insufficient to adopt only cryptographic encryption schemes to provide privacy and integrity of messages in a database system. Blockchain technology is operated and maintained using consensus mechanisms for decentralized processing and distributed storage that depend on smart contract concepts for the transmission of authenticated messages among active nodes in blockchain network. Message authentication is achieved using hashing functions and digital signature schemes. The strength of a hash

function guarantees the security of the message against disclosure compromises particularly in a brute force attack such as the dictionary attack.

1.3 Problem Statement

Several research on multimedia data authentication schemes have been conducted in recent times [7]. Multimedia communication within internet of things networks and its attendant security challenges have become an area of research concern to authorities in the blockchain ecosystem. Several individual attempts and research have been made in the areas of cryptography and blockchains and authentication techniques for data communication but there has not been a deliberate attempt to use cryptographic technique to authenticate multimedia data in internet of things using blockchain-based authentication mechanism for modern IoT architecture that includes a cloud component [8] [9]. Not much research has been conducted around cryptographic authentication techniques in the context of an IoT data communication between sensor end devices, the sink node, and the cloud.

An IoT heterogeneous system involving three subsystems provides a challenge in authenticating the node data as it transmits from the sensor to the cloud. In such a heterogeneous system, it is possible that each subsystem deploys a unique security approach to provide the needed security protection against threats that might compromise the privacy and integrity of data. Since the subsystems of the heterogeneous system must integrate to interact with each other, the absence of a weak system-level security framework to provide and support a secured interface for the physical and cyber subsystems to connect and share data will compromise the security of the entire system. The entry points for the connection to interface and integrate the subsystems together might introduce a security loophole where an external attacker could use to compromise the entire system. At the heart of the security challenges that heterogeneous systems are predisposed to are, impersonation, spoofing, and

eavesdropping attacks. A sufficiently secure internet of things system must provide protection that assure that devices are authenticated, data is secured through a strong encryption for the assurance of data provenance. An introduction of a blockchain overlay network implemented between a physical (on-site Local IoT network) and a remote (cloud network) involving blockchain consensus mechanism and distributed ledger will provide secure interface connection to authenticate IoT data for storage in a peer-to-peer network between IoT gateway and the cloud storage.

1.4 Thesis Objectives

The objectives of this thesis are to design, model and validate blockchain-based authentication mechanism for ensuring message integrity in internet of things. The motivation for undertaking this research is stemmed from the fact that although there are abundant literature and works to suggest that blockchain provided adequate security for data in the form of digital assets, the incidences of data breaches and compromises on wireless sensor networks continue to rise sharply. There seems to be an implementation gap on the possibilities of adopting blockchain technology to improve node data authentication to assure secure and integrity in a modern IoT framework. The security challenges of internet of things and the implementation gap for a blockchain-based authentication mechanism is supported in the state-of-the-art section of this documentation. There is an absence of a cryptographic security solution that adopt the use of blockchain for the authentication of messages in the context of node data in internet of things framework that involves the cloud as an active architectural component.

Research Question:

Towards providing a solid background and justification for undertaking the research, a set of relevant question were asked to provide guidance in the form of research objectives.

i. To what extend has the predisposition of existing IoT network contributed to challenges with node data security, privacy, and integrity compromises?

ii. What are the available lightweight cryptographic schemes in validating IoT devices whiles authentication of node data?

iii. How can IoT architecture be improved to support the implementation of blockchain-based technologies to authenticate node data?

iv. How can software application be used to model, simulate, and validate the security properties of a cryptographic schemes that uses blockchain distributed ledger technology and consensus mechanism to authenticate IoT data for storage on a distributed ledger?

The answers to the research questions are well presented in chapters two, three, four, and five. Thus, in the state of the art, proposed methodology, implementation features of the method of the system, and the validation of the proposed solution, respectively.

In the next subsection, the contribution of the thesis is outlined.

1.5 Contributions

The primary contribution of the thesis is the design of a blockchain-based IoT architecture for data authentication. The design of the architecture was based on a distributed ledger technology for a cyber-physical peer-to-peer security implementation that involved the IoT gateway sink node and the cloud storage.

The second contribution of the thesis is the design of lightweight cryptographic algorithm that is based on the Feistel structure for validating IoT node data.

The final contribution of the thesis is the formal modelling of a blockchain solution for IoT systems for node data authentication. The blockchain authentication mechanism adopted lightweight cryptographic schemes appropriate for satisfying the computational processing, storage, and energy requirements for the resource constrained devices in IoT. A blockchain is an append-only distributed ledger that is based on a distributed database to provide a transparent and permanent storage of an ordered set of transactional data. A blockchain depend on peer-to-peer network with decentralized authentication mechanisms that is based on consensus approaches to eliminate instances of a single point of failure. All active nodes constituting consensus peers retains a copy of the simultaneously validated and updated ledger [7]. The adoption and use of IoT by small and large enterprise networks which is evidenced by the massive investments in budget allocations in the recent past, is an indicative gesture of the acceptance of the benefits that IoT offers in scaling up the scope and coverage of these classical network operations, although there seem to be a parallel match in the rise of IoT network compromises and cyber-attacks on IoT data around the same period. Existing classical centralized IoT architectures serve as a major obstacle to adopting blockchain-based solution . Similarly, since blockchain implementation is based on peer-to-peer networking with heavy computational overhead, directly adopting the authentication approach used in blockchain for existing classical IoT architecture where the main actors have energy, storage and computational processing constrains presents a challenge for a smooth and efficient implementation.

The contributions outlined in this thesis can be summarized as follows:

1.Reviewing of existing cryptographic security mechanisms that are efficient and yet lightweight in nature for IoT node data security implementation. The use of cryptographic primitives was employed in providing improved security that assure privacy, security, and integrity of node data. Symmetric and asymmetric encryption algorithms were explored and used. Analysis of existing security solution were conducted and synthesized in the broad context of the resource constrained nature of IoT nodes.

2. Designing, implementing, and evaluating lightweight authentication methods for Internet of Things towards providing node-node security. We provided cryptographic techniques that assure node data privacy and integrity using encryption algorithms and hash functions. The results of such implementation were published.

3. Designing, implementing, and analysing cryptographic primitives based on the decentralized authentication approaches that is based encryption algorithms and digital signature schemes for authenticating IoT devices and validating node data . The findings of the implementation were published and shared with the scientific community.

4. Designing, implementing, and evaluating IoT architecture that involve physical network, cyber or remote network and virtual network that interface the physical (on-site) and the remote (off-site) networks and run blockchain-based IoT data authentication mechanism to eliminate incidences of a single point of failure to assure data privacy, security, and data availability. The findings from the design and implementations were published and shared with the scientific community.

5. Formal Modeling of a blockchain-based solution for IoT data authentication for distributed node data storage between a physical network (on-premises local IoT gateway persistent storage) and the remote network (off-site storage) cloud storage. The CPN tools was used for validating the security properties in the proposed blockchain-based IoT data authentication mechanism.

1.6 Organisation of the Document

The thesis is structured into two distinct parts: part I and part II. The entire work is organized into chapters. There are six chapters in all. Chapter 1 and chapter 2 constitute part I of the work. Part II is composed of chapter 3, chapter 4, chapter 5, and chapter 6. Chapter 1 introduces the key components and concepts that forms the background of the work. Chapter 1 describes the background information including the definition of concepts, the problem statement, the objectives, and the expected results of the work. Chapter 2 contains the state-of-the-art that outlines the review of related works to identify a gap and justify the relevance of the research. The initial part of chapter 2 considered the state-of-the-art in cryptology, internet of things, blockchain, and message authentication for constrained devices. Reviews were conducted to demonstrate trends in these areas from the classical to the post-modern phase of cryptology. The part II which begins with chapter 3 covers the contribution in the design of blockchain-based IoT data authentication architecture for ensuring message integrity regarding node-to-node communication in internet of things network particularly in sending data from the sink node to the cloud. It follows with chapter 4 that describes the implementation features of the blockchain-based architecture in authentication of node data. Chapter 5 describes the validation of the proposed system using the Coloured Petri Net.

Finally, chapter 6 concludes the research with propositions for future work.

PART I : State of the Art

Chapter 2: Trends in Cryptology and Blockchain Applications to Internet of Things Data Authentication

2.1 Introduction

Over the past decade, there have been a surge and widespread internet security infractions on multimedia data that have resulted in compromising the vulnerabilities in the audio, video, and data storage sources. To provide a solid background to advance this research, a purposeful study of findings into the trends and themes of related works where the differences, commonalities, and nuances on Cryptographic techniques, hash functions, digital signature schemes, blockchain, and IoT data authentication mechanisms is conducted. This section of the thesis covers the review of related works relevant to the goal of this research. The report for this chapter of the thesis constitutes the state-of-the-art and forms the basis for the introduction of the contribution for the thesis.

Although classical public-key cryptographic solutions keep undergoing improvements to make them resistive by increasing the difficulty of compromise in cryptanalysis, the deployment of quantum computers and the availability of complex computing systems raises higher concerns for a cryptographic technique that is efficient and appropriate for wireless networks where the sensors, being the principal actors work with power, computational, and memory capability constrains [10]. Database systems suffer from several challenges due to several factors including eavesdropping attacks that compromise the integrity of the stored data. This part of the thesis covers update on theory, technology, practice, protocols, and standards in cryptology and blockchain in authentication of data in internet of things network. With the advent of modern-day data streaming and social media websites, as well as the

availability of high-end smart devices, the spate and cost of system level security compromises have taken a worrying trend. Generating and communicating information among concerned parties is as old as the existence of man. Traditional communication approaches, technological communication mechanisms to advanced cyber-enabled communication, have all suffered from security challenges in one way or the other.

The category of state-of-the-art approach adopted for the thesis is an umbrella review that sourced and compiled evidence from varied researched works that highlighted interventions for addressing the peculiar security solution requirement of constrained devices and heterogeneously connected networks for internet of things. The searched works included studies on theoretical studies and implemented or practical applications and systems with improved security layers for providing security for node data in internet of things. The result from the search is synthesized in tabular form with narrative commentary. Recommendations for practice and recommendations for future works are provided at the end of this chapter.

2.2 Classical Cryptography

Cryptography involves using techniques and approaches for securing sensitive data from an unauthorized entity. It includes advanced mathematical functions and concepts in changing the nature and format of the sensitive data without affecting the content of the data. The primary goal of cryptography is to secure data (whether a data in transit or data at rest) by hiding the data from agent or attacker. Cryptography therefore allows an original message in plaintext to be converted to a ciphertext consisting of a fixed length of alphanumeric and special characters. The process of converting the plaintext to ciphertext is known as encryption. Thus, encryption involves encoding plaintext using a key such that only those with the access right or the corresponding key can decrypt and use the data. Although encryption does not prevent unauthorized interception, it protects the content of data.

There are two broad categories of cryptography. The categorization is based on the nature of the key used for encoding and decoding the data. One category of cryptography uses the same key for encrypting and decrypting data. The other category of cryptography uses different but related keys for encryption and decryption. The strength of a cryptographic scheme is depended on the size of the key, secured key scheduling and management, and the communication of the key. A loosely secured or communicated key could result in unauthorized users gaining access to the data. The cryptographic algorithm is publicly available hence the need for a secure key to be used to assure the security of the encryption. The security of a cryptographic system is as secure as the key used.

In a cryptographic operation, there exists several critical elements: Digital Signature, Encryption and Hashing. An equally essential component for encryption and decryption is a Key, and Salt. A cryptographic key can be described as a piece of data that will cause a plaintext to become a ciphertext.

Prior to the development and deployment of computing systems, classical ciphers were invented and used. The used of these classical ciphers were popular until the 1950's. Notable environments where the classical ciphers were popular included the military, colonial traditional kings, and governments for their top-level communication needs, For example BiFid, ADFGVX. The classical ciphers have properties including the following:

- They adopted substitution, transposition techniques
- They were used on plain messages involving alphabets, where the original alphabets making the messages were replaced or interchanged with one another.
- The transposition ciphers involved rearranging the alphabets in a unique order.
- They were designed using either monoalphabetic or polyalphabetic methods.

The Bifid cipher involved two approaches to achieve diffusion. By design, it used Polybius square with transposition together with fractionation. Felix Delastelle invented it [11] [12].

According to sources, the Bifid did not feature in popular applications outside of its classified use for military top-level communications .

2.2.1 The Khnumhotep

The Khnumhotep cipher was discovered on the tomb of an Egyptian noble man Khnumhotep II some 4000 years ago. Egyptian historic records which existed about 1900 B.C suggests that an inscription on tomb for Khnumhotep II in the town called Menet Khufu consisted of cryptography. The cipher is operated based on the substitution cipher methodology whereby one symbol within a text is substituted for another symbol [13].

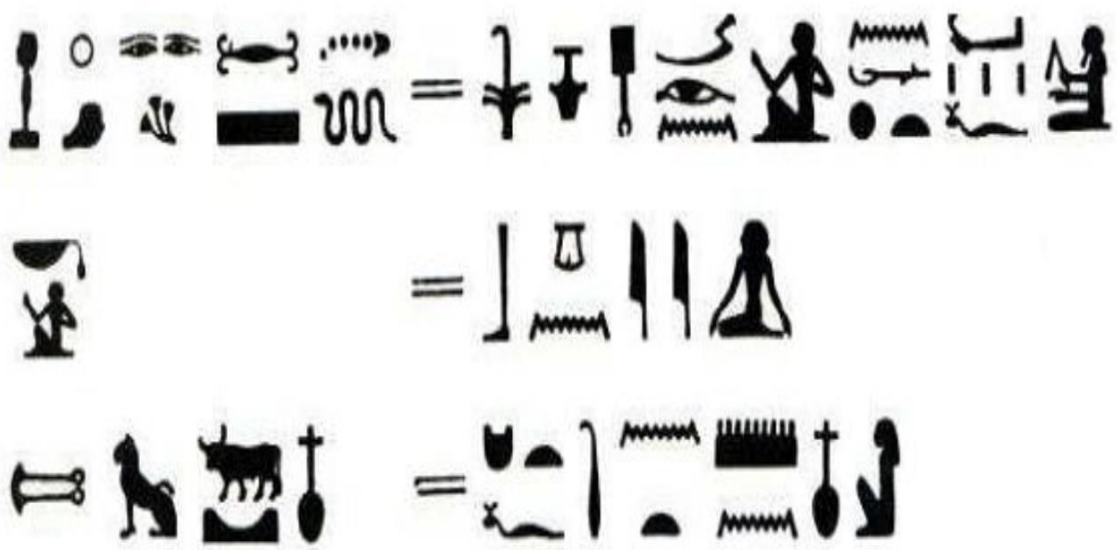

Figure 1 Symbols taken from the tomb of Khnumhotep II

In Figure *1*, the various symbols and totems used for the Khnumhotep II are presented. These symbols were used to inscribe on the tomb of the Khnumhotep II.

2.2.2 The Spartan Scytale

The spartan scytale was one of the popular encryption tools used by the Spartan Military for encoding messages sent between commanders during military campaigns and battles. The device consisted of a ribbon around a dowel which is cylindrical in shape and wrapped of a particular diameter and length for effective coding. An identical dowel was needed by the recipient of the message to decrypt the message [14]. The message to be communicated was written on the ribbon around the dowel and transmitted. The Scytale is one of the initial transposition ciphers [15]. The size of a message to be written on the Scytale is determined by the diameter of the cylinder of the Scytale on which the message will be wrapped. The wrap factor controlled the size of letters to be written in a column. The wrap factor formed the key to the message. Different rods will have dissimilar diameter. A different diameter for dowel will constitute a different secret key. The scytale cipher was a very sophisticated cryptographic scheme used by the ancient Greek Government to send critical mandates or messages to its soldiers [16] [17].

Figure 2: Scytale [17]

In Figure *2,* is a display of an image that contains some characters for the Spartan Scytale encryption tool.

2.2.3 Frequency Analysis

Frequency analysis concerns itself with the examination of the frequency of letters or groups of letters in a ciphertext. The approach is used to break substitution ciphers into blocks. It is popularly applied in mono-alphabetic substitution cipher, Caesar shift cipher, Vatsyayana cipher. Frequency analysis comprises the counting of the occurrence of each letter in a text. Frequency analysis method checks for the possibility of alphabets either as singular or double characters occurrences in a word. Some letters and combinations of letters occur with varying frequencies in a word. For example, whereas English characters such as Q, X and Z are seldomly used, other characters like A, E, T, O, I, and N tend to be frequently used. Frequency analysis is an important activity to check for the distribution pattern of reoccurring letters in a word. The occurrence value of the ciphered letters from the English alphabet and its frequency is plotted in a chat below. A bar chat graph of the substitution of the normal relative occurrence values of related frequencies in determining encrypted messages is provided [18].

Table 1: Percentage of Occurrence of the English Alphabet

Alphabet	Percentage of Occurrence (%)	Alphabet	Percentage of Occurrence (%)
A	8.17	N	6.75
B	1.49	O	7.51
C	2.78	P	1.93
D	4.25	Q	0.1
E	12.7	R	5.99

F	2.23	S	6.33
G	2.02	T	9.06
H	6.09	U	2.76
I	6.79	V	0.98
J	0.15	W	2.36
K	0.77	X	0.15
L	4.03	Y	1.97
M	2.41	Z	0.07

Table 1 shows the English alphabets with the percentage of occurrence of each alphabet. Among the vowels alphabets – a, e, i, o, u; the “e” alphabet has the highest percentage frequency of 12.7% whereas the “u” alphabet has a percentage frequency of 2.76%.

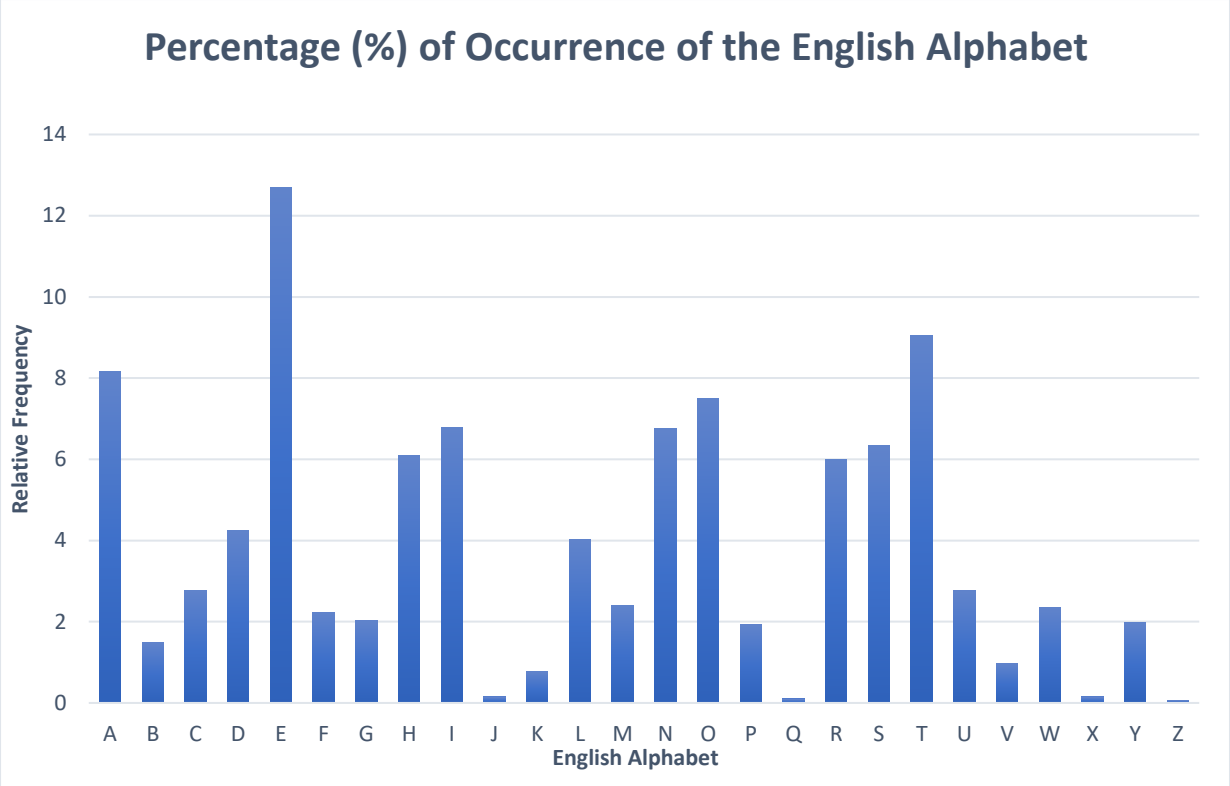

Figure 3: Percentage of Occurrence of the English Alphabet

In Figure 3 the bar chart representation of the percentage of occurrence of the English alphabet is presented. The relative frequency of occurrence is displayed on the y-axis whereas the characters of the English alphabet are shown on the x-axis.

2.2.4 The Playfair Cipher

The Playfair cipher is a block cipher that encrypts pairs of letters or digrams (bigrams) using a pre-shared keys to encrypt and decrypt numeric, letters and special characters based on polygram substitution. The pre-shared key consists of a table key matrix. The classical playfair cipher has a keyword constructed using a 5 x 5 matrix of letters.

[13] To encrypt a message, one would break the message into digrams (groups of 2 letters) such that, for example, "HelloWorld" becomes "HE LL OW OR LD". These digrams will be substituted using the key matrix table. Since encryption requires pairs of letters, messages with an odd number of characters usually append an uncommon letter, such as "X", to

complete the final digram. The two letters of the digram are considered opposite corners of a rectangle in the key table. To perform the substitution, apply the following 4 rules, in order, to each pair of letters in the plaintext:

i. If both letters are the same (or only one letter is left to complete a bigram), add an "X" after the first letter. Encrypt the new pair and continue. Some variants of Playfair use "Q" instead of "X", but any letter, itself uncommon as a repeated pair, will do.

ii. If the letters appear on the same row of your table, replace them with the letters to their immediate right respectively (wrapping around to the left side of the row if a letter in the original pair was on the right side of the row).

iii. If the letters appear on the same column of your table, replace them with the letters immediately below respectively (wrapping around to the top side of the column if a letter in the original pair was on the bottom side of the column).

iv. If the letters are not on the same row or column, replace them with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair. The order is important – the first letter of the encrypted pair is the one that lies on the same row as the first letter of the plaintext pair.

To decrypt, use the inverse of the last 3 rules, and the first as-is [19].

For Example, to encrypt the word " Hello World" and using the phrase "playfair example" as the key.

"Hello World" is broken down into digrams as: "HE LX LO WO RL DQ" to become "DM YR AN VQ LU GO". Thus, "Hello World - HELXLOWORLDQ" as a plaintext becomes "DMYRANVQLUGO" using the "playfair example" as the key.

Similarly, to encrypt the message “Hide the gold in the tree stump” using the key phrase “playfair example”.

The procedure in encrypting the message is:

i. The entire message is split into bigrams, thus pairs of letters. If the letters are identical or if there is only one character remaining, X or Q is inserted.

   HI DE TH EG OL DI NT HE TR EX ES TU MP

ii. If the two letters are on the same line, replace them by the ones on their right. If the edge of the grid is reached, loop to the left

iii. If the two letters are on the same column, replace them by the ones directly under them. If the bottom of the grid is reached, loop to the top.

iv. Replace the letters by the ones forming a rectangle with the original pair. The coded bigram begins with the letter on the same line as the first letter to cipher.

Table 2: Playfair Key Matrix

P	**L**	**A**	**Y**	**F**
I	**R**	**E**	**X**	**M**
B	**C**	**D**	**G**	**H**
K	**N**	**O**	**Q**	**S**
T	**U**	**V**	**W**	**Z**

Table *2* represents the playfair key matrix

Plaintext →”Hide the gold in the tree stump”

Bigram → “HI DE TH EG OL DI NT HE TR EX ES TU MP”

Polygram Substitution →

“HI - BM”,

“DE - OD”,

“TH - ZB”,

“EG - XD”,

“OL - NA”,

“DI - BE”,

“NT - KU”,

“HE - DM”,

“TR - UI”,

“EX - XM”,

“ES - MO”,

“TU - UV”,

“MP - IF”.

New Polygram → “BM OD ZB XD NA BE KU DM UI XM MO UV IF”

Ciphertext therefore is: “BMODZBXDNABEKUDMUIXMMOUVIF”

Modern Playfair cipher uses dynamic key matrix of size m x n. There are key matrix sizes consisting of 6 x 6 matrix, 8 x 8 matrix, and 16 x 16 matrix [20]. The size in the key matrix has been increased to reduce time complexity as well as improve the security of the playfair cipher to provide efficient privacy for messages. Several implementation improvements have been applied to the Playfair cipher in the use of the key matrix table. In [19] A matrix table key size of 16 x 16 incorporated with XOR, bit swapping to improve resistance against avalanche effect. The use of the combined key implementation provided improved protection against brute force attacks.

2.2.5 Vigenère Cipher

The Vigenère cipher was also known as $le\ chiffre\ indechiffrable$. The polyalphabetic cipher provides a strong resistance to frequency analysis and considered unbreakable for some 300 years. The Vigenère cipher's mode of encrypting alphabetic text is based on the way the Caesar cipher encoding of characters is done by using a chosen keyword which is also known as a key. Each letter in an information is shifted and replaced by a different letter based on the key involved. The key used in a Vigenère cipher is not constant for the letters within a particular information. Encoding letters in using this cryptographic approach involves using an encoding table that is known as tabula recta [21]. Algebraically, the cipher is described using the numeric representation of alphabets where the characters A, B, C, … , Z are denoted by 0, 1, 2, …, 25, respectively. Addition in Vigenère is performed in modulo 26.

Assuming, the key for the encryption is K, and the plaintext message to be encoded is M.

The key consists of characters.

$K \rightarrow K_i$, be the ith number in the Keyword K.

$M \rightarrow M_i$, be the ith number in the message M

Vigenère encryption E using the key K can be written as:

$\mathrm{C}_i = \mathrm{E}_K\,(M_i) = (\mathrm{M}_i + K_i) \pmod{26}$

The C_i is the ith number of the ciphertext.

The Decryption function D_K reinstates the plaintext from the ciphertext.

$M_i = D_K(C_i) = (C_i - K_i) \pmod{26}$

The plaintext message M, is encrypted using the keyword E_K to produce the ciphertext C. To decrypt the ciphertext (C), the decryption key D_K.

$$D_K\big(E_K(M_i)\big) = D_K(M_i + K_i) \text{ (mod 26)}$$

$$= (M_i + K_i) - K_i \text{ (mod 26)}$$

$$= M_i$$

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Figure 4: The Vigenère Square

In Figure 4, the diagram illustrates a symmetric table consisting of rows and columns. The table is used to encrypt each letter within a message after matching the row and column of the letter position for each character within the message against the corresponding character position in the keyword, respectively. For example, to encrypt the phrase “provethetheorem” using a keyword “GRAPE”. The procedure for the encryption is as follows.

The number of the characters in the keyword forms the size of the keyword. The size of the keyword is used to group the characters in the messages. In situations where the keyword is

different from the size of the message, the keyword is repeated character by character to match the size of the message.

The character “P” is encoded with a key “G” by looking at row P against column G in the symmetric table. The entry for the key at the intersection forms the ciphertext for that character.

Plaintext	prove	theth	eorem
Key	GRAPE	GRAPE	GRAPE
Ciphertext	VIOKI	ZYEIL	KFRTQ

Thus, the ciphertext for “provethetheorem” using the keyword “GRAPE” is “viokizyeilkfrtq”

The Vigenère cipher is not prone to frequency analysis attacks due to the approach used in encoding character in a message where each character in the message uses a contrasting character within keyword. The operation of encrypting messages is different from a substitution cipher but much closer to the procedure in Caesar cipher [22].

The Kasiski investigation on guessing the key size for a keyword in Vigenère was an attempt to compromise the cipher after predicting the key for encryption of messages. The examination is to identify recurrence of words or letters in the ciphertext. In [23] the Vigenère cipher and Goldbach codes algorithm were combined to provide enhanced data security. The key determination method by Kasiski was made less effective because the Goldbach code compression of the ciphertext produced different ciphertext letters from the original ciphertext. In [24] the procedure in the implementation of the classical Vigenère cipher was modified to improve the resistance against Kasiski and Friedman attacks in pattern prediction and cryptanalysis of estimating the key length for encryption.

2.2.6 Conclusion of Classical Ciphers

The procedure in classical ciphers is based on transposition and substitution methods. Due to high-end computational capabilities that modern computers possess, adopting classical ciphers as the only cryptographic approach is not a good measure since their security and resistance against attacks have proven over the years to be inefficient. However, they possess certain characteristics that make it a good component to be used in a hybrid cryptographic mechanism. The features of Classical ciphers therefore formed the basis on which modern ciphers were developed. Affine cipher, Atbash cipher, Baconian cipher, Codes and Nomenclators cipher, Four-Square cipher, Polybius Square cipher, Rail-fence cipher, ROT13, and Straddle Checkboard cipher are some more examples of classical ciphers [25].

2.3 Modern Ciphers

The developments in computing technology and the availability of advanced mathematical algorithms have resulted in improved cryptographic approaches that assure message secrecy, integrity, and security. These advanced mathematical algorithms have strengthened existing classical ciphers and, in some cases, pioneered new ciphers. The new ciphers have improved resistance against common attacks suffered by classical ciphers [26].

There are two main classifications in modern cryptographic encryption mechanisms. There is symmetric and asymmetric cryptography. The main factors on which modern cryptographic ciphers are categorized are the nature and type of key used, and the input method selection for the plaintext. To encrypt messages, keys are used. Decrypting ciphertext messages also involve using keys. When same key is used for encryption and decryption, the nature and procedure of the encryption is differentiated from the other kind of encryption where dissimilar keys are used.

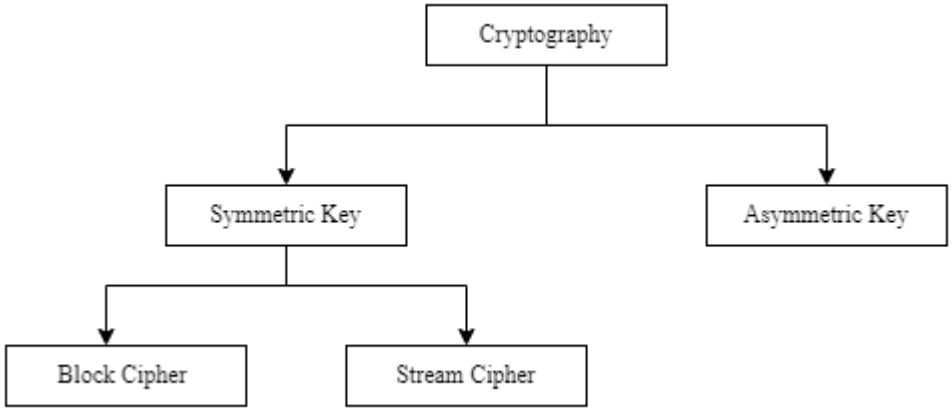

Figure 5: Modern Cipher Classification

In Figure *5*, the two classes of modern cryptographic approaches are presented. Symmetric and Asymmetric ciphers. Symmetric ciphers are grouped into block ciphers and stream ciphers.

Symmetric key algorithms are also referred to as Private-key cryptography. Example of symmetric key algorithms are Data Encryption Standard (DES), Advanced Encryption Standard (AES).

Asymmetric key algorithms are also known as Public-key cryptography. Examples of Asymmetric key algorithms are Rivest-Shamir-Adleman (RSA), Elliptic Curve Cryptography (ECC), Digital Signature Algorithm (DSA), Diffie-Hellman.

In Symmetric key algorithm, both the sender and receiver use same keys to encrypt the message and decrypt the ciphertext messages, respectively. On the other hand, Asymmetric ciphers involve using a key pair to encrypt the message and decrypt the ciphertext. Thus, in asymmetric key algorithms, both the private key of the sender and the public key of the receiver are used to encrypt messages. To decrypt ciphertext, the public key of the sender and the private key of the receiver are used in the process.

Block ciphers convert plaintext data in block into ciphertext in fixed-size blocks. The block size is usually measured in octaves. The encryption scheme in use determines the block size.

In situations where the input size or length of the plaintext is less than multiples of eight(8), the message is padded to ensure complete block formation. For example an encryption scheme that uses 64-bits encryption and yet must work on a 220-bits plaintext, will have 3 of the 64-bit blocks as input plaintext, the remaining 28-bit must will be padded with additional 36-bit to enable it satisfy the bit-size requirement compliance for a plaintext to be worked by the 64-bit encryption scheme [27].

Commonly used block ciphers include Data Encryption Standards (DES), Advanced Encryption Standard (AES), Twofish, Triple Data Encryption Algorithm (3DES), Serpent, and Blowfish.

Contrastingly, a stream cipher encrypts a continuous string of binary digits as plaintext and transforms the plaintext using time varying transformation function. Examples of symmetric stream ciphers are Rivest Cipher (RC4), Software-optimized Encryption Algorithm (SEAL), Grain, Scream, PANAMA, Rabbit, HC-256.

2.3.1 Symmetric Ciphers

Symmetric ciphers are also known as same-key cryptography. It requires the use of a shared secret to encrypt and decrypt messages between two entities. The cipher algorithm using a one-time pad (OTP) system allows for a generated random key to be used once to encrypt a message by a sending node and then decrypted by a receiving node using a matching one-time pad and a single-use pre-shared key. The use of the one-time pad technique in encrypting messages prevents unauthorized access by third parties to manipulate the message. Only nodes that have the pre-shared secret can decrypt the ciphertext and use the message.

Classical ciphers operate based on symmetric cryptographic mechanisms.

2.3.2 The AES-Advanced Encryption Standard

The Advanced Encryption Standard consists of the AES-128, AES-192, and AES-256 variant block ciphers. The AES-128 takes an input plaintext of 128-bit message. Similarly, the AES-192 and AES-256 takes an input plaintext message of message length 192-bits and 256-bits, respectively. In all the variants of AES, both the transmitter and the receiver mush share and use the same key for encryption and decryption of the message.

2.3.3 Key schedule

Key exchange protocols allow entities or communicating nodes who do not have a prior knowledge and trust for each other to undertake a cryptographic activity by providing a mechanism to generate and schedule all the session-keys that will be required for encrypting and decrypting of messages. They are used in calculating all the keys from an initial given key to produce the round keys needed in undertaking a cryptographic operation. The key schedule algorithms produce a relatively substantial and advanced keys from a relatively smaller-sized master key [28]. Key schedule algorithms are useful in agreeing on the round keys that are adopted for product ciphers. Similarly, Key schedule algorithms aid in the initialization of fixed elements of a cryptographic transformation. Additionally, key schedule algorithms are used for starting the state of the stream cipher prior to keystream generation. There are several key schedule algorithms in used today. There are Software-optimized Encryption Algorithm (SEAL), Blowfish, Data Encryption Standard (DES), LOKI, Tiny Encryption Algorithm (TEA), Rivest Cipher (RC with all the variants)

A weak key used in a cryptographic operation will affect the overall security robustness of the cryptographic cipher.

2.3.4 The Feistel Cipher

Block ciphers are developed based on the symmetric structure in Feistel ciphers. A Feistel cipher provides the model on which block ciphers are constructed. It presents critical components for the construction of these ciphers based on the unique attributes of invertible, non-invertible, and self-invertible. Feistel based ciphers operate in rounds of steps where keys are required for each round of the cryptographic operation. Whether in an encryption or decryption operation, the many rounds involved in this operation depend on unique keys for every round involved in the operation. A series of operations that are linked together, form a round.

In [29] the Feistel cipher was adopted in securing node data in a node-node communication within an IoT system. The Race Integrity Primitives Evaluation Message Digest (RIPEMD-128) was used to provide a checksum for the secured data to assure data integrity and detect corruption of the communicated data. The Twofish cryptographic cipher provided a secure IoT data by using the unique sink node attributes to develop a symmetric secure key that guarantee an enhanced node exchange in the IoT system, the Diffie-Hellman key exchange protocol assisted in the exchange of keys between the communicating nodes [30]. The symmetric key structure in Feistel cipher becomes an available option for efficient cryptographic scheme for wireless sensor networks where most of its devices have energy and computational constrains challenges. The Feistel ciphers ensure adequate security in encrypting and decrypting its data by confusion and diffusion mechanisms between the symmetric keys, the ciphertext and the plaintext [31]

2.3.5 Diffie-Hellmann Key Exchange (D-H)

The Diffie-Herman Key exchange is a security protocol that produce the needed keys (Public Key Infrastructure (PKI) for the encryption and decryption of data files. Multiple users can securely exchange secret key for encryption and decryption of messages using the Diffie Hellman key exchange protocol. Several accounts support the fact that DHE is prone to attacks including impersonation attacks and man-in-the-middle attacks. The protocol is only limited to only key exchange but not for message authentication. The authors proposed an improved protocol to address the security challenges inherent in DHE [32].

In [33] a hybrid of the Diffie-Hellman key exchange protocol and the Advanced Encryption Standard (AES) was used. The combined cryptographic algorithm helped in validation, verification and securing of data, as well as providing data confidentiality in the cloud

2.3.6 Conclusion on Modern Ciphers

Modern ciphers use techniques that operate based on binary bit sequences to secure messages. These ciphers rely on secret key of the corresponding parties to securely communicate messages based on mathematical algorithms. For example, the Advanced Encryption Standard (AES) provides the specification on which electronic data is encrypted and decrypted. It is a symmetric block cipher algorithm with several block sizes variants. In encrypting a message, it divides the message into chunks and encrypt each chunk in a separate fashion. It then joins together the encrypted portions of the message chunks to form the ciphertext.

2.4 Cryptographic Hash Functions and Digital Signature

This sub-section of the document reviews existing literature and works that have been done in the context of securing messages using cryptographic hash functions and digital signature.

Cryptographic hash functions and digital signature schemes have been used in different situations and environments to enhance data integrity and message confidentiality. In the subsequent texts for this paragraph, these two although related and yet different concepts shall be explored further.

2.4.1 Cryptographic Hash Functions

Hashing involves a one-way mathematical function that accepts data of arbitrary size to produce an output (message digest or hash) of fixed size. A good hashing algorithm is supposed to be collision resistant. Salt is a random bit of additional data to one-way hash functions in password instances before hashing. In situations where multiple users on a system create same password, the use of salt results in generating unique password hash for these users. It is essential in providing an insolation against hash table attacks. An attacker is required to guess correctly the salt added to the password before encrypting it to be able to successfully compromise those hashes for the password.

There exist several cryptographic primitives for providing message integrity during electronic message communication from one point to the other. These cryptographic algorithms are to ensure that messages are kept securely, and help intended recipient of such messages to validate the genuineness of the source of the message.

Cryptographic hash function ensures the authenticity of messages are maintained. There are several cryptographic hash functions in use today. The Message Digest Algorithm (MD5), Secure Hash Algorithm (SHA), Race Integrity Primitive Evaluation Message Digest (RIPEMD), Twofish, Whirlpool are among the popular and commonly used cryptographic hash algorithms.

Due to several successful attacks on classical symmetric key ciphers in recent times, modern cryptographic schemes have adopted a technique to guarantee that data packets including IoT

node data is protected against alteration or tampering by third party nodes that may have compromised the system. The MD5 is a hashing algorithm that produces a checksum after taking the ciphertext and a key as input. The checksum is then used to digitally sign off the ciphertext using the sender's private key. The MD5 cryptographic hash algorithm was used in encrypting message after securely exchanging keys using the DHE to encrypt the array pixel values of plain images [34].

The Race Integrity Primitive Evaluation Message Digest (RIPEMD 128) maps data of arbitrary size of data as input to an output of fixed size bit string. RIPEMD is a hash function that accepts any arbitrary bit size of data as input, using a generic secret key to produce an output of a fixed size of 128-bit message digest consisting of binary string of 16 bytes. It is appropriate for verification and signatures [35]. RIPEMD-128 is a 128-bit hash function that operates using the Merkle Damgard. It is constructed by iterating 128-bit compression function h, taking an input of arbitrary message block size m and a 128 chaining variable [36].

The whirlpool function is composed of iteration of compression function with 512-bit key space to produce a 512-bit block dedicated cipher. To encrypt data of any size, the data is padded. It is adoptable to hardware implementations on both 8-bit and 64-bit platforms. It uses a substitution box where it generates randomly its 512-bit keys to provide digital signature to data [37].

The Tiger 192 hash function uses large translation tables and runs well on 64-bit platforms to produce a much stronger 24 bytes long output hash. It includes an internal state size of 192 bits, and block size of 512 bit. The 192-bit key size provides a stronger and better encryption. It also supports the secure exchange of keys through the internet for encryption and authentication between two communicating parties. The Tiger and its variant hash functions consumed less energy and yet provided an enhanced security among its peers. The cost in

terms of energy consumption requirements for the Tiger192 is light weight as compared to other hash functions in its category, but it produces an efficient and effective hash value that is suitable for enhancing the security of data[38] [39].

The Tiger hash cryptographic algorithm is applicable and effective for supporting secure access to cloud data. The Tiger hash was adopted at the device or the physical layer level for cloud user enrolling phases for authenticating and granting the appropriate access rights to verified users to access cloud data or services [40].

In the next subsection, the conclusion to cryptographic hash algorithms is presented.

2.4.2 Conclusion on Cryptographic Hash Algorithms

Cryptographic hash algorithms are useful and applicable in a variety of security solutions that include device and message authentication. Cryptographic hash algorithms work to complement other cryptographic schemes by providing authentication to validate integrity of messages for secure IoT communication. The strength of a cryptographic hash algorithm is based on the key size and the quality of the distribution of the hexadecimal characters in the message digest. The goal of a hash function it to reduce the chances hash collisions.

In the context of end devices for IoT, the choice of a cryptographic function is depended on computational processing and efficiency at avoiding collisions. An efficient and lightweight cryptographic hash function suitable for maintaining message integrity in networks where most of the end device are constrained will be an appropriate choice that achieves two unique benefits of efficiency of generation of message digest and lightweight of the hash in terms of storage the security solution.

The challenges with the choice of an appropriate hash function are constraints on communication, computation, storage overheads, and energy consumption [41].

In the next subsection, the digital signature scheme is presented.

2.4.3 Digital Signature Scheme

In electronic communication, the possibility of encountering and exchanging messages from an unauthorized device and user is high. To guarantee that a message is from an authorized device and user, the unique details of the device or user must accompany the message to help the recipient validate the integrity of the user and the device. Digital signature schemes are used to generate unique checksum or message authentication codes on every message propagated from a device within a communication loop. Digital signature schemes comprise cryptographic algorithms that use hash functions to generate keys to create a unique digital signature that is appended to messages to help a recipient of a message to validate the integrity of the message and its sender.

Digital signature schemes are cryptographic algorithms characteristically employed to validate the authenticity and integrity of a message. They are useful in issuing and managing key agreement between end devices in a communication loop, generate unique electronic signature for each message, and assist with verifying electronic signatures from the source of the message. The digital signature validation for a message is done at the receiver node. The unique digital certificates or electronic signature from a sending node for each message sent is to assure a receiving node or device the readiness to assist in verifying and validating the integrity of the source of the message. The digital certificate is to allow for checking non-repudiation in electronic communication. Thus, the public key or the shared secret for a sender can be used to verify the source of the message in the context of an electronic

communication. The message, the public key or shared secret and the digital signature that accompanied the message is used in verifying and validating the integrity of the source of the message. The message integrity validation is done at the receiver node.

In [42] a public key authentication protocol that shared the security properties of the elliptic curve digital signature algorithm was proposed and adopted for authenticating IoT nodes prior to the transmission of data across the nodes. The digital signature for IoT device authentication implementation comprises three phases. The node-to-node verification was based on mutual authentication, node-to-base (one-way authentication) and the key agreement phases. The digital signature scheme integrated a sponge-based hash function as its fundamental hash algorithm in generating the digital signature for authentication of devices.

In [43] a multiple-time digital signature algorithm that adopted the security properties of the elliptic curve signature scheme was used to provide an ultra-lightweight digital signature scheme for IoT devices. The security mechanism did not include the complex computational overhead in classical elliptic curve algorithm. It adopted an efficient multi-time elliptic curve for signature generation at the signer side that involved two hash function calls consisting of modular multiplication and a modular subtraction operation. The reduced computational overhead resulted in an energy efficient operation at the signer node. The digital signature scheme used a hash function that used lightweight signature scheme appropriate for IoT environments.

In [44] a thorough analyses on selected IoT authentication mechanisms was conducted. There are several categories of digital signature schemes. Some of the digital signature schemes involve two steps whiles other types of digital signature involve three steps. Principally, there are three phases involved in a digital signature algorithm. There are the key generation phase, the digital signature generation phase, and the message verification phase. In whatever

category of number of steps involved in the digital signature algorithm, A signature generated by the private key of the signer (sender node) will correspondingly be verified by the second step where the public key of the sender will be part of the parameters for verifying the signature.

There are the RSA (Rivest, Shamir, Adleman) based signature scheme, the Elliptic Curve El-Gamal Digital Signature Scheme, Elliptic Curve Digital Signature Algorithm, Edwards-Curve Digital Signature (EdDSA), Shortened Complex Digital Signature Scheme (SCDSA), Lamport Signature Scheme (LSS), Proxy Blind ECDS, Cloud-Based Digital Signature Application [45] [46].

In [44] The performance analysis of digital signature schemes was carried out in their work. The characteristics of some selected digital signature schemes for internet of things environment was analysed. The factors such as the key size, signature size, signature generation time, and signature verification time were comparatively examined. Although a key size plays a significant role in determining the security strength of a cryptosystem, equally the time taken by a digital signature in the verification phase of a digital signature plays a significant role in adding to the turnaround time for the computation of the security solution system. Particularly for the constrained devices in wireless sensor networks, the size of the generated signature, and the time taken for the verification process of a digital signature is crucial since it could affect the operational efficiency as well as the strength of the cryptosystem. Whereas, the key size of the digital signature scheme is important in contributing to the size of the digital signature, equally critical factors that are useful in the context of the constrained nature of the devices in IoT, are the time taken in the generation of digital signature, and the turnaround time for the verification phase of the digital signature at the receiver because it would form the basis in the decision of selection of a particular digital signature scheme for an IoT network.

Additionally, some digital signature schemes like the EdDSA avoid utilizing branch processing and array indexing measures that are tied to secret data in other to increase security resistance against side channel attacks. The EdDSA, randomly generates and assigns a nonce (secret value) for each signature value generated. The nonce used in EdDSA are deterministically applied and hashed forming part of the private key and the message [47] [48].

Table 3: Characteristics of Digital Signature Schemes in IoT

Signature Scheme	Key size (bits)	Signature size (bytes)	Signature generation time (ms)	Signature verification time (ms)
RSA	1024	128	7.8	0.4
DSA	1024	384	3.5	4.5
ECDSA	160	64	3.1	8.2
EdDSA	256	512	0.08	0.16
BLS	100	20	2.2	8.6

Table 3 highlights a comparative analysis of the performance of selected digital signature schemes for internet of things. The measurement units for the characteristics of the digital signature schemes for the key size, signature size, generation and verification of signature are in bits, bytes, and microseconds, respectively. The Edwards Curve Digital Signature (EdDSA) has the least signature generation time (0.8ms) and signature verification time (0.16ms) with a key size and signature size of 256 bits and 512 bytes, respectively.

2.4.4 Conclusion on Digital Signature Schemes

The heterogeneous nature of a classical IoT network and the unsecure nature of the internet which also serve as a conduit for transmitting messages across end devices in an IoT network give room for the possibility of having an authorized user or device interrupt, intercept, falsify, and manipulate messages in transit. These unauthorized activities could result in corrupting the message and resending it to an authorized node. The use of a digital signature algorithm will help in the key management, digital signature generation, and verification of messages to assure and validate the integrity of the source of messages.

In the next subsection, the blockchain technology is presented.

2.5 Blockchain

Blockchain is a database framework consisting of a sequence of blocks which keeps a total list of transaction records like a conventional public ledger [49]. In other words, it describes a distributed ledger that provides an append-only sequentially chained blocks. Blockchain technology became popular with the introduction of the Bitcoin cryptocurrency whose white paper was authored by the famous Satoshi Nakamoto [50].

The classical Blockchain technology served as the foundation for generating, storing, and transferring the Bitcoin cryptocurrency. The technology has however undergone a lot of changes that have resulted in the creating of technologies that adopt that unique feature of decentralized processing and computation, distributed storage, transparent transaction, tamper-proof systems, non-repudiation of transaction activities, permanent storage that uses append-only log approach

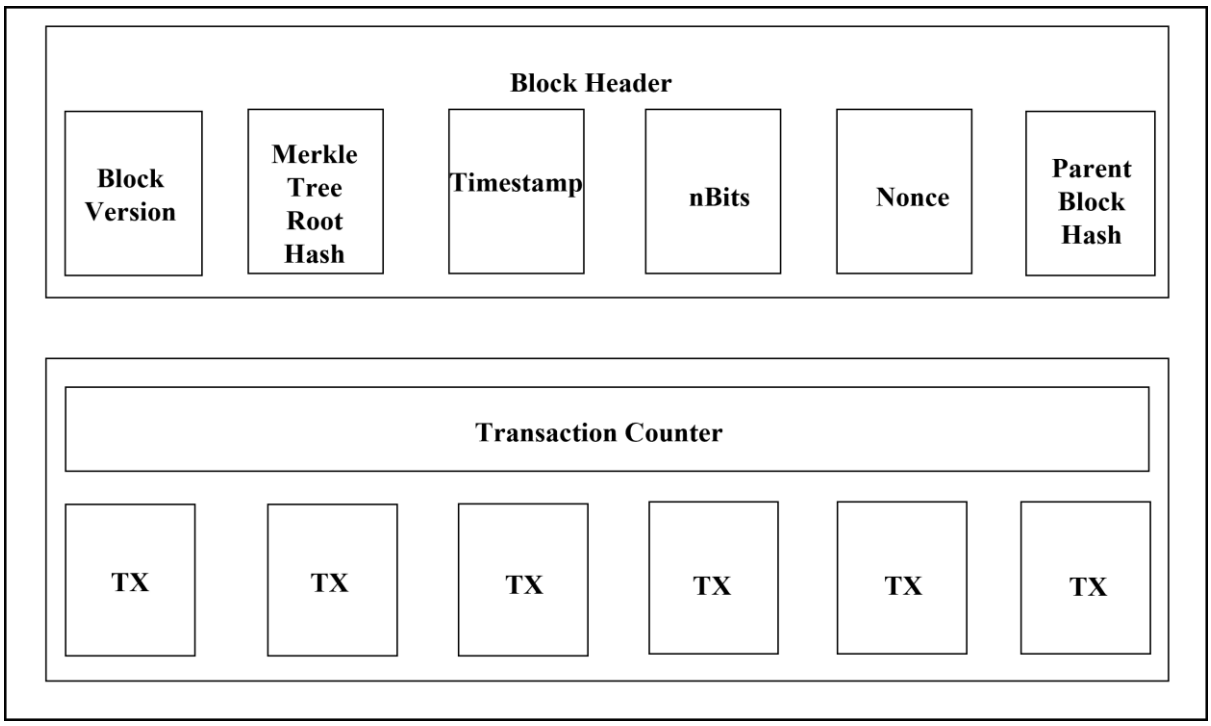

Figure 6: Blockchain block structure [49]

Figure 6 displays the block structure of a blockchain highlighting the components that constitute a typical block in a blockchain network. A block comprises the block header and the block body. The block header contains the Block version, Merkle tree root hash, Timestamp, nBits, Nonce, Parent block.

- The set of block validation rules governing signing blocks onto the blockchain is indicated by the block version.
- The Merkle tree root hash contains the hash value for every transaction in the block.
- Timestamp indicates and registers the current time in seconds in universal time since January 1, 1970.
- nBits describes a target threshold of a valid block hash.
- Nonce is a "A time-varying value that has at most a negligible chance of repeating; for example, a random value that is generated anew for each use, a timestamp, a

sequence number, or some combination of these. It can be a secret or non-secret value" [51]. 4-byte field that contains a number which indicates the difficulty level limitations attached to a hashed or encrypted block in a blockchain. In a bitcoin cryptocurrency, miners solve for the nonce to receive bitcoin as an incentive. The first miner to solve for the nonce obtains the bitcoin reward.

- Parent block hash is a hexadecimal 256-bit hash value that points to the most recent block chained to blockchain. It connects to the previous block in the blockchain.

The block body on the other hand consists of a transaction counter and list of transactions. Asymmetric cryptographic mechanisms are used to validate the authenticity of blockchain transactions [52].

The classical cryptocurrency sharing program has undergone several modifications that has resulted in a disruptive infrastructural platform to serve as a database framework and maintains the core features and characteristics of the traditional Blockchain. Blockchain offers an append-only log whose data is publicly available and employs a mathematical scheme to validate transactions, order them and chain the validated transactions together. Transactions that are stored on a Blockchain is finalized or committed after all the active nodes have reached a global agreement on the current state and content of the Blockchain [53]. Consensus algorithms ensure that there is a global agreement between all the active nodes and users on the network. New transactions are mined or validated in other to be appended on the existing chain of validated transaction that constitutes the blockchain. An update request to validate a transaction is initiated from any of the active nodes in a process called mining. Validation of transactions are conducted using timestamp and digital signature schemes that check the states, contents, and the length of the chained transactions from previously signed chains. Digital signatures ensure data provenance. The first node to

compute and validate the transaction request is rewarded for energy and computational power used in the mining process. Consensus algorithms are used during the process of mining transactions. Consensus algorithms used in classical blockchain mining include Proof-of-work, proof-stake, All active nodes have. There are several consensus algorithms in use today.

Blockchain is a promising infrastructural technology that is finding its way into a growing number of domains like big data, finance, and the field of medicine for health records management.

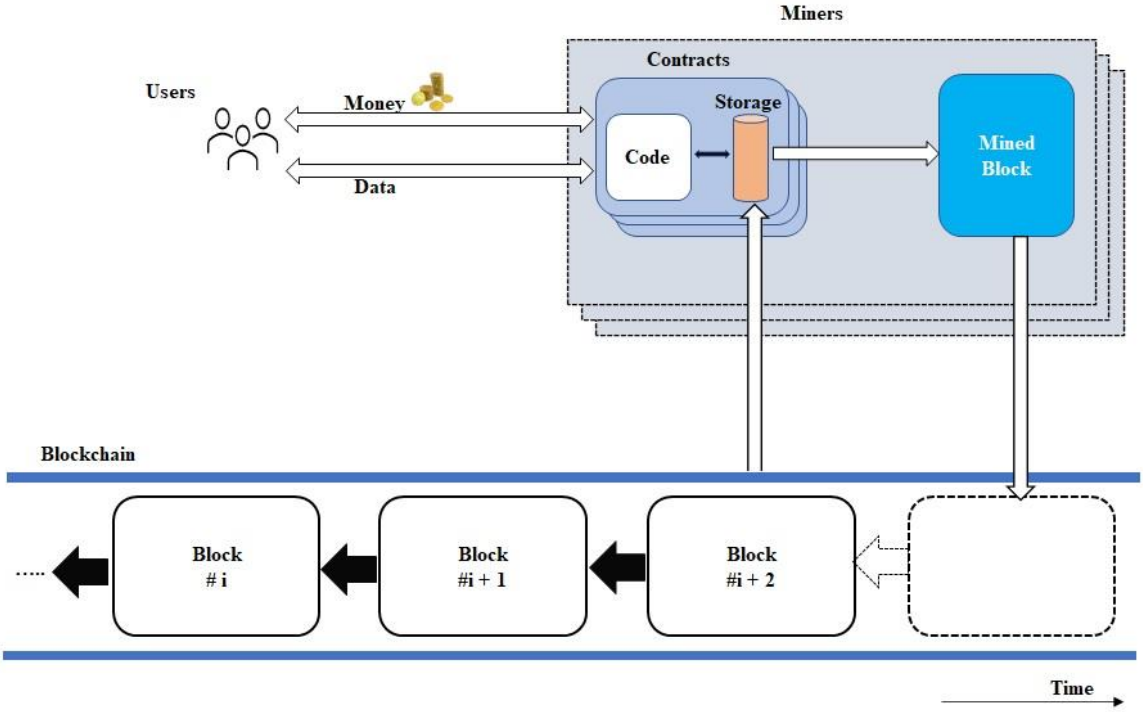

Figure 7: Blockchain with a smart contract [53]

In Figure 7 a schematic diagram of a cryptocurrency system with smart contracts is illustrated.

A network of miners or transaction validators execute smart contract in a blockchain network. The contract's state on the blockchain is updated using the outcome of the execution by the miners.

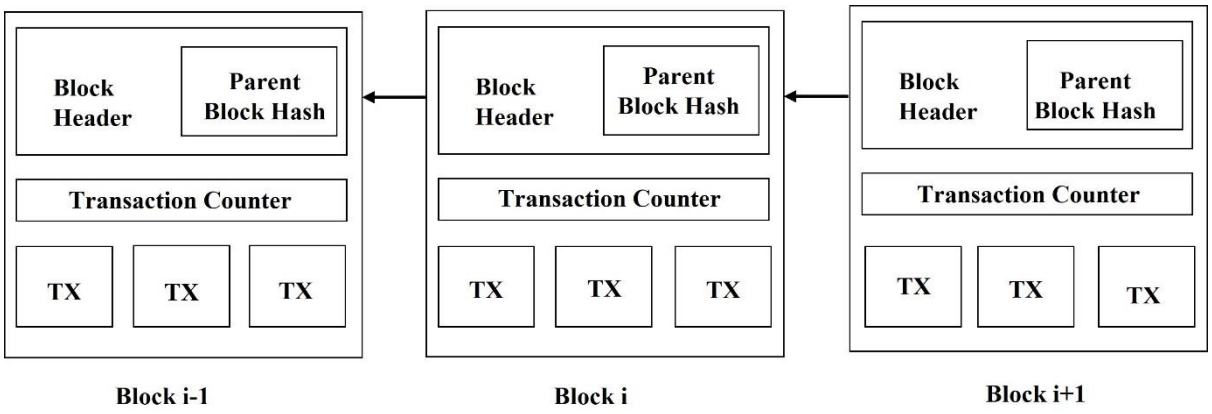

Figure 8: An example of blockchain consisting of a continuous sequence of blocks [54]

In Figure *8*, the schematic diagram of a blockchain which comprises a chained sequence of blocks is presented.

2.5.1 The workflow for Blockchain

To add a new block to the blockchain, there are background processes that must satisfactorily be met which is managed by smart contract and consensus nodes. The processes are:

i. A transaction must be grouped with other recently authenticated transactions in a block.
ii. The consensus nodes validate messages or transactions integrity by verifying that the block transactions comply with defined rules.
iii. The consensus nodes execute consensus mechanism to validate transactions.
iv. Consensus mechanisms includes incentives to reward active nodes that undertake the transaction verification before they are appended onto the block.
v. The verified transactions are stored into blocks and the blocks are chained together to form the blockchain

2.5.2 Blockchain Platforms

Blockchain has evolved since its creation some 14 years ago as a peer-to-peer network. It keeps undergoing changes. There are several blockchain platforms available today. Notable examples however are Bitcoin, Ethereum, and Hyperledger Fabric.

Bitcoin is a cryptocurrency that operate on the peer-to-peer network as a verification system for the transfer of digital assets or the cryptocurrency. It involves actively the use of proof of work consensus protocol for validation of transaction through a decentralized peer-to-peer mechanism, which eliminate instances of a single point of failure that is popular with centralized authority for authentication of financial transactions. The Bitcoin blockchain platform uses timestamp and stronger cryptographic hash functions to maintain the state of transactions by hashing the timestamp of ongoing transactions to a block header forming the public transaction ledger for the cryptocurrency. Like classical electronic funds transfer systems that depend on a centralized trusted third-party authority, the bitcoin uses cryptographic proofs to validate transactions. It allows users of the system to create payment accounts using their email address [55] [56].

Ethereum functions as a transaction validation system and as a digital asset for a cryptocurrency which is referred to as Ether (ETH). It also provides a platform for smart contract applications using the Ethereum Virtual Machine (EVM). Thus, the Ethereum blockchain platform allows for the validation of financial transactions, provision of smart contract for validation of other non-financial data [57] [58].

In the next subsection, the classification of blockchain is presented.

2.5.3 Classification of Blockchain

There are several classifications of Blockchain. Blockchains could be categorized based on the architecture, uses, and level of involvement in consensus. Blockchains can be categorized into Permissioned blockchain, and Permissionless blockchain. The categorization of the blockchain is based on how member nodes join the blockchain network to validate or append transactions to an existing block. In permissionless blockchain, all active nodes can validate transactions after expending some resources to authenticate the data appending process but joining the network as validator nodes is free. Permissioned blockchains on the hand, restricts external nodes from participating in ordering and updating transactions into blocks [59]. There are four types of blockchains. Public Blockchains, Private Blockchains, Consortium Blockchains, and Hybrid blockchains [60].

In a public blockchain, all active nodes have equal chance of reading and writing onto the publicly available ledger. There are no access restrictions in public blockchains. The distributed ledger can be written onto by anyone who has the updated list of transactions that reflects the current state of transactional history. All active nodes have same chance of becoming validator nodes to approve transaction to be appended onto the chained blocks. Popular consensus algorithms used in public blockchains are proof-of-work, and proof-of-stake. Bitcoin and Ethereum are examples of public blockchain.

In private blockchain, writing or appending or updating the current state of transactions to the block is undertaken by only selected users who have write permissions. It operates in a permissioned fashion. Distributed Ledger mechanisms are used to maintain the decentralized peer-to-peer database. The Hyperledger, Multichain projects operate as private blockchains. In private blockchain, privacy is controlled by a group of trusted and verified users. The

trusted group of users manage and maintain the blockchain network through a consensus mechanism.

Hybrid blockchains, on the other hand combine the unique features and qualities of both public blockchains and private blockchains. Thus, centralized, and decentralized databases are integrated into hybrid blockchains. The Dragonchain is a popular example of the hybrid blockchain.

Lastly, the consortium blockchains adopt features of classical blockchain and operate based on partially decentralized database framework. Such blockchains are maintained by a group of entities with a common operational control and interest such as what happens in bank with several networked branches, and central Government [61].

In the next subsection, the blockchain use cases are presented.

2.5.4 Use cases

Several existing business processes across multiple fields can be modernized by restructuring and automating them to reduce risk whiles improving on the security of implementation, as well as enhancing the turn-around time for undertaking business transactions and protecting investments to save on resources including money. The blockchain technology has created a new business model with its security improvement potential for data driven industries and businesses. The possibility of distributing data and ensuring decentralized processing and authentication of data for networks that spans several localities and environments makes a good and business investment justification for designing, implementing, and deploying blockchain solutions on existing networks. Trust, transparency of transactions, and tracking

of business processes across partner platforms are enhanced using blockchain solutions. The distributed ledger technology assures improved security, cost-effective investments, and trust enforcement approaches between several business entities that integrate to benefit from common data source. The financial service sectors that rely on partner platforms for ensuring electronic transaction in the form of electronic funds transfers, electronic commerce payments, trading of stocks and bonds, deposits and electronic cash withdrawals can benefit from business blockchains to reduce risks, track transactions, and remotely authenticate to authorize business processes [62] [63].

The distributed ledger technology (DLT) provides an integrated platform that runs smart contract microservices and implemented on containerized applications like the Hyperledger fabric to allow all connected parties to automate the authentication and authorization of processes and transactions across already established agreements that bind connected parties.

In the next subsection, the blockchain consensus algorithms are presented.

2.5.5 Consensus Algorithms

Blockchain eliminates trusted-third party intermediaries in authenticating transactions. Authentication of transactions in blockchain is achieved using consensus algorithms. Nodes that store the current state with complete history of transactions constitute the active nodes that qualify to take part in a consensus. The nodes that undertake consensus do not have prior knowledge of each other, and are organised in such a way to make them Byzantine- fault tolerant [64]. The goal of consensus algorithm is to allow multiple nodes working on same data with multiple processes to agree on the state of that single data [65].

Blockchain system relies on consensus as a fundamental requirement to enforce trust among all participating nodes in the provision of guaranteed ordering of transactions and validating the blocks of transaction. Consensus is managed by the consensus layer of the Hyperledger architecture. Consensus algorithm provides the medium for generating an agreement on the order and confirming the correctness of the set of transactions that constitute a block. The blockchain system introduces a trust system middleware layer that eliminates completely the involvement of a trusted third-party entity for authenticating and authorizing transactions. In a blockchain system, trust is enforced using consensus approach [66] [67].

The applicability and efficiency of blockchain consensus protocols are based on the three properties: safety, liveness, and fault-tolerance [68].

- The safety property of blockchain consensus algorithm is the quality of all consensus nodes producing valid output to describe consistent shared state of transactions. The rules for running the consensus form the basis for determining the validity of the output.
- The liveliness property of consensus protocol on the other hand concerns itself with the condition where all non-faulty nodes undertaking the consensus exhibit the capacity to produce a value for an output.
- The last property of a consensus protocol describes the quality of a consensus node recovering from failure to take part in the consensus till a block is formed. This quality constitutes the fault tolerance property.

There are several types of consensus mechanisms available for undertaking blockchain implementations. The choice of implementation of a consensus type is depended on the network requirements and fault tolerance models. There are the lottery-based algorithms including proof of elapsed time (PoET) and Proof of Work (PoW) or using voting-based

methods like the Redundant Byzantine Fault Tolerance (RBFT), Proof of stake (PoS), Practical Byzantine Fault Tolerance (PBFT), Proof of Authority (PoA), Proof of Personhood (PoP), Proof of space (PoS), Proof of concept (PoC).

Consensus algorithms used in blockchain based authentication systems for IoT have included Proof of Work (PoW), Byzantine-Fault Tolerance (BFT), Proof of Stake (PoS). In those works, identity models were based on public key infrastructure in device access control, data authentication schemes, detection and prevention of intruder nodes, and validator for establishing connection between sensors and high-end sink nodes [69],[70] ,[71].

Proof of Work (PoW) consensus protocol ensures that each node of the network is determined through recalculation of the hash of the block header. The hash value of the block header keeps changing to reflect its contents. As new transactions are added to form blocks, the hash value for the block correspondingly get updated. In PoW, a nonce which determines the difficulty levels at determining a hash value must either be the same or slightly less than that nonce value to close a PoW consensus session. The first consensus node to calculate to get the hash value answer, gets rewarded. The calculation of the hash value for the block header involves several attempts at recalculating by using different nonces. PoW is the fundamental consensus protocol for Bitcoin network. Recalculation of the hash of the block header involves a great amount of time and computational power. The Bitcoin cryptocurrency rewards the consensus nodes that correctly solves the block header hash value puzzle a portion of the cryptocurrency [72] [73].

Proof of Stake (PoS) achieves consensus through voting. It involves using voting-based mechanisms to validate new blocks prior to appending that new block to the blockchain. In blockchains that operate based on PoS, the blockchain is assigned an amount of

cryptocurrency value in bitcoin. To confirm work done using the PoS, a validator node from among the consensus nodes must prove possession of more of the blockchain native currency. Comparatively the PoS uses less energy in validating new blocks than using PoW [74].

There is Proof-of-Activity (PoA) that is a fusion of PoW and PoS. The PoA operated on an economic phenomenon with the assumption of "Tragedy of the Commons" which described a situation where a limited resource for several agents could be ruined in situations where there is uncontrolled use [75].

The Ripple consensus algorithm is a permissioned blockchain consensus algorithm that requires access permission because it is not publicly accessible. It operates in rounds using active nodes as servers. It adopts an approach of closing an active ledger updating session once a consensus is reached to store and maintain an identical state of the ledger on all active nodes [76].

A consensus mechanism must provide a trade-off between performance, fairness, and security.

In the next subsection, the blockchain smart contract is presented.

2.5.6 Smart Contracts

A smart contract could be described as self-executing programs that are stored on a blockchain and get triggered or run when a preestablished agreement or condition expressly captured in the execution policies is met. There are specific platforms for executing or deploying smart contracts. A smart contract comprises a computer program, a storage file, an

account balance and executed by all consensus nodes on the network. The program code of a contract cannot be modified once a transaction is created. The Consensus on a transaction is reached by a network of consensus nodes. The outcome on the consensus is added onto the blockchain by an update execution that is conducted by the contract's program logic.

A contract is created by users on the network by posting a transaction to be added onto the blockchain. Any user on the network can create a contract by posting a transaction [77]. The smart contract in blockchain provide business logic and business process mechanism capabilities into the blockchain. Users and nodes on the blockchain posts transactions where the miners or consensus nodes check for the correctness and availability of transaction in order add the validated transaction to the chained of blocks.
The smart contract provides an autonomous execution for transactions by checking for correctness and availability without checking for privacy of transaction.

In summary, a smart contract is a computing protocol for distributed systems that rely on a blockchain based system implemented as distributed ledger technology (DLT). The smart contract assists blockchain-based protocols to facilitate the creation of digital self-executing policies or contracts, verify, and deploy these digitized policies for the smooth transaction of information between communicating nodes and entities without any intermediary application or trusted third-party system.

A classical blockchain smart contract require an energy heavy consensus mechanism to run smart contracts on the blockchain. The potential of the blockchain and smart contract capabilities provide an interesting advantage to address the security vulnerabilities that IoT networks are predisposed to due to their inherent limitations in the design and security

specifications of the major device actors in sech networks. A deliberate analysis of existing architectural models needs to be made to inform a choice for an appropriate modern architectural model that can support the implementation of a blockchain-based cryptographic mechanism that uses smart contract to authenticate node data.

The next subsection presents the illustration of blockchain and the conclusion on blockchain.

2.5.7 Conclusion on Blockchain

Since the advent of the bitcoin cryptocurrency blockchain, there have been applications that adopt the concept and framework of the traditional blockchain for specific operating environments like the internet of things. There are consensus algorithms that does not involve monetary incentive for authenticating transactions. The non-monetary incentive-based blockchain consensus that rely on lightweight smart contracts provide useful support for authentication of IoT data.

In the next subsections, the concept of internet of thing, and the classifications of IoT are presented.

2.6 Internet of Things

The connection of multiple devices to generate and share massive data from one end device to the other using the internet has been made possible due to the concept that allows ordinary devices to be given computational and networking capabilities to capture critical data from the immediate surroundings of the end devices and communicated the data among other end devices through the internet Machine learning approaches, artificial intelligence, the internet of things, and other associated pervasive technologies have enabled the creation and the

subsequent sharing of massive data across devices, subsystems and platforms using the unsecured channels that the internet provides. The internet of things allows the instant collection and transmission of nodal data in instant streaming fashion. Although there may exist at the hardware level an embedded security mechanism for assuring the security, privacy, and the confidentiality of the node data that they may be sensing, analysing, and communicating, the heterogeneous operating environment may weaken the general security of the system. Any weakness in one of the end devices either a sensor, actuator, or diode will introduce a security lapse to the networked devices and subsequently increase the attack area within the IoT [78].

In a classical internet of things system, the design structure for the connection and development of the system is based on three orientations. The internet-oriented (middleware), Things-oriented (sensors) and semantic-oriented (knowledge). Depending on the deployment goal of the internet of things, one of the three design-orientations are adopted. Smart applications and automation projects that require smart objects to be used adapts the internet-oriented design approach where the design orientation is geared towards the protocols for the middleware. The things-oriented design-based internet of things rely on the appropriateness of the main hardware actors of the IoT including sensors, actuators, and edge device connectivity gadgets for autonomous applications. The platforms which offer the knowledge for handling data representation, analysis, and manipulations by collaborating with all the smart objects and the provision of connectivity for data to be shared across the devices and platforms describes how the semantic-oriented design functions. In other words, the internet of things is the collection of the three design orientation that involves applications, backend services, the internet, local network, and devices [79]

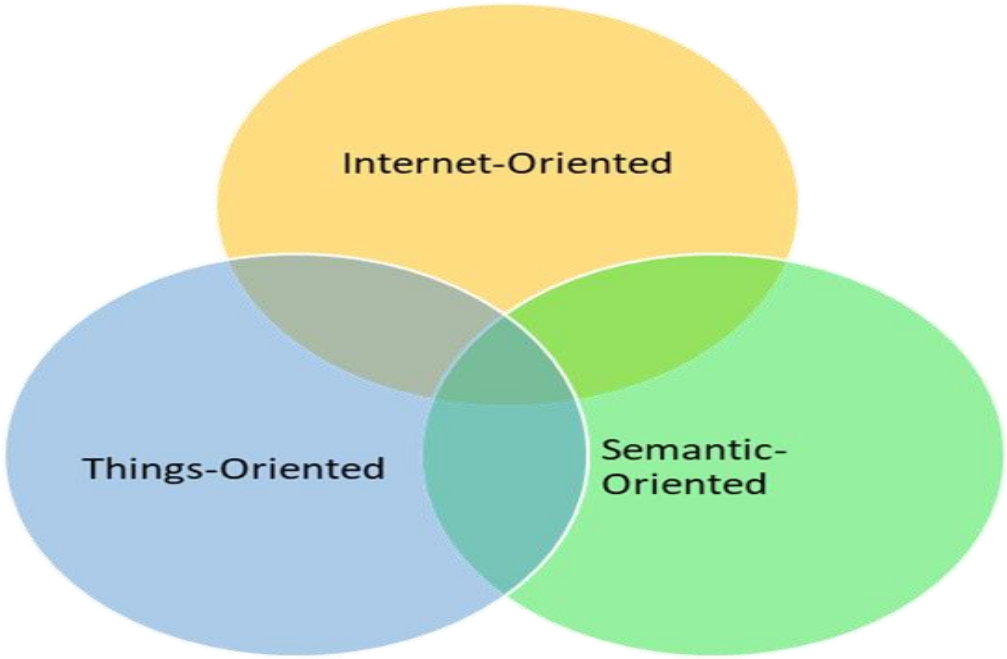

Figure 9: The Internet of Things design orientations [79]

Figure 9 illustrates the three orientation designs of Internet of Things.

In the next subsection, the classification of internet of things is presented.

2.6.1 Classification of Internet of Things

Internet of things networks are grouped based on the devices used, the location or environment of use, and the criticality of impact in the failure or malfunctioning of the IoT device on living things. IoT end devices operate and communicate based on the communication model that is applied at the design stage of the network. There are four communication models used by IoT devices. These communication models define the architectural framework that details the specific devices and components that makeup the IoT network. These IoT communication models are device-to-device communication model, device-to-cloud communication model, device-to-gateway communication model, back-end data-sharing communication model [80].

The popular IoT microcontroller boards are Arduino, and Raspberry pi [81] . Based on the purpose and devices used, IoT can be classified into five main forms.

Table 4: IoT sensor classification, functions, and devices

IoT Classification	Referenced Paper	Devices Used	Functions
Position, Occupancy, and Motion based IoT	[82], [83], [84], [85], [86]	Potentiometer, Inclinometer, Proximity sensor, photoelectric sensor	Measures the position of objects from any direction. Identification of the existence of entities (human and animal) in the observation area. Movement detection of entities (human and animal)
Velocity, force, and Pressure based IoT	[87], [88], [89]	Accelerometer, gyroscope, force gauge, viscometer, tactile sensor, barometer, bourdon gauge, piezometer	Movement measurements and the indication of movement velocity along a particular path (linear and angular).

IoT Classification	Referenced Paper	Devices Used	Functions
			Identification and measurement of magnitude threshold and the force applied
Flow and Chemical based IoT	[90], [82], [83], [87]	Anemometer, mass flow sensor, water meter, breathalyzer, olfactometer, smoke detector	Identification and the measurement of rate of fluid flow. Identification and measurement of the concentration of chemicals.
Acoustic and light based IoT	[83], [89]	Microphone, geophone, hydrophone, infrared sensor, photodetector, flame detector	Moise or sound level measurements. Identification of the presence of light.
Humidity, Temperature, and Radiation based IoT	[91], [83], [92], [93], [94], [95], [85], [96]	Hygrometer, humistor, soil moisture sensor, thermometer, calorimeter, temperature gauge, scintillator, neutron detector	Detection and measurement of water valour in the air. Sensing of environmental radiations. Temperature

IoT Classification	Referenced Paper	Devices Used	Functions
			measurement

Table 4 shows the classification, functions, and device types for Internet-of-Things

In the next subsections, hardware components for IoT, IoT hardware, IoT operating systems, IoT architecture, IoT communication protocols, network security vulnerabilities, and message authentication in IoT are presented.

2.6.2 Hardware

Internet of things setup forms the testbed for running applications that are appropriate for constraints devices. There are hardware devices that make up a wireless sensor network. The ecosystem for an internet of things comprises the native IoT with an external network to be able to provide an inter node communication from any environment where the sensors extract measurements from the environment and transfer the collected information to an exterior network [97]. The ideal IoT network which is engineered based on the proposed architecture comprises low-level sensors and actuators, the edge routers or gateway devices, the internet itself, and the cloud. An IoT network consists of a power module, communication channel, low power sensing node, and an embedded processor. Each of these hardware components performs unique functions within the network. They perform complementary roles by integrating their functions together to make it connect and collect data, aggregate, and process the data and communicate the data across nodes using the internet.

The low power sensing node is used for collection of external data consisting of physical quantities. These physical quantities include temperature, humidity, pressure, air quality, and

frequency. The power module provides the units for converting the alternating current AC voltages between devices to an appropriate power rating to give life to the IoT network.

2.6.2.1 Operating System

Internet of Things operating system is an onboard system application designed to perform within the constrained environments to provide the connectivity, networking, security, storage, remote device management needs for internet of things networks. These operating systems are designed for the peculiar demands and specifications of IoT devices. The operating system for IoT provides a platform to connect other resource constrained devices and applications such as the cloud to collect, analyse, transmit and store sensor data. There are several kinds for operating systems for IoT. The RIOT, TinyOS, OpenWrrt, Contiki, FreeRTOS, MbedOS, Micro Python, Windows10 IoT, and Embedded Linux are examples of IoT specific operating systems [98].

2.6.2.2 Internet of Things Architecture

Internet-of-Things are differentiated from each other based on several factors including the devices used in the design and construction of the system. The architecture of an IoT describes the structural and the compositional elements that collectively support the flow of data from the sensor through a network to another end device for analysis, manipulation, and storage. Some of the elements making up the IoT architecture includes sensors, sink nodes, cloud services, actuators, protocols, layers, and gateways. Traditional IoT architecture is composed of sensing or perception layer, networking layer, middleware layer, application layer, and business layer. The perception layer describes the physical objects or the sensing devices that are used to collect, analyse, manipulate, transmit and store node data.

The perception layer contains the devices for sensing or recording physical parameters from the immediate environment of the sensor and transmitting the sensed data to the middleware layer through the network layer. The end devices that make up the perception layer include Barcode, sensors, radio frequency identifications (RFID), infrared, and actuator [99].

The network layer uses different networking devices, services and protocols including Bluetooth, wireless Fidelity (Wi-Fi), Light Fidelity (Li-Fi), Near Field Communication (NFC), Z-Wave, Long Term Evolution (LTE), Low-Power Wide Area Networking (LPWAN), Very Small Aperture Terminal (VSAT), Power-Line Communication (PLC), ZigBees for coordinating and correlating other smart devices, network devices and gateways for transmitting and processing of the sensed physical parameter. There are several categories of the communication protocols. These categories include short-range wireless communication protocols, Medium–range wireless communication protocols, Long-range wireless communication protocol, Wired communication protocols [100].

Table 5: IoT Architecture

Business Layer	Business Model Flowcharts and Graphs
Application Layer	IoT Smart Applications and Management. Smart Homes, Smart Grid, Healthcare system, Intelligent transportation
Middleware Layer	Database, Service Management
Network Layer	Communication networks: VSAT, LPWAN, LTE, 5G, Z-

	Wave, NFC, 3G, UMTS, WI-FI, Bluetooth, Infrared, ZigBee
Perception Layer	Physical Objects – Wireless Sensor and Actuator Networks (WSAN), RFID, Barcode.

Table 5 illustrates the various component categorization for IoT architecture.

2.6.2.3 Communication Protocols

Communication protocols define the framework for device interaction and message transmission within a computer network [101]. Internet of things devices use various protocols to communicate across other devices to share critical information [102]. IoT protocol stack can be grouped into Application layer protocol, Transport layer protocol, Network layer protocol, Data link layer protocol, and physical layer protocol [103].

Table 6: IoT Communication Protocols

Name of Communication Protocol	Description and Use
Constraint Application Protocol (CoAP)	These are lightweight protocol for delivering machine-to-machine communication. It offers secured, reliable, and flexible routing of messages. They are responsible for delivering message passing communication between low cost and low power resource constrained IoT devices. The choice for any of the protocols are depended on the quality of service, security, flexibility, applicability, and responsiveness. They provide publish/subscribe messaging services.
Message Queuing Telemetry Transport (MQTT)	
Extendible Message Persistent Protocol (XMPP)	
Advanced Message Queuing Protocol (AMQP)	

Table 6 shows the communication protocols for IoT

The middleware layer describes the database structure and the management services for the collection, processing, analysis, and storage of sensed data. Several technologies such as cloud computing, databases, and big data analytics are implemented at the middleware layer.

The application layer is concerned with providing the framework for interfacing and deploying IoT applications and services to the user. These services include smart applications and automation activities such as smart home, smart city, smart healthcare, smart grid.
The business layer depends on business models that use flowcharts, graphs, and other visualization elements for managing the services and the entirety of the IoT system [104].

In the next subsection, the IoT network security is presented.

2.6.3 Security

Security in its generic form including computer security, network security, and information security offers a collection of rules, controls, procedures, and technologies that work together to safeguard the computer or network devices, the network infrastructure, cloud-based systems, and data from unauthorized users. Security in the context of internet of things, concerns the provision of a protection framework that maximises the safety, privacy and availability of end devices, digital access channels, and the network infrastructure.
Threat models for software defined networks and recent research on internet of things security points to an IoT security landscape of an increased vulnerability exploitation of such networks. A security vulnerability is a “weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a

threat source" [105]. IoT network security vulnerabilities include hardware design vulnerabilities and IoT specific software vulnerabilities. The vulnerabilities serve as an entry source for attackers to compromise the confidentiality, integrity, availability of devices and data. These weaknesses mainly exist and exploited on the perception, network, and application layers [106]. Popular cybersecurity vulnerabilities are triggered by system misconfigurations, out-of-date or unpatched software, weak authorization credentials, malicious insider threats, poor data encryption mechanisms, and zero-day vulnerabilities [107] [108]. The interconnectedness of the devices and interfaces in a heterogeneous IoT network generate multiple points of attacks for such networks. These points of connection could serve as an entry point for attackers to gain access to such networks. A basic architectural design for an IoT comprises different components, applications, and devices. Each of the devices used have their own unique design and security vulnerabilities. These devices and components interact internally to provide a channel for transmitting information from one end device to another end device. There are attacks that occur at each of the layers within the open systems interconnection (OSI) model. The physical layer is liable to sniffing attacks. The data link layer suffers from sniffing attacks, the network layer is predisposed to attacks including sybil attacks, wormhole, sinkhole, backhole, IP spoofing, hijacking, and man-in-the-middle attacks. The session layer is prone to hijacking attacks. The presentation layer is susceptible to phishing attacks. Similarly, the application layer is predisposed to exploitation attacks. Any attempt at designing a security solution must propose mechanisms that greatly offer protection against a greater number of the attacks that these layers are predisposed to [109] [110].

In the next subsections, the attack scenario in IoT, is presented.

2.6.4.1 Introductory Concept on Attack Scenarios

This subsection explores the related literature on the approaches and the environments that attackers might use to explore the vulnerabilities in a network. Since a network system comprises a lot of components, attackers adopt ways and techniques to exploits the deficiencies in the network.

2.6.4.2 Attack scenario in Internet of Things

The attack scenario adopts an architecture with the goal of defining the security requirements expected of a solution that weakens the possibility of an unauthorized sensor entry into a local IoT network and providing a protection scheme to safeguard the assets or the communication of critical data within the IoT. The scenario describes a methodical procedure for identifying sensitive assets, threats to those assets, and vulnerabilities that make threats a fundamental problem for IoT. A compromise on the security of an end device has a direct effect on the security of the whole IoT architecture.

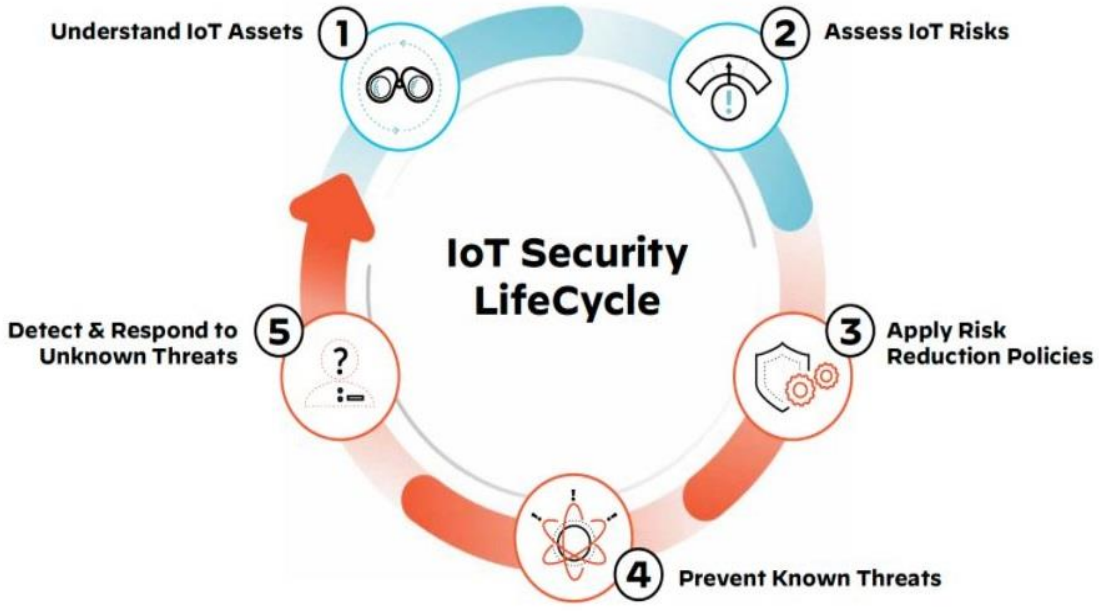

Figure 10: IoT Security Lifecycle [111]

In Figure 10, the landscape for the lifecycle of IoT security is presented. In ensuring a secure environment is created for communication of node data across an unsecured channel, deliberate attempts must be made to minimise risks at each of the phases that constitute the IoT security lifecycle. There exist threats that compromise the security in IoT networks. Some threats are popular and can be predicted. Others are not popular but are environmentally dependent as well as device native. Particularly for the unknown threats, a security mechanism that mitigates the chances of a compromise is desired.

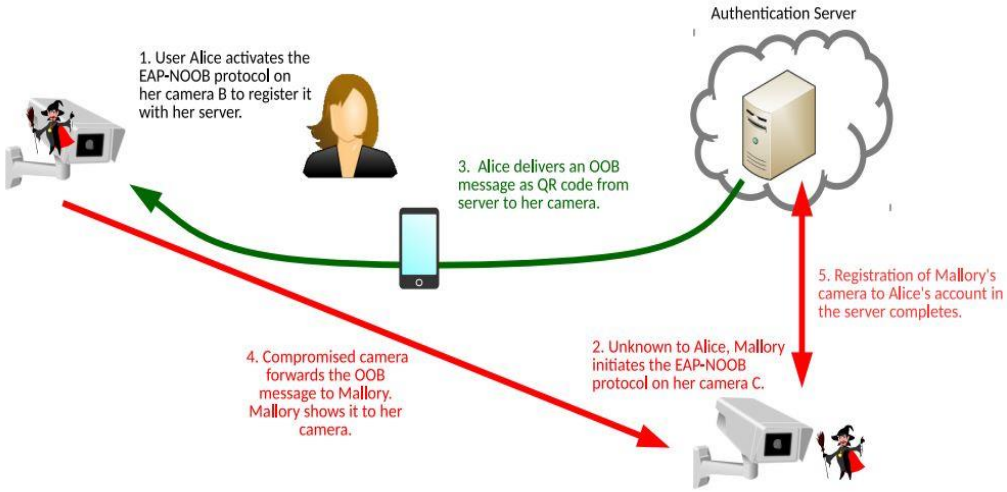

Figure 11: IoT Threat Attack Scenario [112]

Figure 11 illustrates the threat scenario of the possibility of a compromise to the local IoT coming from a wiretapper with its target on the user's camera and the authentication server constituting the sensor and the sink node, respectively.

In the next subsection, the cryptographic system for data confidentiality is presented.

2.6.5 Systems Confidentiality and Cryptography

[113] proposed a cryptographic scheme that operated on a cloud-based architecture to ensure devices, users and cloud elements were protected from unauthorized access. In their approach, a cryptographic security model that integrated the generic algorithm methodology in generating keys for encryption and decryption to ensure user data was preserved and protected from unauthorized third parties was proposed.

In [114] a deterministic data integrity check mechanism for provable data possession is implemented for cloud based IoT system. The proposed system adopted a cryptographic approach based on the Rivest-Shamir-Adleman (RSA) protocol to verify outsourced data stored on the cloud. Their system operated on fog computing architecture where only the message tags are stored on the data owner's device for efficient verification process. The system protects unauthorized parties from tampering the data.

In the next subsections, the following presentations are made.

- System Integrity and Hashing
- Security Threat Model Approaches
- Authentication
- Authentication Schemes in IoT

2.6.6 Systems Integrity and Hashing

There are available security solutions today that improve the integrity of data in IoT communication. To assure integrity of data communicated from one node to the other node in an IoT environment, hashing functions are employed to provide cryptographic hashing to provide message integrity to data. During hashing, a key exchange algorithm is used to establish a set of keys among the communicating nodes. The sending node uses a hashing

algorithm and applies the plaintext and key as input parameters to generate a ciphertext. The key management protocols support a pre-session key generation of a shared secret common to the devices in the communication loop. An effective key management protocol provides a collision-resistant message authentication code. In IoT, system integrity is achieved through the use of an efficient and effective hashing algorithms and secure key management protocols [115] [116] [117].

2.6.7 Security Threat Models

Understanding potential security threats that a system is likely to suffer from, and how an attacker can compromise a system due to certain vulnerabilities that may be present in an existing system is important to guide in the designing phase of a network. The knowledge from potential security vulnerabilities that an attacker might use to compromise a system will offer the basis for appropriate mitigations to be integrated into the system at the design phase. The design phase provides flexibility to make alterations to the system than when the system is developed and deployed.

Threat modelling is a standard method of detecting possible security risks such as vulnerabilities or lack of defines structures and prioritizing mitigation strategies in order to safeguard sensitive data [118].

Threat modelling frameworks in the context of internet of things environment exits to outline and categorize threats and their effects on networks with the goal of providing an effective documentation with details including detection, analyses, classification, and prioritization of the threats. In the context of internet of things, there are several threats modelling that describe the potential security threats and their effects [119]. The procedure in modelling threats involves identifying the assets, outlining the architecture, breaking down the

application, identifying the threats, classifying, and structuring threats, and rating the severity of the threats [120].

During the threat modelling, the whole solution is examined by analysing the security and privacy features of the system, the features of the system whose failures are security relevant and could serve as a weak link for attackers to compromise the system. Similarly, the features that touch a trust boundary for the system is studied and evaluated during the threat modelling of the solution.

Threats are modelled based on approaches. Threat modelling approaches for IoT include STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation), LINDDUN (Linkability, Identifiability, Non-repudiation, Detectability, Disclosure of Information, Unawareness, Non-compliance).

The various elements in the architecture diagram for the system is susceptible to a variety of STRIDE threats.

Processes are subject to spoofing, tampering, repudiation, information disclosure, denial of service, and elevation (STRIDE) attacks. Data flows are predisposed to tampering, information disclosure, denial of service (TID) attacks. Similarly, the data stores are prone to tampering, information disclosure, denial of service, and repudiation attacks. In addition, external actors or entities that may be interacting with the system are liable to spoofing, repudiation, and denial of service (SRD) attacks.

Prioritizing the security for the solution, the system has been zoned and segmented to help isolate damages and restrict any impact from one zone to another zone. To prioritize security solution for a system, the system architecture will have to be segmented into the devices zone, field gateway zone, cloud gateway zone, and services zone. Trust boundaries separate one zone from the other.

In the next subsection, authentication mechanisms are presented.

2.6.8 Authentication

Authentication is a security mechanism that verifies users, processes, or device's identity before granting access to information system resources. Message authentication is a security quality in validating the integrity of a message. Digital signature algorithms are used to uniquely generate corresponding digital signature for messages. In classical data communication, device authentication mechanisms have evolved significantly. From the classical one time password, where a device is assigned a set of stringed characters to grant access to a network, the use of two-factor authentication (2FA) where exactly two access tokens for example password and a phone text or code communicated through the email are required for user identification to successfully gain enrolment to a network; to multi-factor authentication (MFA) technique that uses more than one access token for verification of a user or devices. Single sign-in authentication involves using a single window particularly an email to access a package where a user uses their email to gain access to a third-party application without necessary creating a new log in account for that third party. For example, using a google email account to access non-native or non-proprietary application of google. Whereas password protected authentication is not time-bound but case sensitive and can provide a measure for authenticating a user or device, two factor or multi-factor authentication uses another authentication mechanism in addition to the password. The use of the second authentication mechanism introduces another layer of protection to the stored data. This second authentication method is usually timed and has a brief time to live before expiring. This second authentication protects systems against unauthorized access and tampering of stored data.

There are variety of authentication protocols in use today. The following are the popular ones:

- Password Authentication Protocol (PAP) is the classic protocol with the use of a static password.

- Challenge Handshake Authentication Protocol (CHAP) refers to the challenge-response protocol whereby the challenge usually in the form of a random number is sent and the response depends on a password.
- Lightweight Extensible Authentication Protocol (LEAP) provides a password-based mutual authentication with one-way hashing. It is a proprietary solution proposed by CISCO and chosen by other vendors because of its ease of implementation on 802.11 equipment.
- Extensible Authentication Protocol - Message Digest 5 (EAP-MD5) is a password-based challenge-response protocol with one-way hashing. It does not authenticate the server.
- Extensible Authentication Protocol – Transport Layer Security (EAP-TLS) is a mutual authentication based on SSL. It uses both server and client certificates and incorporates dynamic and distribution of symmetric keys for confidentiality of subsequent exchanges between applications over the internet.
- Extensible Authentication Protocol – Tunnelling Transport layer Security (EAP-TTLS) is server authentication with a server certificate, client authentication with PAP, CHAP or MS-CHAPv2. It incorporates dynamic generation of symmetric keys.

- Extensible Authentication Protocol - Subscriber Identity Module (EAP-SIM) is a mutual authentication using the SIM chip of the GSM (2G). It incorporates dynamic symmetric key generation during the authentication process.
- Extensible Authentication Protocol - Authentication and Key Agreement (EAP-AKA) is a mutual authentication that uses the SIM chip of the UMTS (3G) environment, remains compatible with the GSM (2G) environment and incorporates dynamic symmetric key generation.

Advanced Encryption Standard-Counter mode with CBC Mac Protocol (WPA2 uses AES-CCMP). It corresponds to the 802.11i standard and requires a hardware upgrade. AES-CCMP is considered more secure than RC4-TKIP. It is WPA2 in shared key mode with CCMP encryption (based on AES). It is becoming more common for business use.

Simple Authentication and Security Layer (SASL) adds authentication support to connection-based protocols.

Web Services Security (WSS) provides integrity, confidentiality, and authentication of the source of SOAP messages (several cryptographic architectures supported). Simple Object Access Protocol (SOAP) It is web communication protocol for exchanging structured information over computer networks.

Single Sign-On (SSO) is an authentication scheme that permits a user to log in with single ID to all the severally related, yet independent software systems or web applications. Thus, a user can use a single set of username and password credentials or even multi-factor authentication to access multiple applications.

Generic Security Service Application Program Interface – (GSSAPI) describes in a very generic way the provision of security services based on Kerberos which is the authentication protocol that provides the most single sign-on (SSO) function in a distributed environment.

Transport Layer Security (TLS), formerly SSL (Secure Sockets Layer) is the protocol used to ensure the confidentiality of exchanges (HTTPS, FTPS, POPS) with an "s" for

secure. It is implemented on servers and allows users and devices to be authenticated using a server certificate or a user certificate. The TLS version 1 protocol, authentication by certificate, corresponds to SSLv3 (version 3). In an 802.1X environment EAP is a generic protocol that only indicates that the data carried in the protocol is data useful for authentication. This data is formatted according to the chosen EAP method. Thus, EAP makes it possible to determine which authentication method is used. The 802.1x is an IEEE networking authentication protocol standard specifically for port-based network access control on wired and wireless access points [121].

In the next subsection, is the presentation of internet of things authentication schemes.

2.6.9 Authentication Schemes in Internet of Things

There are several authentication mechanisms available for internet-of-things network today. Popular IoT message authentication protocols include the following:

- Common Authentication Methods
- Password Authentication Protocol (PAP),
- Authentication Token,
- Symmetric-Key Authentication, and
- Biometric Authentication.

These mechanisms attach an information tag from a source node to a destination node for the purposes of validating data provenance. IoT message authentication schemes are expected to produce collision-resistant authentication codes as well as a computationally lightweight processing overhead considering the resource-constrained nature of IoT devices for ensuring data provenance [122], [123], [124].

Authentication techniques in IoT have evolved to include modern disruptive technologies that promises improved security against known cybersecurity attacks that wireless sensor networks have been predisposed to in recent times. The broad categories of authentication methods for IoT comprises shared secrets, one-time password, token involvement, intrinsic authentication, behavioural, and next generation authentication techniques. Share secrets authentication methods involve the use of static password. Time based, challenge-response based, out-ot-band transmission , and locked based methods of authentication are the several types of one-time password authentication techniques. The token-based authentication systems are sub-divided into software-based tokens and hardware-based tokens. The secure socket layer (SSL) or certificate exchange, key exchange, and third-party tokens authentications are software-based token authentication methods. The hardware-based token authentication systems include connected tokens, connection-less tokens, and disconnected tokens. Hardware connected tokens authentication techniques include smartcards, USB keys, and Key Fob.

There are also the intrinsic, and behavioural authentication methods. The kinds of authentication methods that make up the next generation authentication mechanisms including the Fast Identity Online (FIDO), Cryptophoto, and Blockchain [125].

The next generation authentication methods used multi-factor authentication systems that combine the traditional authentication and other advanced authentication schemes to improve

resilience against the security vulnerabilities of classical authentication methods. Human properties like biometric is used in addition with existing authentication method and communication standards like software and hardware tokens in the trusted platform module (TPM) for authenticating IoT devices. The universal two-factor authentication (U2F) is particularly used in the FIDO method of authenticating IoT devices. The FIDO authentication requires a physical form of identification. The cryptophoto authentication method relies on a two-factor authentication structure for two-channel mutual authentication. In the cryptophoto authentication method, a random photo is sent from the physical token device of the client where the client selects the photo and a one-time authentication code to complete the authentication. The cryptophoto authentication method works by allowing a client to detect identical images sent to their physical token devices including computer and hand-held devices. It offers a strong security protection against man-in-the-middle attacks, phishing, social engineering, and hijacking. The resource constraint nature of IoT devices makes the choice of the cryptophoto authentication system unsuitable due to the computational overhead required in the processing of images [126].

Blockchain authentication technique allows machine-to-machine authentication without requiring password. Distributed ledger technology that is based on blockchain to record and track past transactions forming blocks and storing the transactions in a cryptographically linked blocks to form a chain of transparent history of transactions used blockchain-based authentication technique [127]. The classical authentical scheme that adopts a consensus mechanism to validate transactions to be stored in the blockchain involves complex computational overheads. Blockchain authentication schemes do not require human intervention for deployment. It requires directly, machine -to-machine interaction for deployment [128].

Internet of things authentications protocols are categorized based on several factors including the methodology of the authentication architecture, the IoT layer for the deployment, the token orientation used for the authentication, the identity and cryptographic primitive context used, the authentication factor levels, and the unique strengths and weaknesses of the authentication scheme to be used. The similarities and differences among the popular authentication schemes for internet of things are captured in the table below.

Table 7: Comparison of existing authentication schemes in Internet of things [129].

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
Ahmad Abusukhon, Zeyad Mohammad, Ali Al-Thaher (2021)	Perfect forward secrecy and a digital signature scheme that is based on zero knowledge prove for	Centralized	Enhanced Multiple session key protocol, Elliptic curve Diffie-Hellman	Scyther Simulator	The provision of a protocol that provided multiple session key per session	Multiple key generation per session and	Centralized implementation of the architecture

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	validating the authenticity of messages						
[7]	The encryption scheme does not involve tokens and is deployed on the application layer	Centralized and hierarchical	Encryption			The encryption architecture involved two-way packet encapsulation that relieved the data overhead of data resources	The implementation context for the authentication scheme did not involve blockchain technologies.

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
[8]	The application of the security mechanism involved the encryption and hashing of data occurred on the application and network layers of the IoT layer stack.	Centralized and hierarchical	Encryption and hashing mechanisms using asymmetric RSA hashing algorithm		The provision of a low overhead and improved interoperability of cryptographic mechanism for authentication.	No tokens were used. It did not involve the datagram layer and transport security tokens for authorization from the authentication server.	The implementation of the cryptographic primitive involved an encryption that used a symmetric asynchronous one-time password

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
							protocol. The user datagram protocol used by the security mechanism led to unreliable communication.
[9]	The authentication mechanism is implemented and deployed	Centralized and hierarchical	Cryptographic primitive was based on an Encryption and a symmetric		A security mechanism that is resilient against replay, and denial-of-	The security protocol involved two-way entities for its	The implementation of the cryptographic primitive

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	on the physical layer.		asynchronous one-time password for authentication.		service (DoS) attacks.	implementation	involved an encryption that used a symmetric asynchronous one-time password protocol. The solution did not require blockchain technology for

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
							its deployment.
[10]	The authentication protocol was deployed on the network layer to secure attacks against path detection and capture of IoT node	Distributed and hierarchical architecture for implementation	The cryptographic solution was based on symmetric encryption. The node-ID, indices of space, and seed were parameters used			It depended on three-way entities for its deployment	The security solution is resource demanding since it needed excessive cost of energy in establishing encryption keys

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
			for the encryption of messages				
[11]	The security mechanism for authentication was based on a symmetric encryption and implemented on the network layer	Distributed hierarchical architectural	Symmetric encryption with polynomial ID tokens			The methodology for the deployment of the solution required three-way entities	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
[12]	An authentication method that is deployed on the application and network layers of IoT network to ensure malicious entry of unauthorized nodes are	Distributed and hierarchical methodology	Asymmetric encryption with information token		Secured security layer to protect unauthorized nodes from gaining access to the network. The cryptographic primitive involved a key-based	Two-way entity	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	avoided				asymmetric technique		
[13]	The cryptographic mechanism was deployed on the application and network layers of the IoT network	Centralized and hierarchical architecture	Asymmetric key-based encryption algorithm using formID tokens			Two-way entities were engaged in the provision of appropriate security.	
[15]	The security	The	Symmetric		The security	Two-way	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	mechanism was deployed at the network and perception layers	cryptographic solution was deployed on a centralized or flat architecture	encryption using XOR		solution provided authentication for RFID tags with readers	entities involving encryption and hashing algorithms	
[16]	The network and perception layers of IoT hosted an authentication scheme that involved	Centralized architecture was used	Symmetric encryption and hashing using a nonce token		The mechanism provided resistance against replay attacks, man-in-the-middle	Two-way entities	The computational cost involved in the security implementation is slightly higher.

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	encryption and hashing of data				attacks, impersonation attacks, privileged insider attacks for smart cards		
[17]	The cryptographic primitive involved asymmetric encryption	Centralized and hierarchical methodology was used for the design and deployment of	Asymmetric encryption using the elliptic curve cryptographic algorithm		The security solution was resistive to denial-of-service attacks, replay attacks,	Two-way function that was based on the ECC algorithm	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	using the elliptic curve cryptographic algorithm. It was implemented on the network and perception layers	the cryptosystem.			eavesdropping, node capture, man-in-the-middle attacks		
[18]	A public key cryptosystem implemented	Centralized architecture	The RSA, ECC and cryptographic		Improved performance against man-in-	Two-way entities were involved	The cryptanalysis of the security

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	on the application , network, and perception layers		protocols were used in the public key cryptographic mechanism		the-middle (mitm) attacks.		solution did not consider denial-of-service (DoS) and replay attacks
[19]	The targeted layers on which the security solution was deployed were the application,	Distributed architecture	The asymmetric cryptosystem was based on encryption protocols and hash function using the ECC.			Two-way entities	It required high memory that directly resulted heavy computational overheads.

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	network, and perception layers.						The security system is not resistive against node capture attacks.
[20] [21]	The transport layer security protocol was used to design a cryptosystem that was based on the	Centralized and hierarchical architecture	TLS communication protocol		The communication protocol was resilient against replay and impersonation attacks.	One-way entity that assured secure communication over IoT network	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	OAuth2.0 framework. The security solution was deployed on the application layer.						
[22]	The cryptosystem was to provide security at the application	Centralized methodology involving a server that provided the	Encryption with key management protocols based on the		It provided a security layer for data against MITM, brute force attacks,	One-time entity Encryption protocol	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	layer. It employed simple authentication scheme for encrypting data	necessary authentication for all the clients	Advanced Encryption Standard (AES), and Diffie-Hellman Encryption (DHE)		and side channel attacks or timing attacks. It involved two severs		
[23]	A cryptosystem comprising a combined cryptographic primitive of an	Centralized and hierarchical methodology	Symmetric encryption using the AES		The security solution provided resistance against split	Two-way entities	

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	encryption algorithm and a hash protocol deployed to address security vulnerabilities at the perception layer				attacks. A security protection against incidences of occurrence of detaching and swapping tags from products.		
[24]	The authentication	Distributed architecture	No encryption and hash		A resistive mechanism that	One-way	It required high storage space.

References	Issue	Structure	Cryptographic Primitive Identity/Context credentials	Tool	Objective	Evaluation Parameters	Limitations
	method used non encryption protocols and hash algorithm but involved a token deployed at the application layer.		protocols were directly involved, tokens were used		is based on a secret sharing scheme against replay, DoS, Eavesdropping and MITM attacks		

Table 7 shows the comparisons of existing authentication methods for internet of things

In the next subsection, the conclusion on internet of thing is presented

2.6.10 Conclusion on Internet of Things

Access, connectivity, and data sharing capabilities have enabled special-purpose devices like temperature sensors and industrial production control systems to measure physical quantities, analyse, and report the environmental situations of these sensors to immediately undertake actions like sound alarms, turn valves on and off, switch lights, control opening and closing of doors. The technical designs and continuous operations of the main actors of internet of things constrains operational energy, available memory for computation, and security. Modern enterprise computer networks have integrated ordinary objects with connectivity capabilities to expand their coverage to environments that are restricted and offer unfavourable operational support for the classical network devices. Due to the resource-constrained nature of devices used in IoT networks, they become easy targets in vulnerability attack situations. Designing an appropriate security solution for IoT systems require appropriate mechanism to provide balance between a lightweight solution that does not offer any security efficiency trade-offs.

In the next subsection, the conclusion for the state-of-the-art is presented.

2.7 Conclusion

The state-of-the-art research conducted in this section have provided a summary of available knowledge and research output which form the foundation for comparing, contrasting, and synthesizing the proposed solution in this thesis. The accumulated knowledge extracted from these sources were discussed. Even though existing blockchain solutions for data authentication in wireless sensor networks have increased in recent times, there is limited research on authentication mechanism for an IoT architecture that involves the active

participation of the sensor, sink node and cloud. Other blockchain-based solutions for cyber-physical system, limited the blockchain use to physical network elements only. Additionally, there is limited literature on an architecture that actively engaged all key actors namely the sensor, sink node and cloud components in the implementation of cryptographic primitives and blockchain smart contract for authenticating IoT data. The state-of-the-art showed an important gap in existing solutions that involved the use of blockchain-based solution to authenticate IoT data between actively connected network elements of the sink node and the cloud. Although available research papers have covered solutions for improving node data security in internet of things to assure the privacy, integrity, authorization, and availability of data, the use of blockchain-based cryptographic security solution that is implemented based on an architecture that included a direct involvement of the cloud and active nodes within sink node clusters from connected local IoT networks has not been researched on. The state-of-art also highlighted the absence of research on the formalization of blockchain-based authentication mechanism that directly involved the sensor, sink node and cloud as active network elements for a blockchain-based IoT architecture.

The next chapter begins the second part of the documentation that covers the contribution of designing a blockchain-based architecture for IoT data authentication. The chapter contains the methodology that defined the formalization, modelling, and simulation for the validation of the critical properties of a security mechanism for IoT data authentication. This contribution formed the blueprint for the proposed solution.

PART II: Contribution -Newly Proposed Blockchain Architecture and Implementation Features

Chapter 3: Blockchain-Based Architecture for IoT Data Authentication

3.1 Introduction

This chapter focuses on the methodology forming the framework for the design of the blockchain-based cryptographic application for authenticating IoT data. It outlines the blueprint for the proposed system by designing the architecture, describing the components and their properties, the methods, and functions that each component possesses and the interactions between the various components.

The architecture adopted for the system constitutes a heterogeneous system with three subsystems. These subsystems integrate and connect into the other components to provide an overall cyber-physical network security solution.

Specifically, the chapter covers the following subsections:

- Design Assumptions
- The Architectural view of the System
- The Description of Architectural Components
- Description of the working of the System
- Conclusion

The design assumptions are presented in the next subsection.

3.2 Design Assumptions

The proposed system has the goal of registering a sensor to collect data and transmit the data to the cloud. The sensor and the sink node undergo a registration phase using the Public Key Infrastructure (PKI). The registration allows both the sensor and the sink to acknowledge each other before allowing IoT data to be transmitted between them. The sensor collects the data from its environment and sends the data to the sink node initially. This data will later be sent to the cloud for storage. The data from the sink node is stored on the cloud using a peer-to-peer distributed storage mechanism between the internal IoT gateway persistent storage (IoT gateway internal memory) and the cloud. The devices used in the system are sensors, sink nodes, IoT gateway, and cloud. The data from the sensor node must be authenticated and passed for integrity compliance before it can be stored on the sink node. The data on the sink node is validated based on a Blockchain consensus before the data is stored in the form of blocks onto a distributed ledger that run on the IoT gateway internal memory and the cloud.

We made some assumptions in our design model.

- That a sensor node is an unreliable IoT end device that wants to gain access to the network
- The sink node forms an integral part of IoT architecture and wants to gain control of the network
- The IoT gateway coordinates and maintains sink nodes from other local IoT subsystems and the sink nodes from all the subsystems are clustered and managed using consensus mechanisms to form a fully decentralized authority responsible for authenticating sink nodes and validating messages from sink nodes for storage on a distributed ledger.
- The communication channels between the sensor end device and sink nodes could offer a distrusted path where an attacker can use to compromise the integrity of the

data. Similarly, the communication channel between the sink nodes and the cloud is susceptible to attacks where an attacker can listen to the communication between the sensor and the sink node as well as the communication between the validator nodes and the cloud, respectively.

The design assumptions provide scenarios to suggest that IoT devices, data, and IoT architecture are susceptible to sophisticated collaborated attacks since modern cyber-attackers employ such collaborated attacks in compromising systems. Such design assumptions are critical because they provide useful information to guide the design process of producing a relevant and effective solution. The goal of the research is to design an appropriate security mechanism for the authentication of IoT data. Similarly, the stated assumptions were indirectly extracted from the state-of-the-art. These assumptions formed the basis upon which certain kinds of cryptographic primitives were selected in the design of the blockchain-based authentication mechanism to address the modern security challenges that current IoT systems are susceptible to.

In the next subsection, the architecture view of the proposed system is presented.

3.3 The Architecture View for the System

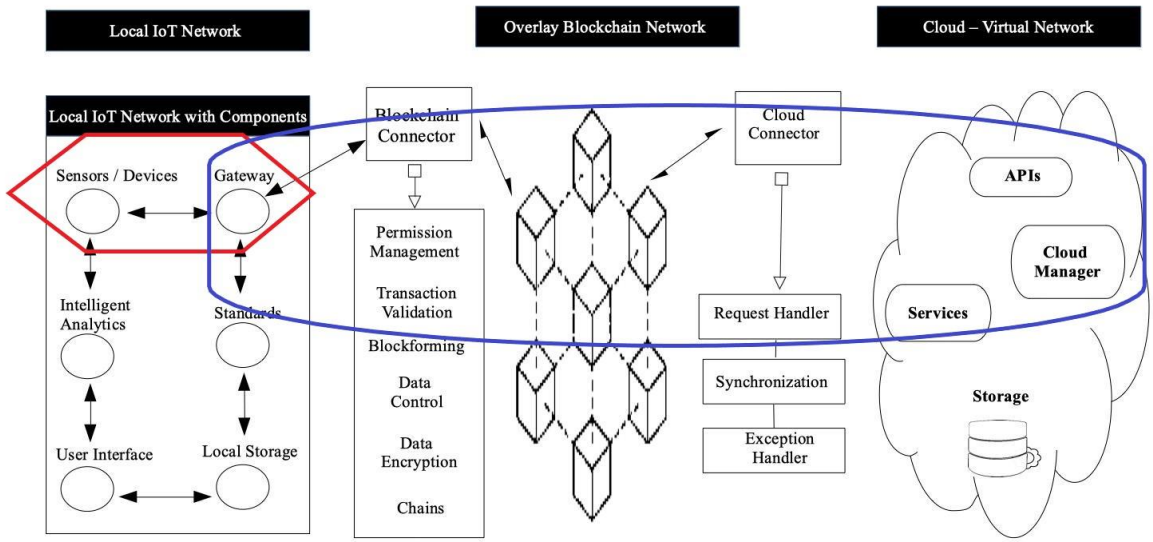

Figure 12: The blockchain-based IoT Architecture [130]

In Figure 12 is the representation of the initial architecture that was adopted and modified to reflect the specific challenges that the proposed system will be addressing. This architecture in its current form has a lot of gaps that have been highlighted using the shapes in red and blue.

The red shaped hexagonal shape that enclosed the sensor devices and the gateway elements within the local IoT network depicts the missing details of how the sensor and the sink nodes are registered to allow them exchange data safely. The blue shaped plain figure that encompassed the gateway and the cloud equally did not show how the physical local network element (IoT Gateway) and the cloud storage were connected and managed to allow for IoT data transmission between them.

In the modified architecture for the proposed system as seen in [130], there are additional intermediary elements: the Public Key Infrastructure (PKI) introduced between the sensor and the sink node to represent how the sensor and the sink node are registered. Similarly,

there is a blockchain smart contract between the sink node and the cloud to show how the physical element within the IoT local network and the remote or cloud network is connected.

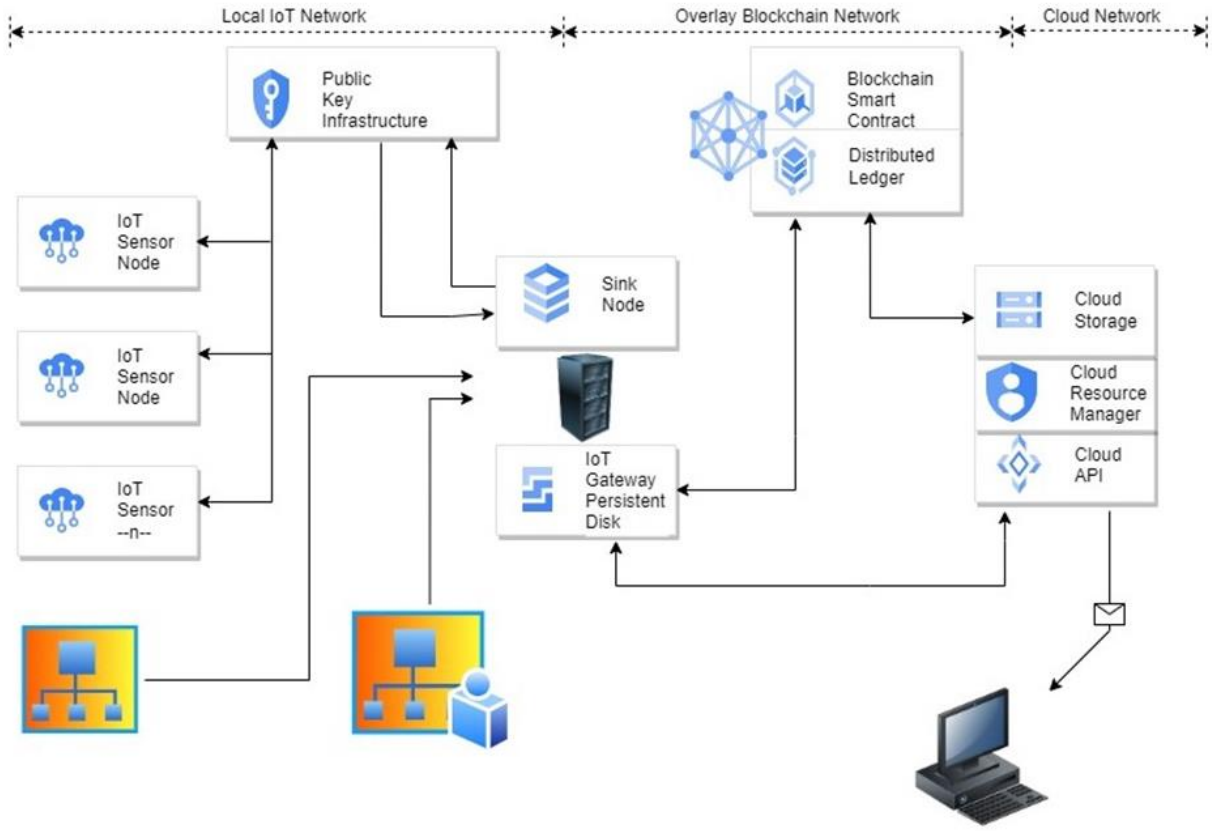

Figure 13:Architecture of the Whole System [Researcher's Own Architecture]

In Figure 13 the architecture displaying the key components of the system is presented.

The proposed architectural model comprises three distinct subsystems. Each subsystem has key components that perform a set goal for that subsystem. There is a physical component that comprises a local IoT network with several sensors connecting to a sink node. The local IoT network is also referred to as on-site network. Similarly, there is a cyber network that is represented by the cloud network. The cloud network is also known as an off-site network. Finally, there is a virtual overlay network that connected the physical and cyber networks by creating a channel for authentication of IoT data for secure transmission between the sink

node and the cloud. The overlay network is a distributed ledger technology (DLT) that relies on blockchain-based cryptographic mechanisms for authentication of active parties that is IoT gateway node and the cloud. There are two distinct authentication methods present in the architecture. A centralized authentication system that is deployed within the local internet of things network and a decentralized authentication scheme implemented in the DLT that constituted a peer-to-peer network between the local IoT network and the cloud network.

The local IoT network consists of sensitive end devices and a dedicated base station node which is a sink node. The sensitive devices represented sensors whiles the dedicated base station represented sink node. The sensors collect critical data involving physical measurements from its immediate environs and communicate the critical data to the sink node.

The local IoT network used key-based credentials that is based on public key infrastructure in assigning and attaching key pairs with sensors and sink nodes to ensure the creation of encryption and generation of digital certificates in managing encryption. The key pairs are the private and public keys. The public key infrastructure is implemented on the sink nodes and operationalized through three component mechanisms: certificate authority, registration authority and validation authority to facilitate the secure connection by registering and authenticating sensors for secure communication of critical messages between the sensors and the sink node.

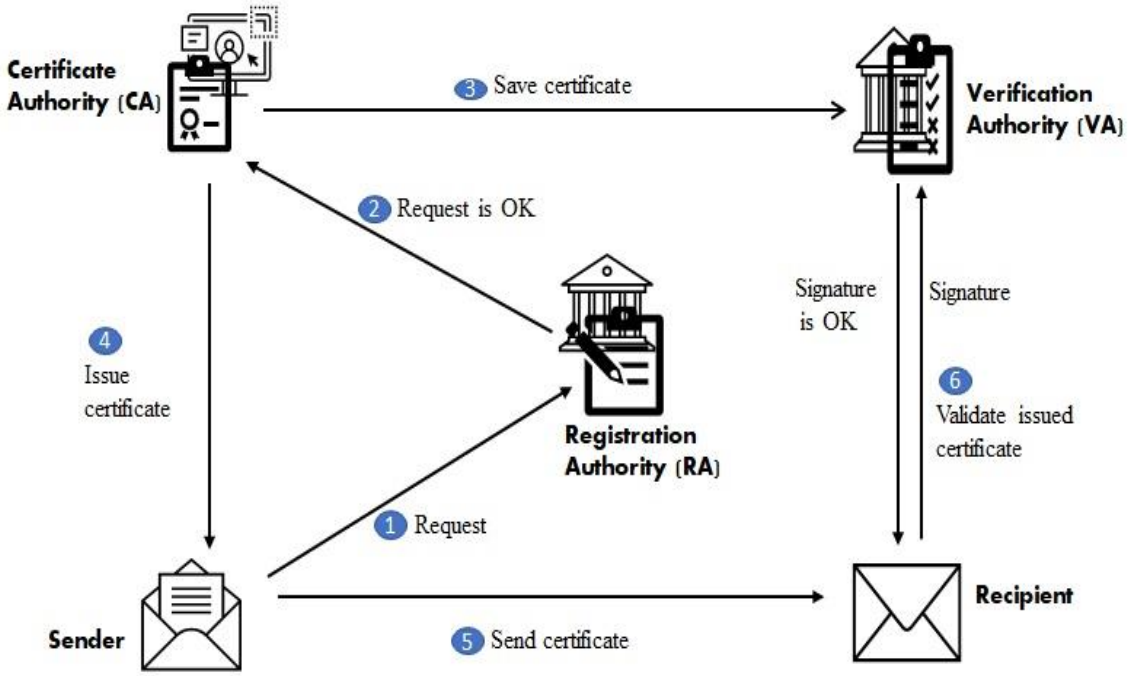

Figure 14: The Public Key Infrastructure

Figure 14 illustrates the data flow for the public key infrastructure used for establishing and maintaining authentication that assures privacy and security of communication between sensors and the sink node.

The certificate authority (CA) is responsible for registering new sensors and issuing certificates to bind the sensors to the sink node. The certificate authority ensured appropriate registration and enrolment of sensors. The registration authority (RA) is responsible for the identification and authentication of communicating end nodes (sensors and sink nodes) request to subscribe for new digital certificates or reusing old certificate as well as overseeing the approval or rejection by such subscription requests. Similarly, the registration authority is responsible for sensor enrolment to approve or reject the digital certificate credentials of sensors.

The validation authority (VA) provided appropriate mechanism for entity replacement of the certificate authority for verifying and validating a digital certificate. The public key infrastructure used these three mechanisms for approving or rejecting new sensor

subscriptions through verification and authentication of digital certificate that is essential for message communication between the sink and the sensors to commence.

The public key cryptographic scheme employed in the local IoT network is implemented using a centralized authentication approach to allow several sensor end devices to be authenticated and authorized to send privately collected data from its environment to a centralized dedicated sink node. The data from the sensor is first encrypted and a corresponding digital certificate is created based on the node data. The encrypted message and the digital signature are communicated to the sink node as a first step for a later transmission to the cloud. The working of the three subsystems complements each other to integrate into the general operation of the entire solution. The local IoT network, an overlay blockchain network and a virtual cloud network are the three subsystems in the architecture. The overlay blockchain network connects and regulates the direct communication between the local IoT network and the virtual cloud network. The local IoT subsystem represents an onsite IoT network that comprises sensors as end node devices that have computational capabilities, energy, and storage limitations. Each of the sensors are represented using unique identifiers (IDs) on the network. The sensors are also assigned device addresses. For instance, [D_ID_1, D_ID_2, …, D_ID_n] and [D_Addr_1, D_Addr_2, …, D_Addr_n] represents the sensor unique identifiers and device addresses, respectively.

There is also a sink node present in the local IoT network. Sink node have enhanced computational capabilities, storage, and energy than the sensor devices. The sink node manages the connection of sensors, perform device authentication, and provide connection for external networks. The sink node run a centralized authentication mechanism that involve a trusted third-party application for managing the connected sensors. Sink nodes also have unique addresses that represent their location, and their cardinality in that location. The assumption here is that the local IoT spans a larger area, and the large area is grouped into

locations. Each location is bounded by a jurisdiction for a particular goal and business target towards collecting data for that business jurisdiction. The device address for the sink nodes for example is [$L.ID_n$_$AddrID_n$]. Thus, the location identification ($L.ID_n$) and sink node device

address identification ($AddrID_n$) The jurisdiction and IDs are assigned hexadecimal values.

For instance, sink node identifiers will be [$L_1.ID_1$, $L_1.ID_2$, …, $L_1.ID_n$] and [$L_n.ID_1$, $L_n.ID_2$, …, $L_n.ID_n$] to denote the location identifications for location 1, sink node identifier 1, location 1, sink node with the last identifier represented as N; and the last location with identifier N with sink node identifier 1, up to the last assigned identifier, which is represented as N, respectively.

Each sink node within the local IoT network assigns, coordinates and manages all the sensors that are connected to it. It authenticates and authorizes the connected sensors to collect and communicate data from its environment to be verified and stored to the cloud through the sink node.

In the next subsection, the architectural components are described

3.4 Description of Architectural Components

- Local IoT Network

In a local IoT network, there are sensors that collects various data from their immediate environment. The sensors are of different communication frequencies, sensitivity, operational modes, and operation-environment (fluids, air, light, sound, heat, etc). Within the local IoT network, centralized authentication and authorization approach is used in managing device registration, connection, and service requests. The sink nodes adopt blockchain smart contracts that is decentralized in managing the connection between the local network and the

cloud network. The local IoT network is organised into multiple cluster networks. A single node among a cluster of sensors is elected as a cluster head CH to coordinate the direct communication of member nodes. The cluster head sink node in the architecture is represented as an IoT gateway. The IoT gateway node performs the aggregation of the packets received from all the sink nodes in that cluster. This approach optimizes better the transmission energy of nodes than multi-hop forwarding of data transmission towards the sink node.

The local IoT network consists of the following zones.

- The device environment provides a connection between sensors and sink nodes that is distinct and isolated from the public internet. Peer-to-peer communication within the local IoT network is achieved by using short-range wireless radio technology.
- The field gateway zone includes all the clustered sinks nodes that are connected to the gateway. The field gateway is outside the local IoT network and dedicated to data processing. It has limited operational redundancy and liable to physical intrusion attacks. The field gateway zone has two distinct interface areas. The inside and outside interfaces. The inside face provides the interface for attaching clustered sink nodes. The outside face offers the interface for external parties including the cloud or remote storage.
- The cloud gateway zone provides a cloud-based control and data analysis system that enables remote communication from and to devices or field gateways from numerous sites across the public network. The cloud gateway zone refers to remote data processing and storage system that is not on the same site of the devices and the field gateway. The cloud gateway integrates device control system, remote processing, and storage capabilities. It comprises the cloud

gateway, the field gateways and devices that are directly and indirectly attached to it. All external parties communicate through a distinct surface area provided by the edge of the cloud gateway zone.

- Cloud Network

The cloud storage provides services with dedicated space on remote servers that offer features such as application programming interfaces (APIs), cloud manager, storage, and cloud services as Infrastructure as a service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS), Database a as a Service (DBaaS).

- Overlay Blockchain Network

The overlay blockchain network subsystem is based on smart contract that involve consensus nodes and IoT gateway nodes to implement decentralized authentication of IoT devices and data authentication for data communication between sink node and the cloud. The use of blockchain tool for sharing sink data with the cloud is implemented using a distributed ledger technology (DLT). The distributed ledger technology (DLT) has the following features [131].

The blockchain distributed ledger technology layers and their definition:

- Application layer is responsible for generating rule-bases and program codes for smart contract, chain codes, and atomic swaps. It provides a platform for users, nodes, and external applications to interact with the distributed ledger technology. The application layer supports down-ward two-way communication for inter DLT interaction and upward communication through application programming interfaces (APIs) and oracles for external communication with third party applications.

- Execution layer serves as the repository for the rules and policies that define the program logic for smart contract, and chain code for the running of the DLT. The execution layer works directly with the application layer in software applications to trigger the codes and rules contained in the execution layer towards the execution of a transaction across concerned parties. The execution works with on-chain and off-chain databases by activating internally stored oracles from the application layer to retrieve data from the on/off-chain database sources to the execution layer code.

- Consensus layer forms the fulcrum of the distributed ledger technology (DLT) and blockchains. The primary goals of the consensus layer are to synchronise and establish agreement among distributed nodes, as well as guarantee a fault-tolerant operations in validating transactions. The distribution of trust, control, and ownership of digital assets within the blockchain is expedited by the consensus layer. Consensus formation is based on either proof-based or voting based. This architecture supports sink nodes from several geographical locations and from different networks that must agree to validate the integrity of sink node data through consensus. The consensus nodes are the various sink nodes from the numerous local IoT networks and the IoT gateway that support assist the transmission of verified IoT data to the cloud.

- Data Model layer oversees the creation and maintenance of blocks, ordering services, chain structure, data, and time synchronization within the blockchain ledger. It is responsible for ordering transaction into blocks, appending blocks onto the distributed ledger and replicating the updated ledger across the network

whiles preserving a common state of ledger across all the concerned nodes. The data model layer provides the creation of orchestration processes for distributed ledgers and databases.

- Network layer supports the communication of node data as transactions, between the sink node and the cloud. The communicating peers constituting the consensus nodes and the cloud use protocols and methods that are defined in the network layer to allow messages to be digitally signed, verified, and validated for chain forming of transaction blocks. Protocol suites such as the Transport layer security (TLS) are defined in the network layer for establishing secure handshaking.

- Infrastructure layer provides the interface platform for correspondence between the virtual and physical software agents representing the active blockchain nodes. The active nodes constituting the infrastructure layer perform cryptographic operations relating to digital signature generation and verification, validating nodes and messages based on consensus rules and policies. Access control mechanisms are employed through the blockchain membership service provider to validate the identity of nodes, as well as provide permission for nodes chain signing.

- The blockchain virtual layer contains additional components such as the:
    - Smart Contract Layer - Responsible for processing transaction requests and determining if transactions are valid by executing business logic. The transactions in this context represent

validated messages from sink nodes. The validated message from sink node is hosted on the IoT gateway node.

- Communication Layer - Responsible for peer-to-peer message transport between the nodes that participate in a shared ledger instance.
- Data Store Abstraction - Allows different data-stores to be used by other modules.
- Crypto Abstraction - Allows different crypto algorithms or modules to be swapped out without affecting other modules.
- Identity Services - Enables the establishment of a root of trust during setup of a blockchain instance, the enrolment and registration of identities or system entities during network operation, and the management of changes like drops, adds, and revocations. It also provides authentication and authorization.
- Policy Services - Responsible for policy management of various policies specified in the system, such as the endorsement policy, consensus policy, or group management policy. It interfaces and depends on other modules to enforce the various policies.
- APIs - Application Programming Interfaces enables clients and applications to interface and interact with the blockchain.
- Interoperation - Supports the interoperation between different blockchain instances.

In the next subsection, the description of the working of the system for request-response activities by the sensor and the sink node is presented.

3.5 Description of the working of the System

The system has two main operational orientations for undertaking request-response activities. The centralized authentication that adopts the client-server approach, and the decentralized authentication approach that rely on distributed consensus mechanism to validate messages for storage in a distributed ledger. A centralized management orientation is employed to ensure request-response activities between the sensors and the sink node where the sensors constitute the clients whereas the sink node becomes the server. On the other hand, the decentralized authentication is based on the distributed consensus mechanism for validating sink node messages for storage in the distributed ledger that is operated in a peer-to-peer approach between the IoT gateway and the cloud

By default, a sensor is placed in an unregistered state. An unregistered sensor cannot send data to a sink node. A sensor starts by requesting registration from a sink node. The registration of a sensor is conducted through the certificate authority and registration authority components of the public key infrastructure. A registration is either accepted or refused. A registered sensor is enabled to collect data and prepare it for transmission. The sink node has a list of parameters including the available number of sensors that can be registered. Each registered sensor is assigned a unique identifier.

The registered sensor is now placed in a position or state of collecting data, encrypting the data, and signing the data before transmission to the sink node.

The Public Key Infrastructure (PKI) receives sensor registration request with sink nodes. It accepts a registration or refuse a registration request.

The sink node by default is in an unregistered state. An unregistered sink node cannot transmit it content with the cloud. The PKI registers sink node to enable sensors to connect to it. The IoT gateway connects sink nodes from different local IoT sites and locations. The registration of a sink node involves checking for conformity with expected set of parameters to validate the integrity requirements of a valid sink node.

The IoT gateway validates a sink node and its data to enable the sink node to transmit the data to the cloud. An unvalidated sink node data cannot be transmitted to the cloud. A registered and validated sink node processes its data for transmission through the IoT gateway to the cloud. The smart contract for transmitting data from sink node through the IoT gateway to the cloud first checks for the integrity and verifies the validity of the sink nodes before triggering the formation of data blocks onto the ledger that resides on the IoT gateway and the cloud.

The blockchain receives validated data from the IoT gateway to be appended onto the digital ledger. Through consensus algorithms, IoT gateway transmits to append validated sink node data to the distributed ledger that is stored on the cloud and the IoT gateway.

The cloud awaits data update on the blockchain from IoT gateway. It receives data updated on the digital ledger. It prepares to receive the next data appending request and transmission in from IoT gateway to be stored.

Concisely, the proposed system offers a blockchain-based authentication scheme based on the public key infrastructure in validating IoT data for transmission from the sensor to the cloud. Towards achieving the goal of migrating IoT data from the sensor to the cloud, public key infrastructure and key-based cryptographic algorithms were used to encrypt and generate digital signature scheme to help validate the integrity of the data from the sensor for onward transmission to the cloud through an IoT gateway. The digital signature serves as a digital

fingerprint for each data that sent from the sensor. The sink node uses the digital signature to verify the integrity of the data from a sensor. A verified data is again aggregated onto an IoT gateway through a blockchain consensus mechanism to be appended onto a distributed ledger that is running on an IoT gateway and the cloud. An IoT gateway aggregates sink nodes from several local IoT sources to form a cluster. Multiple consensus rules and policies are implemented through smart contract and triggered to coordinate the consensus formation and signing of new verified transmitted node data into blocks to be appended onto existing distributed ledger hosted on the IoT gateway and the cloud. Thus, node data is stored permanently using smart contracts onto distributed ledger that is shared between the IoT gateway and the cloud.

In the next subsection, the formal description of the system architecture is presented. The formal description includes the structural description of the system and the behavioural description of the system.

3.6 Formal Description of the Architecture

The formal description of the architecture is composed of the structural features of the system and the behaviour of the system.

3.6.1 Structural Description of the System

This subsection of the document presents the detailed structure of the system using a formal approach that makes it possible to take advantage of automated tools for the implementation of the system.

The structural view of the system is represented to show the components of the system, their attributes, operations, and the relationship of the various components using the Unified

modelling Language (UML) tool. The various instances, interactions, and the behaviour of the components of the system and the dataflow modelling are represented using Class diagram, Activity diagram, Context and process modelling, Interactive diagram, and Sequence diagram.

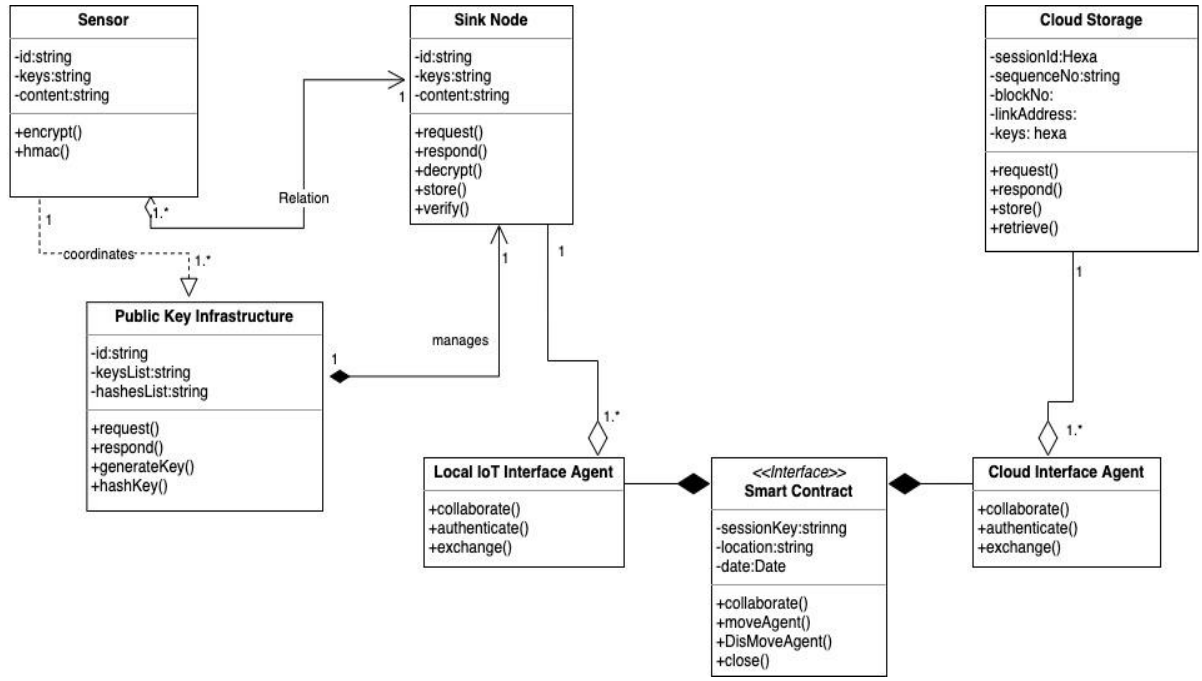

Figure 15: Class Diagram for the proposed system

In Figure 15, the entities together with the attributes and functions, the associations and relationship between the various entities are outlined. It shows the structural overview of the process flow for the entire blockchain-based cryptographic mechanism employed for data authentication. It displays the collection of classes, interfaces, associations, collaborations, and constraints for establishing the hierarchical relationship and associations between the elements in the local IoT network on one hand and the sink node-to-remote storage interaction on the other hand.

The class diagram shows the graphical representation of the static top-level view for the system. The sink node class is composed of Local IoT interface agent that provided the interface for executing the smart contract on the sink node through an IoT gateway and the public key infrastructure class which is responsible for providing the mechanism for encrypting and generating digital signature for validating sensor data by verifying the integrity of messages communicated between the sensor and the sink node within the system.

The sensor and sink node classes constitute the native IoT platform which is representing a local IoT network. The remote storage network is represented by the cloud storage class. The cloud storage class is composed of the cloud interface agent for implementing smart contract agreement between the local IoT sink node representation through a gateway and the cloud storage.

The Blockchain overlay network is represented by the smart contract class. It is composed of the local IoT interface agent class and the cloud interface agent class. The smart contract class oversees agreement policies implemented in a distributed and decentralized manner across the sink nodes and the cloud storage. These self-executing agreement policies are responsible for authenticating data for storage across several devices and locations through a distributed ledger to store and share node data.

The sensor class is responsible for collecting data from the immediate environments of the sensors, encrypting the data, generating a digital signature for each collected data, and transmitting the data to the sink node. It adopts and uses the public key infrastructure (PKI) to encrypt and generate the digital signature for each sensor data. The sink node class is responsible for implementing a centralized authentication mechanism for creating access credentials for sensors towards establishing registration connection requests with the sink nodes to transmit sensor data to the sink node. The local IoT acting through a gateway runs

smart contract agreement policies for the facilitation of decentralized and distributed storage through the smart contract interface class to the remote storage class. The cloud interface agent class is responsible for providing the blockchain cloud connector interface for running the smart contract between the sink node and remote storage class. The smart contract helps with authenticating data storage sharing requests data from the sink nodes through consensus mechanisms implemented on the IoT gateway.

The local IoT interface agent class is implemented on the sink node class for providing interface to receive block forming request through consensus nodes that are led by the IoT gateway. The cloud storage class provides interface through APIs to allow for external entities to interact and share the node data from sensors. The smart contract class also functions to primarily provide a platform for collaborating entities to duplicate and distribute storage between the IoT gateway persistent storage and the cloud. The smart contract class represents the collection of policies and protocols appropriate for connecting sink nodes from the local IoT interface agent class. The cloud interface agent class is responsible for connecting the sink nodes and the remote storage to duplicate by offloading IoT data storage using distributed ledger technology.

In the next subsection, the behaviour with a formal description of the system is presented.

3.6.2 Formal Behavioural Description of the System

3.6.2.1 Formalization Approach

The behaviour of the system is formalized using the activity, interaction, and Sequence diagrams to describe how the system functions in using key schedule and management protocols in registering sensors and sink nodes. The model demonstrates how the sensor collects data from the environment, encrypts and generates a digital signature for the

ciphertext, illustrates how the sink node authenticates sensor data by using the digital signature algorithm to verify the digital signature to ensure data integrity for storage.

In the next subsection, the activity diagrams for sensor data authentication are presented. The activity diagrams for sensor data authentication detail the steps and procedure for generating digital signature by the sensor and verifying the digital signature by the sink node.

3.6.2.2 Activity Diagrams for Sensor Data Authentication

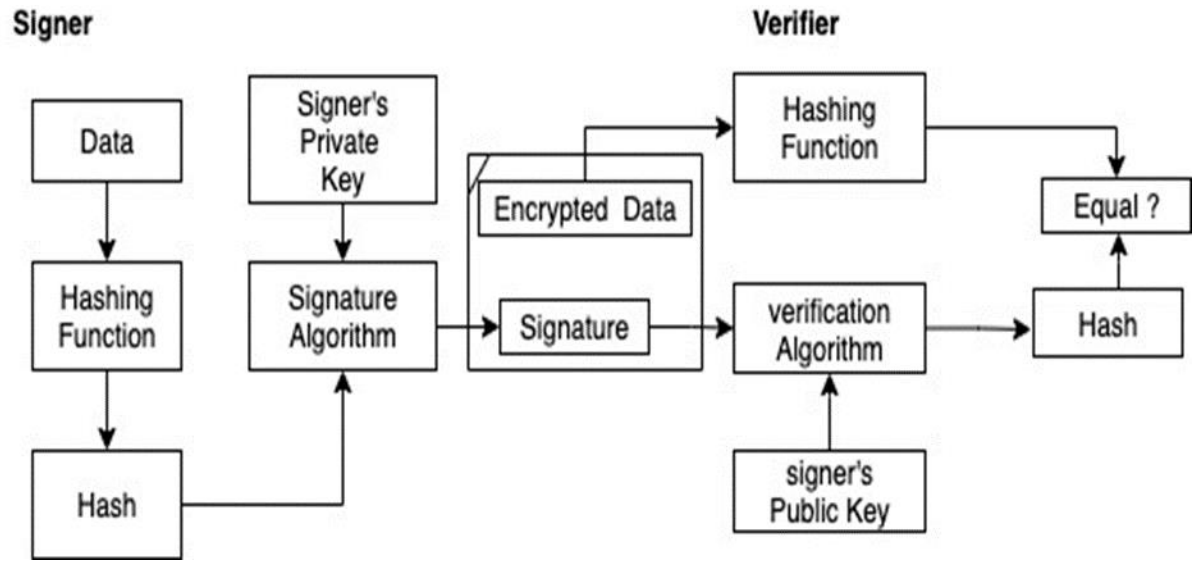

Figure 16: Activity diagram for data authentication

In Figure 16, the flow of controls and activities describing how hash values are created and subsequent digital signature value are generated is presented. A message (plaintext) acting as input into a hash function produces the hash value for the message. The signer (sender) of the message uses a shared secret and private key, the IoT as inputs for the digital signature scheme to produce the digital signature that will enforce data integrity. The encrypted data is

sent together with the digital signature to the receiver (sink node). The sink node verifies the integrity of the received IoT data. The signature that accompanied the message is used for validating the integrity of the message using a verification algorithm component of the digital signature algorithm.

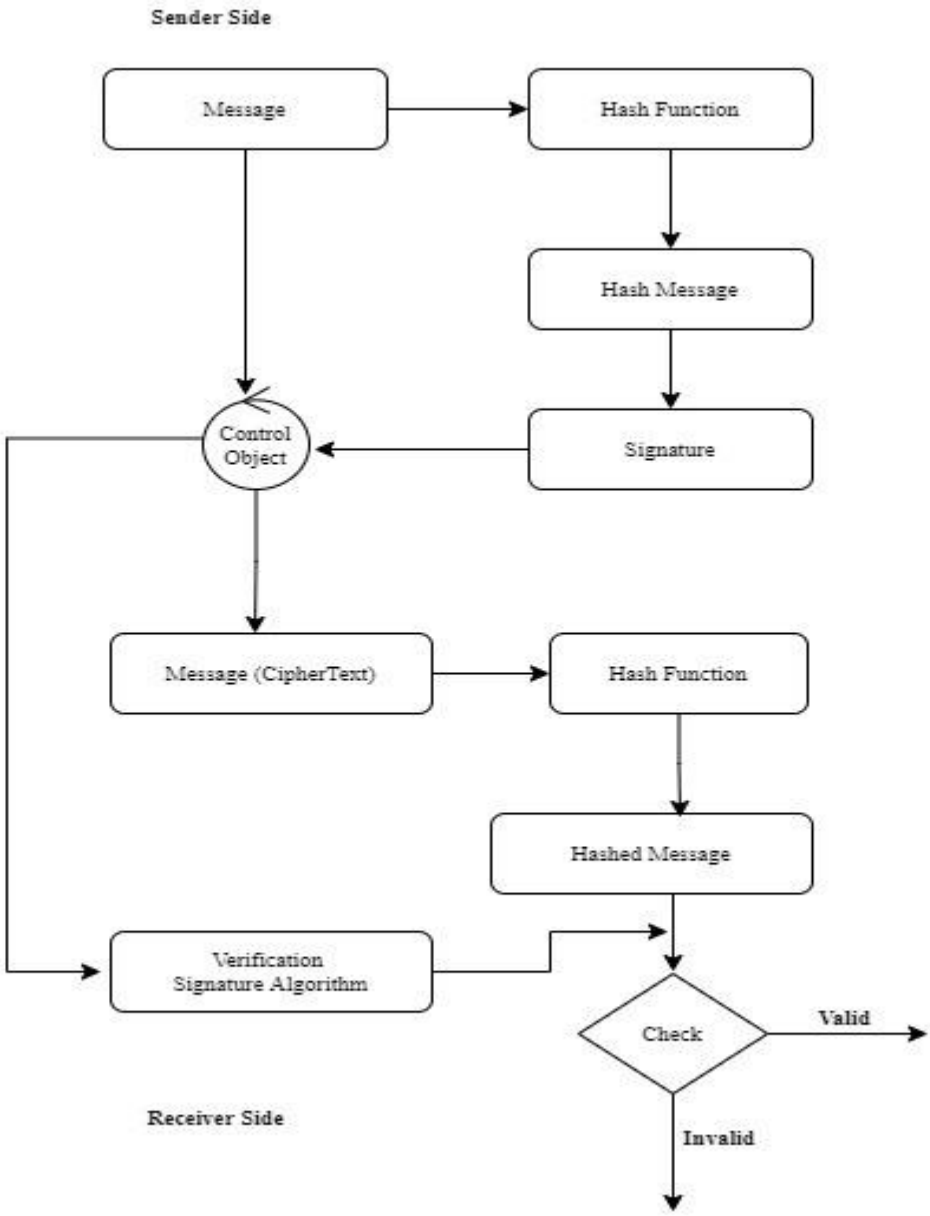

Figure 17: Activity Diagram for digital signature verification

In Figure 17, the presentation of the process of generating the digital signature for messages is illustrated. The hash value, the private key of the sender and the public key of the receiver are applied to a digital signature algorithm to generate the digital signature for the message.

The verification of a digital signature is described in a block diagram. The hash value, and the digital signature are both obtained from the sender end device. The private key of the sink node and the public key of the sensor are applied to the hash through a verification algorithm. The private key of the sensor, the public key of the sensor and the hash is used to generate a new Signature for the received data. The received digital signature (given signature) and the newly generated signature (derived signature) values are compared. If the same digital signature values are obtained, the ciphertext and the signature are retained. Any difference in the signature values will result in the message (ciphertext) and the signature being rejected.

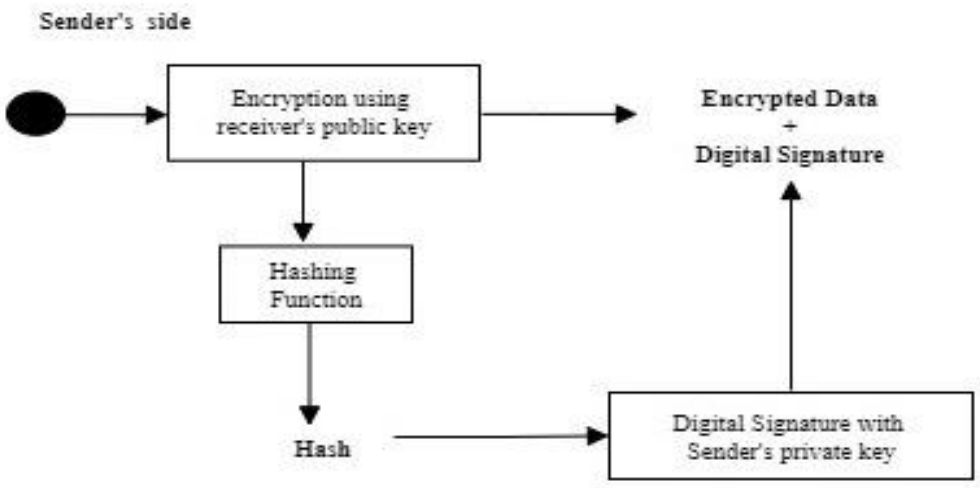

Figure 18: Activity Diagram for the encryption and digital signature

In Figure 18, the block diagram for the various activities that occur at the sensor is presented. It highlights the preparations made in securing the node data before transmitting it to the receiver node. In preparing the node data, the raw message is encrypted and then a corresponding digital signature for the ciphertext is generated.

In the next subsection, the interaction diagram describing the chronological flow for sensor data authentication is presented.

3.6.2.3 Interaction Diagram of the Chronological Flow for Sensor Data Authentication

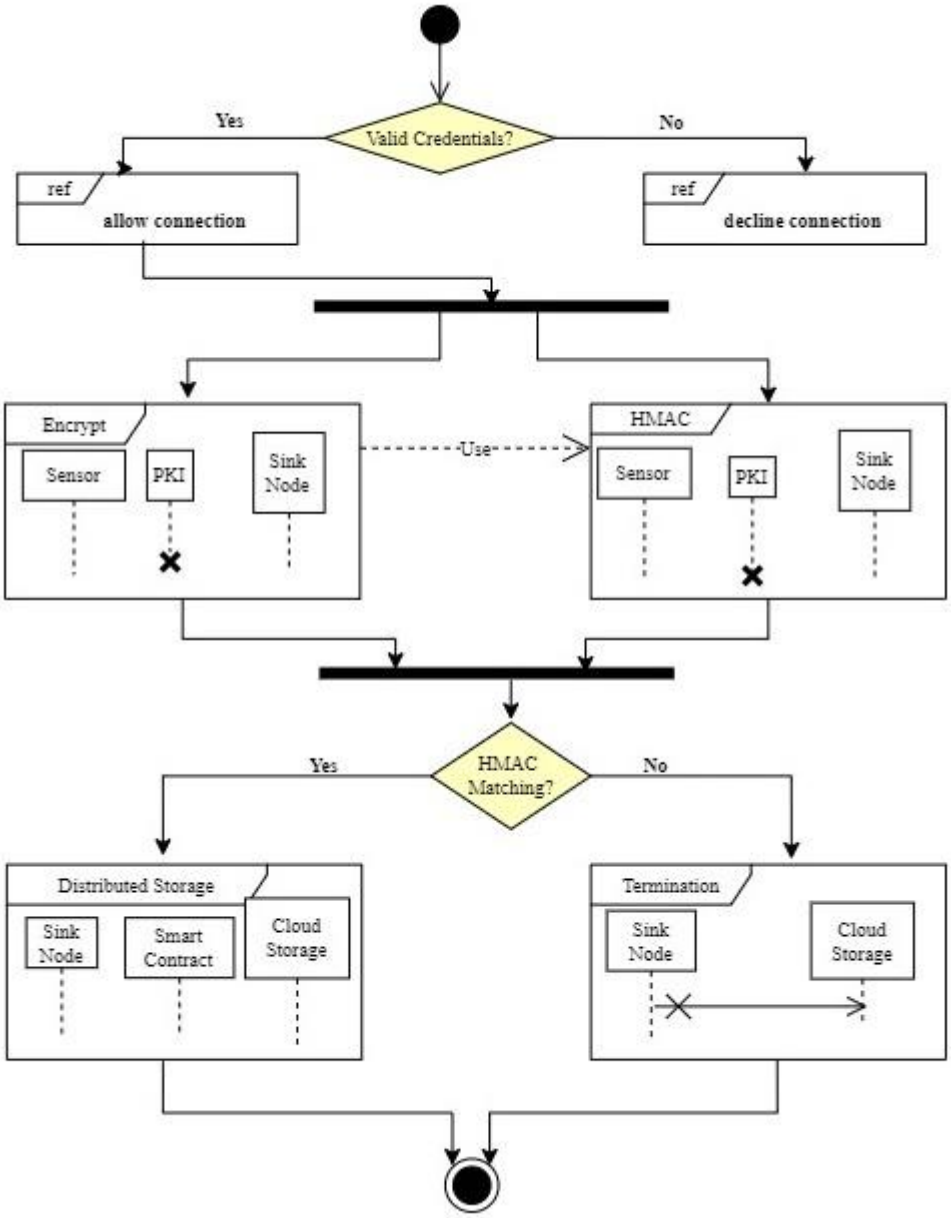

Figure 19: Interactive diagram for the system

In Figure 19, the chronological flow of data stream within the system is presented. The series of activities towards data authentication within the proposed architecture is sequentially outlined. It shows the high-level graphical representation of the cryptographic system outlining the functions and connections between the individual components. The activity is denoted by rectangular shapes or blocks. Each subsystem comprises blocks and their internal interactions.

The interactive diagram is divided into subsystem and each subsystem comprises individual actions that cover the following events: Sensor registration and connection to sink node; Sensor validation; Sensor data encryption and digital signature generation; Communication of ciphertext and digital signature, verification of the integrity of IoT data. Blockchain smart contract use policies and rules on a consensus mechanism to validate IoT data and store the data on IoT gateway persistent storage and the cloud.

In the next subsection, the sequence diagram of the proposed architecture is presented.

3.6.2.4 Sequence Diagram of the Proposed Architecture

The Sequence diagram consists of the components, the interaction between these components and the sequencing of the interaction from how sensors connect to the sink nodes within a local IoT network by first registering with the PKI, to encrypt IoT data and generate digital signature for the transmission of the data to the sink node. The blockchain acts through a blockchain-based consensus mechanism to distribute storage between the IoT gateway persistent storage and the cloud. Additionally, it outlines the various interactive steps or procedure for validating sensors, for a secure connection to the sink node. Thus, it illustrates the sequencing of the messages for subsystem interaction within the entire blockchain-based cryptographic system.

The sequence diagram depicts the aspects of the system that manages asymmetric encryption, hash function application and the generation of a digital signature algorithm. It describes the sequential operations involved in the application of cryptographic mechanism that leverages on the distributed ledger technology (DLT) for the provision of a distributed storage across connected subsystems for an IoT network. The system consists of a fusion of asymmetric encryption, digital signature scheme and blockchain-based smart contract for storing the validated hashes onto the cloud.

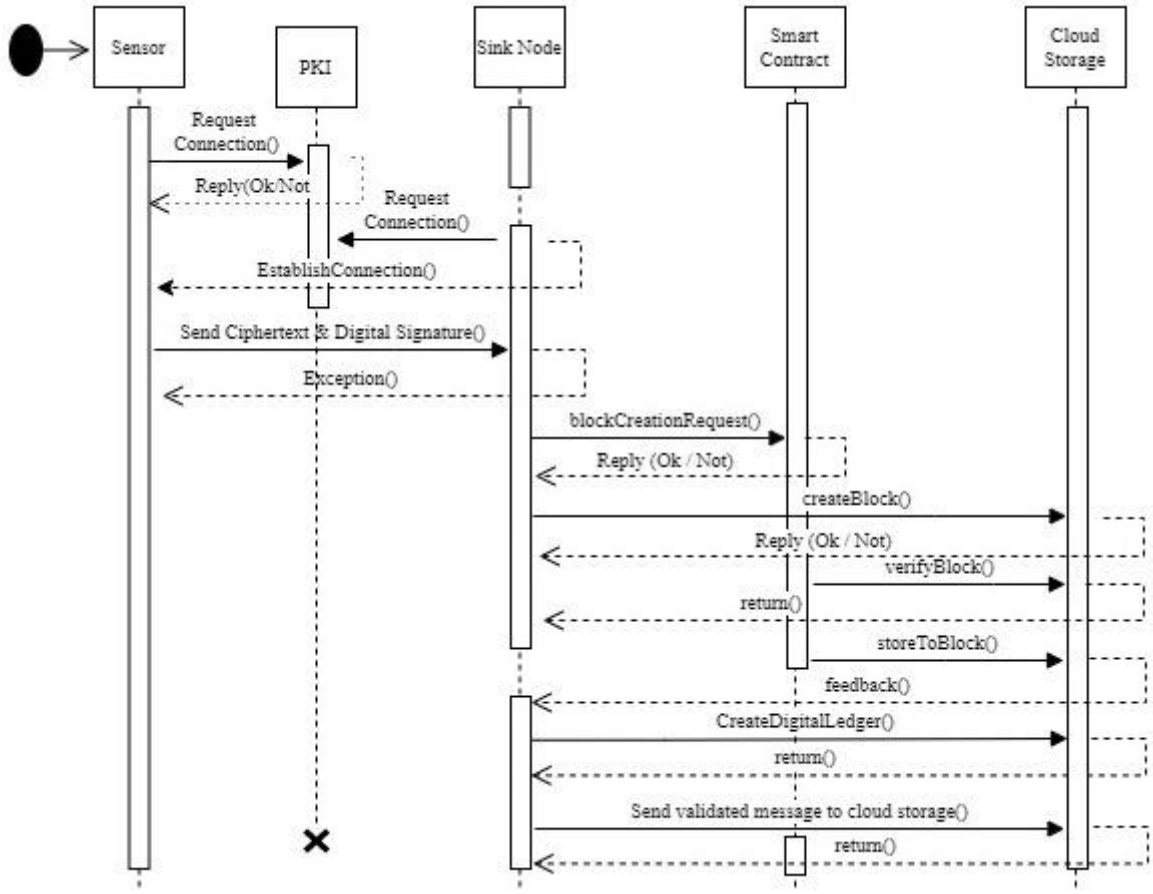

Figure 20: The Sequence diagram for the proposed architecture

In Figure 20, the events and event scenarios with their interactions arranged in time sequence is presented. The events that core actors (sensors, sink node, IoT gateway, and the cloud) generate, the ordering of the events and the interactions between these events are outlined.

The main entities representing lifelines in the diagrams are: The sensor, PKI, sink node, smart contract, and cloud storage. The sensor entity represents sensors. The PKI entity represents a cryptographic subsystem for generating key pairs and managing keys for use in encrypting and generating digital signature. The sink node entity represents base station for aggregating sensor data from the sensor.

- The sensor requests an access token from the PKI entity in other to interact with the sink node. The PKI entity grants the sensor the access token to be used for registering to the sink node. The sink node through the PKI entity regulates connection requests initiated by the sensor node and respond to such request by using session keys and other access tokens necessary for registering sensors for message transmission between the sensor and the sink node.
- The smart contract lifeline provides a platform that uses self-executing contracts and policies to enforce agreements in a distributed and decentralized manner across the sink nodes for consensus mechanism.
- Cloud storage is a service that provides remote storage for the IoT data.

The sequence diagram depicts the data flow controls and authentication mechanisms of how the proposed system manages the capturing of sensor data by the sensors till the data is stored across remote servers on the cloud.

In the last subsection for this chapter, the conclusion for the contribution in the proposed system is presented.

3.7 Conclusion

The chapter highlighted the general IoT security landscape and proposed an architecture to be used for authentication of messages using blockchain technology.

The components of the architectural framework for the entire system were individually described. The Unified Modeling Language (UML) was employed to describe the components of the system.

The motivation for proposing an architecture was necessitated by the non-existence of an appropriate IoT architecture that can support a blockchain-based authentication scheme to fit the objective for the thesis. Existing IoT architecture could not support a blockchain-based authentication mechanism for IoT node data using blockchain. Available IoT architecture employed a centralized authentication approach and that was not suitable towards achieving the intended goal of the thesis. The proposal of a novel IoT architecture that supported hierarchical network was therefore needed to assist the implementation of a blockchain-based decentralized consensus mechanism for node data authentication.

The IoT architecture supported blockchain-based authentication mechanism to allow sensors from different local IoT networks to connect and share critical data across within the hierarchical network. The sensors from these local IoT networks are validated through a network of sink nodes that form the consensus nodes for sensor validation and IoT data authentication. The proposed architecture consisted of a local IoT networks (on-premises network) and a remote network (off-site network).

The sink nodes from the local IoT networks were joined to form a cluster of sink nodes that functioned as the consensus nodes for authentication of node data. In each of the local IoT network, a centralized authentication scheme that employed the public key infrastructure was comprising cryptographic function and digital signature schemes to assure privacy, security, and integrity of data.

The proposed architecture has been partially implemented. The features implemented are presented in the next chapter.

Chapter 4: Implementation Features for the Blockchain-based Architecture for IoT Data Authentication

4.1 Introduction

The various cryptographic primitives used in the system for authenticating IoT data for the peer-to-peer blockchain storage is outlined. The Feistel cipher, the MD5 hash algorithm, the digital signature algorithm, smart contract, and the blockchain consensus algorithm are described

The chapter specifically, covers the following subsections:

- Public-Key Cryptographic Feature
- Hash Functions Feature
- Digital Signature Feature
- Edwards-Curve Digital Signature
- Formal description of the Edwards-Curve Digital Signature
- Blockchain Smart Contract for the Model
- Blockchain Consensus for the Model
- Conclusion

In the next subsection, the algorithm for a key-based cryptographic encryption and decryption algorithms using the Feistel Cipher are presented.

4.2 Public-Key Cryptographic Feature

The key based encryption and authentication scheme is used on the initial communication of node data from the sensors and the sink node that form the fundamental end devices for the IoT.

Algorithm 1: Feistel Cryptographic Cipher - Encryption

Algorithm 1: Feistel Cryptographic Scheme for securing Node Data

---

Encryption (plaintext M, Key k)

1. Split plaintext data block into halves (L_o, R_o)

2. Generate sub-keys to match the number of rounds for the encryption K_0, K_1, K_2, …, K_n

3. Compute the encryption of the separate halves of the plaintext. Using: En (M, K_i)

4. Interchanging the outputs of the sides. $L_{i+1} = R_1$

5. Apply the Round function interactively: $R_{i+1} = L_i \oplus F(R_i, K_i)$

6. Compute the ciphertext C, using (R_{n+1}, L_{n+1})

Algorithm 1 represents the encryption procedure in using the Feistel cipher to encrypt data.

Algorithm 2: Feistel Cryptographic Cipher - Decryption

Algorithm 2: Feistel Cryptographic Scheme for securing Node Data

---

Decryption (Ciphertext C, Key k)

1. Accept the cipher text and the hash of the key

2. Decompose the ciphertext into its halves (R_{n+1}, L_{n+1})

3 Using the sub key for each round: number of rounds for the encryption K_0, K_1, K_2, …, K_n

4. Apply the Round Function iteratively: R_{n-1}, = L_{n-1}

5. Compute the decryption of the separate halves of the plaintext. Using: Dec (C, K_i)

6. Interchanging the outputs of the sides. $L_{n-1} = R_1$

7. Apply the Round function interactively: $R_{n-1} = L_n \oplus F(R_n, K_n)$

8. Compute the plaintext M, using (L_n, R_n)

9. Hash the plaintext using the MD5sum

Algorithm 2 portrays the sequencing of the decryption instruction using the Feistel cipher to decrypt a ciphertext.

In the next subsection, a cryptographic hash function based on the MD5 hash algorithm is presented.

4.3 Hash Functions Feature

Algorithm 3: MD5 Hash Function

Algorithm 3: MD5 Hashing
1. At the source node 2.Generate the Message digest (MD) using the plaintext M, and concatenation of the (ID, SN, Timestamp) as input. 2. Append the MD to the Ciphertext (C) 3. Send the Ciphertext and the Message Digest (C, MD) to the receiver node. 5. At the receiver, decrypt the Ciphertext (C) to get your plaintext (M) 6. Generate a Message Digest with the decrypted text as input 7. Compare the new MD with the one that accompanied the ciphertext 8. MD = MD, the decrypted text is the original text. 9. MD =! MD, the decrypted text is corrupted or have been altered in transit.

Algorithm 3 outlines the steps in hashing a message using the Message Digest MD5 cryptographic hash function.

In the next subsection, digital signature feature of the Edwards-Curve digital signature is presented.

4.4 Digital Signature Feature

4.4.1 Edwards-Curve Digital Signature

The digital signature scheme adopted for the implementation of the work is the Edwards-curve digital signature algorithm.

An informal description of the algorithm is provided in the table below.

Table 8: Digital Signature Components

Parameter Notation	Description and value
p	An odd prime number. The EDDSA uses an elliptic curve over the finite or Galois fields GF(p)
b	An integer with b bits such that 2^(b-1) > p. EdDSA public keys have exactly b-bits. An EdDSA signature has 2*b bits. Normally b is multiple of 8 Number of octets).
A	(b-1) bit encoding of elements of the finite fields GF(p).
H	A cryptographic hash function H producing 2*b bits output.
c	An integer, usually 2 or 3. The base – 2 logarithms of the cofactor.
n	An integer such that c <= n <b.
d	A non-square element of GF(p). Usually, a value nearest to zero that gives an acceptable curve.
a	A non-zero square element of GF(p). For best performance, a = -1 if p mod 4 = 1, and a = 1 if p mode 4 = 3.
B	An element. B != (0,1) of the set E = {(x, y) is a member of GF(p) x GF(p) such that: a* x^2 + y^2 = 1 + d * x^2 * y^2}

Parameter Notation	Description and value
L	An odd prime. [L]B = 0 and 2^c * L = #E. The number of points on the curve (#E) is part of the standard data provided for an elliptic curve E, or can be computed as cofactor * order
PH	A pre-hash function. PH(M) = SHA-512(M). Where PH is the identity function.

Table 8 represents the components with their description for the Edwards-curve digital signature algorithm.

The notation parameter table listed and described the elements in the Edwards curve digital signature algorithm. Each element represented a key aspect of the algorithm [132].

The notation and conventions used:

p	Denotes the prime number defining the underlying field
GF(p)	Finite field with p elements
x^y	x multiplied by itself y times
B	Generator of the group or subgroup of interest
[n]X	x added to itself n times
h[i]	The i'th octet of octet string
h_i	The i'th bit of h
a | | b	(bit-) string a concatenated with (bit-) string b
a <= b	a is less than or equal to b
a >= b	a is greater than or equal to b
i+j	Sum of i and j

i*j	Multiplication of i and j
i-j	Division of i by j
i/j	Division of i by j
i x j	Cartesian product of i and j
(u, v)	Elliptic curve point with x-coordinate u and y-coordinate v
SHAKE256(x, y)	The y first octets of SHAKE256 output for input x
OCTET(x)	The octet with value x
OLEN(x)	The number of octets in string x

The Edward curve digital signature algorithm is a variant of the schnorr signature that is based on the twisted Edwards curve that is based on the twisted Edwards curve. It employs a malleability check during verification which prevents unauthorized third parties from compromising the signature of an existing signature. It offers a fast and secure digital signature for IoT devices nodes.

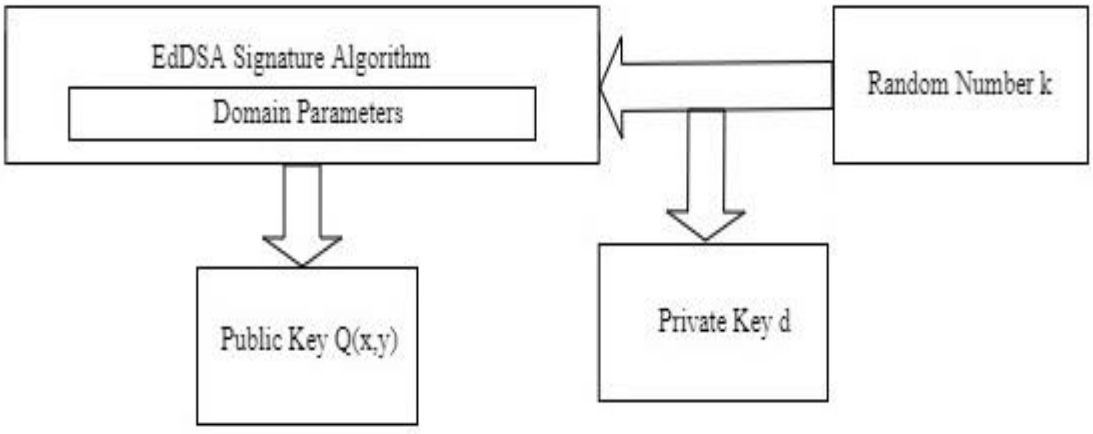

Figure 21: EdDSA key pair generation

Figure 21 represents the process for producing the key pair for the each communicating node. The digital signature scheme adopted for generating and digitally signing messages from the sensor is the Edwards-curve digital signature algorithm (EdDSA). The EdDSA depends on public key and private key to generate and digital signature for verifying the integrity of messages. Each node within the sensor-to-sink node communication loop stores both its private key and public key.

The Edwards-curve digital signature algorithm

Signing Process:

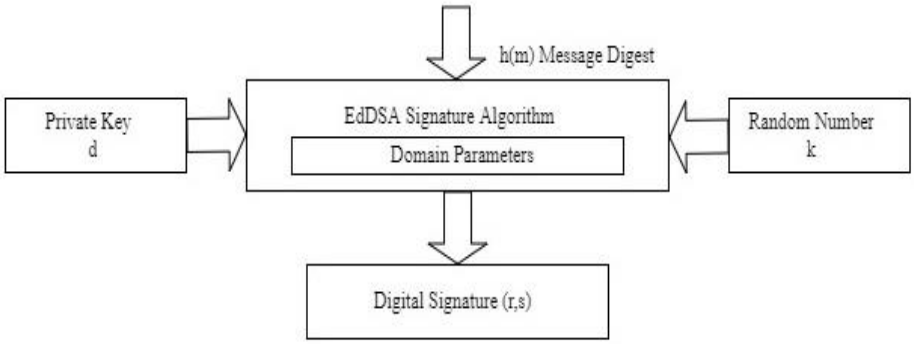

Figure 22: EdDSA Signature Computation Process

Figure 22 depicts the signature computation process for the Edwards-curve digital signature algorithm.

Verification Process:

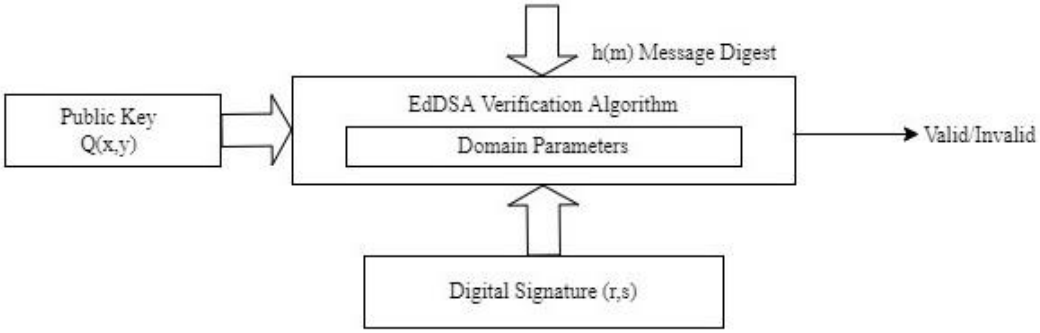

Figure 23: EdDSA Signature verification process

In Figure 23 is the dataflow that represents the signature verification process for the Edwards-curve digital signature algorithm.

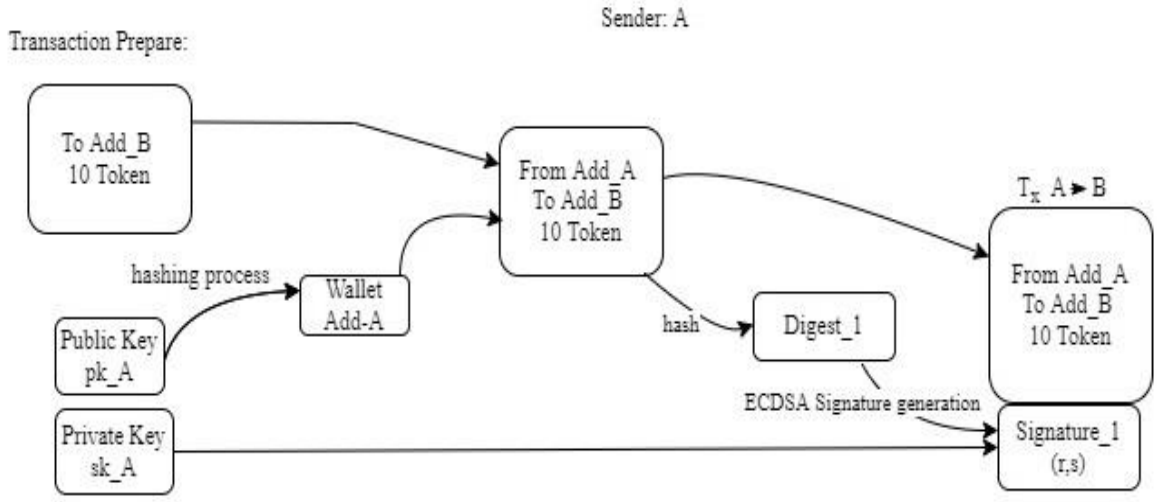

Figure 24: EdDSA Signature generation

In Figure 24 is a representation of the transmission procedure of message from the sink node and the IoT gateway. The sink node is denoted by character ‘A’ whereas the IoT gateway cluster is represented by character ‘B’. The use of the Edwards-curve digital signature

algorithm (EdDSA) is used to authenticate sink nodes from different local IoT networks towards forming a cluster for running a consensus protocol to validate messages.

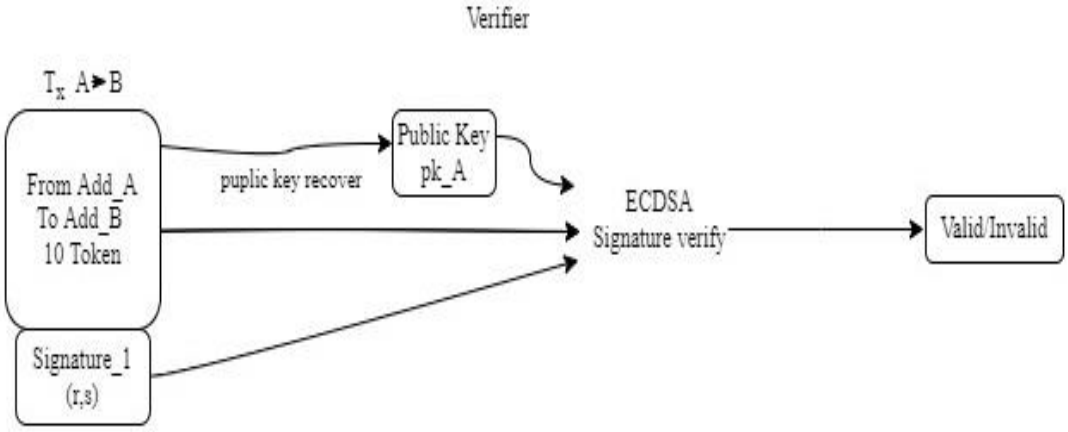

Figure 25: Edward-Curve signature verification

Figure 25 represents the digital signature verification procedure of messages from sink nodes onto the IoT gateway. The sink nodes cluster gateway is denoted by the character ‘B’. The cloud storage is represented by the character ‘B’. The process represents the verification of a sink node prior to joining the consensus nodes.

In the next subsection, the formal description of the Edwards-curve digital signature is presented.

4.4.2 Formal Description of the Authentication Technique

The formal specification and verification of the authentication technique for the system is presented through the formalization of the Edwards-curve digital signature.

The security properties of the Edwards-curve Digital Signature Algorithm (EdDSA) design are defined using formal models. The security property of the cryptographic scheme for authentication of nodal data is based on the Algebraic group model. The security property for the Edwards-curve signature is dependent on the efficiency requirements of the hashing function to satisfy the uniformity of outputs, and the second preimage resistance. Towards that end, the Algebraic group model is used to model the formalization of the Edwards-curve digital signature algorithm.

The deterministic signature property of the EdDSA requires the generation of an ephemeral value r. The ephemeral value is a unique key generated that lasts for the session of the key establishment process in a cryptographic scheme. The digital signature in the EdDSA is created based on the difficulty challenge of maintaining a strong signature that is not easily predictable for the verifier. Hence the output of the hash must be well distributed, no matter the nature of the input such as those inputs with low entropy source.

The second security property requirement for the hash function in the EdDSA concerns the quality of a weak collision resistance where a hash function will not produce the same message digest even for the second input of a different message. Thus, the quality of the hash function to reduce to the minimal level the possibility of generating the same hexadecimal values for a message digest for two different inputs. An attacker sees a signature $\sigma = (R, S)$ on a message m for public key A. If an attacker can computationally use another message m' to produce the same hash H. Thus, $H(R, A, m) = H(R, A, m')$, then σ is also a signature of m'. In such a situation, the hash H has a weak resistance to collision. The two messages have different binary strings: $m \neq m'$.

The algebraic group model is an idealized model for the security analysis of cryptosystems. Adversaries are modelled as algebraic elements using a computational approach in the Algebraic group model [133]. Security implications are proved through reduction.

A group description is a tuple $\Gamma = (\mathscr{p}, \mathbb{G}, \mathrm{G})$ where $\mathscr{p}$ is an odd prime, $\mathbb{G}$ is an abelian group of order $\mathscr{p}$, and G is the generator of $\mathbb{G}$.

$H \implies alg\ G$ for two primitives H and G.

The signature generation and signing procedure using the EdDSA:

The Edwards-curve Digital Signature Algorithm (EdDSA) produces a EdDSA signature (R, s) using the secure hashing Algorithm SHA-255 and curve 25519. The Algorithm helps in signing and verification of message communication or (transactions) [134].

The general form of the curve is:

$x^2 = (y^2 + 1)/(dy^2 + 1)(mod\ p)$

and is specifically defined as:

$$-x^2 + y^2 = 1 - \left(\frac{121665}{121666}\right)x^2y^2\ (mod\ p)$$

With a prime number (p) of $2^{255} - 19$ and the order (L) of

$2^{255} + 27742317777372353535851937790883648493$

The base point (G) on the elliptic curve is: $y = {}^{4}/_{5}$,

The sender (prover) generates a 32-byte secret key (sk) and then creates a public key of:

$pk = sk.G;$

Where G is the base point of the curve.

The sender (prover) creates a SHA-512 hash of her private key:

$h = HASH(sk)$

Creates "r" from the upper 32 bytes of hash and the message m.

$r = HASH(h[32:] \,||\, m)$, where "||" represents a concatenation of the byte arrays values.

$R = r.G$, the sender then computes "s" with:

$s = r + \left(HASH\left(R \;||\, pk \,||m\right)\right).sk$

The signature is: (R, s). The values of "R" and "s" are 32 bytes long, and thus the signature is 64 bytes long.

The verification Procedure:

The receiver (prover) creates S using R, pk, and m:

$S = HASH(R \,||\, pk \,||\, m)$, then next creates two verification values:

$v_1 = s.G$

$u_2 = R + pk.S$

If $v_1 == \; u_1$ the signature checks.

$v_1 = sB = \left(r + \left(HASH\left(R||pk||m\right)\right).sk\right).G = rG + sk.B.(HASH\left(R||pk||m\right)$

$= R + pk.S = \; u_2 \; (mode \; q)$

In the next subsection, the blockchain smart contract feature is presented.

4.5 Blockchain Smart Contract Feature

The overlay network implements a decentralized trust authority based on smart contract to enrol sink nodes that serve as active nodes for executing the blockchain consensus. The smart contract ensures a decentralized access control policy for external users of the node data.

The smart contract manages the user registration, device registration, write and read policy access for hash data storage on the cloud.

Algorithm 4: Smart Contract Pseudo-code

Algorithm 4: Smart Contract Pseudo-code

1: HashMap deviceRegistry(key:ownerAddress, value:List[deviceIDs])

2: HashMap deviceData(key:(ownerAddress, deviceID), value:List[DataHash])

3: HashMap DataAccessRegistry(key:(ownerAddress, thirdpartyAddress, deviceID), value: bool isAllowed)

4: **function** REGISTERDEVICE(ownerAddress, deviceID)

5: InsertToHashMap(ownerAddress, deviceID)

6: **end function**

7: **function** WRITEDATA(ownerAddress, deviceID, Data)

8: **if** *owner == ownerAddress*

9: deviceData([owner, deviceID],List.InsertData(hash(Data)))

10: **end function**

11: **function** READDATA(ownerAddress, thirdPartyAddress, deviceID)

12: **if** DataAccessRegistry(ownerAddress, thirdPartyAddress, deviceID) == true

13: return deviceData[ownerAddress, deviceID]

14: **end function**

15: **function** GRANTACCESS(ownerAddress, thirdPartyAddress, deviceID)

16: **if** *owner == ownerAddress*

17: DataAccessRegistry[ownerAddress, thirdPartyAddress, deviceID] = true

18: **end function**

19: **function** REVOKEACCESS(ownerAddress, thirdPartyAddress, deviceID)

20: **if** *owner == ownerAddress*

21: DataAccessRegistry[ownerAddress, thirdPartyAddress, deviceID] = false

22: **end function**

Algorithm 4 represents the smart contract Pseudo-code

Where:

OwnerAddress: Sink Node Identity (SNId)

Device : Sensor

DeviceID: SensorID (SsId)

The following are the components of the smart contract.

- User Registration. The user registration component of the smart contract symbolizes the registration of users to join the blockchain. Users join the blockchain network after generating a pair of keys that present a private key and a public key that uniquely identify users of the blockchain. The key pairs enable users to interact with the smart contract in device registration and data access.

- The device Registration component allows authenticated users of the system to register their IoT device. IoT device registration involved providing a set of credentials that uniquely identify the device. A hash table is provided in the smart

contract to map the device with its location address that corresponds with the location that is connected to the respective local IoT address on the blockchain.

- Data Write Access Policy. Sink nodes write data to the blockchain after the sink node has been verified with respect to where the sink node is located with reference to the local IoT address to provide the local IoT address , the sensor ID (SsId), and the sink node identity (SNId) prior to writing data to the blockchain. A necessary pre-condition for devices to write to the blockchain is for the device to produce the owner's address and the device id with the data before the device can write the data to the blockchain. The owner address and the device Id are concatenated. The sink node ID, the dataLabel and the Sensor ID are combined to form the key to the individual sink node data in a hash table. (sensorID, sink node ID, data label) = list of device data. The hash table provides a list of hashes of data written by the sink node. The smart contract always checks to verify the sink node ID and the location ID respective to the local IoT to ensure that the sink node is transmitting sensor data that corresponds to the sink node ID before allowing the sink node to write to the blockchain. This sink node checking mechanism ensures that its only verified devices that can execute write operations.

- Device Data Read Access Policy. For a third-party user to have access to a device data, the third-party user needs to be granted the permission to read the data. The distributed ledger employs a write only mechanism through consensus. A user requesting access to a data will have to demonstrate enough permission to access the data by providing the address of the owner of the device and the device ID of the device. The hash table maps the collection of the device owner and address and the

device ID as the key with tuple of the third-party users as values which is maintained within the smart contract. The hash map is always checked to validate the access right status of a requester. All third-party users are registered and stored in the hash map. The access right status is represented as *((owner, deviceID, thirdPartyUseraddress) = bool access)*

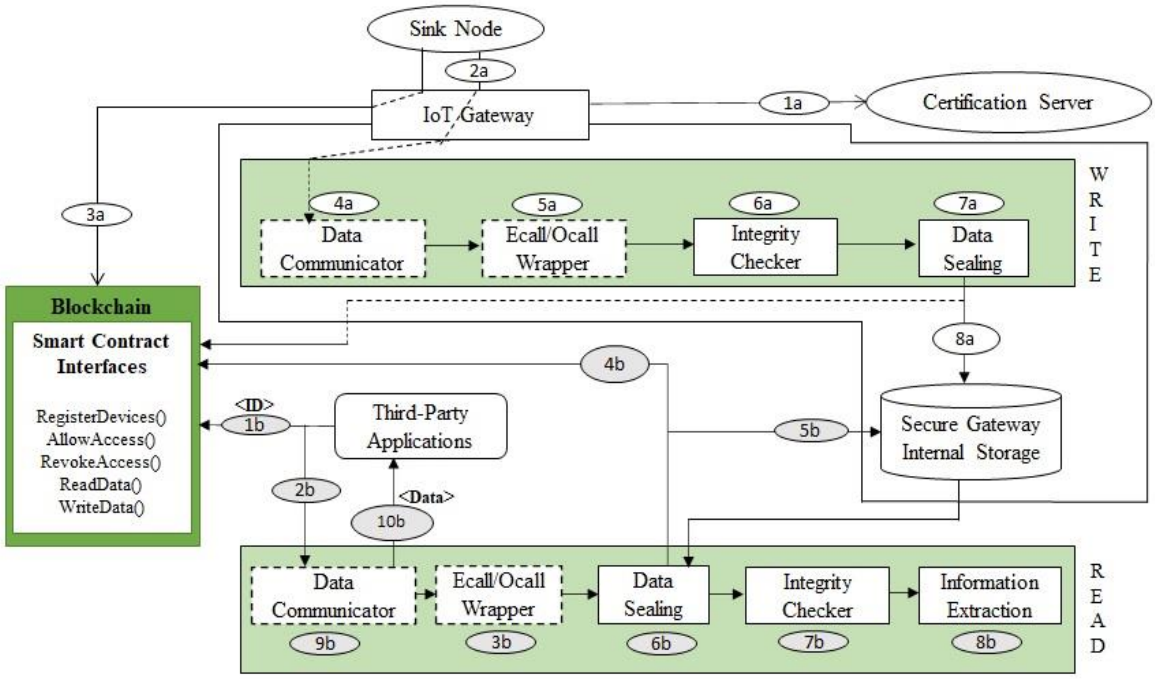

Figure 26: Data flow of the smart contract [135]

Figure 26 provides a data flow of the working of the smart contract that runs on the distributed ledger to ensure users, devices and data are verified and validated for data storage operations onto the blockchain or for reading operations by external users. Data from the sink node gets stored on the distributed ledger through the IoT gateway. The sink node registers itself on the IoT Gateway through the blockchain smart contract RegisterDevices function in step 3a. Before any processing, the IoT gateway first verifies the state of the blockchain by performing a request to a verification server (step 1a). Connected sink nodes constitute the consensus nodes for performing user, device and data integrity checking before either writing onto the blockchain or access data from the blockchain. Data writing operations on the

blockchain is done by the sink nodes, to append a data onto the blockchain which is also referred to as data write operations in step 2a. The *writedata* function in the smart contract is used to append the hash of the sink node data onto the blockchain. The encrypted data is then written to the gateway internal memory using step 4a – 8a. The Ecall/Ocall wrapper communicates with the gateway internal memory as illustrated in the step 5a. The hash of the data from the sink node is verified by recalculating the hash-based message authentication code (HMAC) based on the encrypted and comparing the given hash with the derived hash. The Integrity Checker element verifies and validate IoT data by ensuring that the given hash and the derived hash are the same, the encrypted data is sealed and written to disk in step 7a. If the report from the Integrity Checker shows a difference in the string structure of the derived hash from the given hash, that will result in discarding the data including the hash from the sink node.

Data accessing activities from the blockchain is done using the data read module. A third-party user first registers using the *allowAccess* method with the smart contract. The *revokeAcces* function makes it possible to revoke access for a user. Step 1 outlines the interaction of the third-party user with the smart contract in obtaining the hash of the data generated by the sink node after providing the device ID of the sink node. The smart contract checks if the third-party have the validation necessary to access the data after doing integrity checking. The hash of the sink node data is only returned from the cloud storage after the integrity checker grants the access permission to the third-party user to enable it to access the data from the IoT gateway persistent storage (IoT gateway internal memory) that represents local storage of the data. The smart contract uses the *READDATA* API as illustrated in step 4, to confirm if the third-party user has the access permission to read the data hash identifier supplied by the third-party request. In step 5, it illustrates how data is retrieved from the secured internal gateway storage once data access permission is granted. The data is unsealed

in step 6, and the integrity of the data is checked in step 7, after recalculating and verifying the digital signature by comparing the given and the derived digital signatures. The sensor data stored in the gateway internal memory is read and returned to the user only after the digital signature verification is completed. Steps 9 and 10 illustrate the data flow for this operation.

In the next subsection, the blockchain consensus for the model is presented.

4.6 Blockchain Consensus for the Model

The developed system relied on a blockchain consensus to validate messages from the various sink nodes before storing it on the distributed ledger. The ripple blockchain consensus was adopted for providing the needed agreement mechanism for validating messages from sink nodes onto the distributed ledger.

The ripple consensus protocol required little time in achieving agreement to maintain the correctness of transactions on the network. The ripple consensus algorithm operates in rounds using active nodes as servers. It maintains the identical state of the ledger on all active nodes after reaching an agreement to close a session of a ledger updating operation that involved all active nodes [76]. The consensus protocol of the ripple algorithm attains consensus correctness and protect against byzantine attacks by approving transactions from servers only after 80% of the validating nodes have reached consensus.

The consensus mechanism for the blockchain-based system for IoT data authentication functions using this approach:

i. New validated message from a sink node is broadcast to all nodes within the cluster by a sink node (local validator).

ii. Each sink node receives the broadcast and decides on the validity of the message.

iii. Based on the rules outlined in the consensus algorithm , each sink node tries to validate a newly formed block to be appended onto the blockchain.

iv. Consensus nodes accept or reject a newly formed block based on the validity of the state and history of transactions in the block. A block whose transactions are legitimately formed, and content validated are accepted.

v. Active nodes express their acceptance of newly formed block by initiating the creation of the next block in the chain. The hash of the previously accepted block acts as the previous hash, a critical component for chaining operation of blocks.

In the next subsection, the conclusion on the implementation features of the method is presented.

4.7 Conclusion on the Implementation Features of the Method

The chapter described the algorithms, smart contract, blockchain consensus that formed the implementation features for the blockchain-based architecture for IoT data authentication.

To validate the proposed architecture, the entire system will be modelled, simulated, and verified using Coloured Petri net and CPN tools. The description and visualization of the state transition phases within the system in the form of flow diagrams with guard conditions, synchronizations, actions, and resets will be illustrated in the CPN tool.

In the next chapter, the validation of the proposed solution is presented.

Chapter 5: Validation of the Proposed Solution

5.1 Introduction

This chapter provides the modelling of the proposed system as well as the simulation results and analysis of the proposed system based on the CPN model to validate the solution. The system is formally modelled and simulated using the Petri net mathematical modelling language platform. Critical security properties of the system were simulated using the Coloured Petri net. Coloured Petri nets (CPN) allowed the graphical representation of dataflow represented as tokens to be described within the system.

The various states and the expected actions to be executed from the sensor data till the destination of the data being the cloud was simulated. The cryptographic primitives consisting of encryption and decryption algorithms, and digital signature schemes were represented using CPN places. The various interactions between the components of the system were denoted using transitions from CPN. The simulation and monitoring capabilities of CPN tools were employed to model and simulate the working of the system. The key security properties of the system were validated through the simulation.

The CPN functioned as a visualization tool for representing the graphical simulation of the system. The visualization of the system enabled the animation of the model as well as examine the key aspects of the system relating to the security specifications in the data authentication system. The design requirements of the system were outlined and represented in the modelling of the system.

The simulation of the system involved the visualization of most of the critical internal working regarding sensor enrolment, sensor verification, environmental data collection by the sensor, encryption, and digital signature generation for the collected sensor data within the local IoT network. The dataflow digrams symbolized the registration of sensors using key-

based tokens from the public key infrastructure to assign key pairs to both sensors and sink nodes to enable these end devices to use cryptographic primitives to encrypt and digitally sign node data from the sensor.

All registered sensors are assigned unique credentials to allow it send and communicate data to the sink node. These unique credentials associated with registered sensors are recorded and stored on the sink node. The sink node registers specific number of sensors to enable the sensors collect data and transmit the data to the sink node.

In all, there were four different simulations done. The simulations included, entire system simulation, the local IoT network simulation, the Blockchain network simulation, and the level-0 dataflow diagram simulation.

The entire system simulation involved the registration and connection of sensors using PKI tokens onto the sink node. The registered sensor becomes active and ready to communicate data to a sink node. The active sensor collects data and encrypts the data as well as generate a digital signature for every data value collected from the environment. Both the ciphertext and digital signature are transmitted to the sink node.

The chapter specifically, covers the following subsections:

- Petri-Net Notations
- System Modelling
- System Simulation
- Conclusion

In the next subsection, the Petri-Net notations are presented.

5.2 Petri Net Notations

The CPN tool is a mathematical and graphical modelling tool. The CPN is composed of a collection of directed arcs for connecting places and transitions. Transitions are events that bring about state changes within the system. States within the system are represented by Places. A Place also denotes a system condition. Events or actions are performed by the system when a system condition has been satisfactorily met. The events and actions represent transition. Places are usually accompanied with markings. A marking is a number attached to a place to indicate the number of tokens inside the place. Places are connected to transitions by oriented arcs (arrows).

A transition has both input and output parameters. An input parameter constitutes a condition from a preceding place(s) with available tokens necessary to cause the firing of the transition. The firing of a transition will result in the removal of tokens from preceding places and adding of tokens in succeeding places of that transition. The preceding place arc description serve as an input parameter while the arc description from the transition represents the output parameter that will be used for the firing of the transition. The output parameter to a place represents a change of state.

Formal definition of Colored Petri Nets [136]

A CPN has nine (9) tuples.

$CPN = (\sum, P, T, A, N, C, G, E, I)$, where:

$\sum$ Represents a finite set of *colours* , which are nonempty types.

$P = \{P_1, P_2, \dots, P_n\}$ is a finite set of *places*, which are denoted for the states.

$T = \{T_1, T_2, \dots, T_n\}$ is a set of *transitions*, which are representations for events between the states.

A is a finite set of *arcs*, fulfilling $P \cap T = P \cap A = T \cap A = \theta$ and depicting the combined flow and interaction between the places and transitions.

N is the *node* function, defined from A into $P \times T \cup T \times P$;

C represents *colour* function, defined from P into $\sum$;

G is a *guard* function, defined from T into expressions such as $\left[\forall t \in T: Type\left(G(t)\right) = Bool \wedge Type(Var\left(E(a)\right)) \subseteq \sum\right]$

E is an *arc expression* function, defined from A into expressions such as $[\forall a \in A: Type(E(a)) = C(p(a))_{MS} \wedge Type(Var\left(E(a)\right)) \subseteq \Sigma]$, with p(a) being the place of N(a);

I denotes an *initialization* function, defined from P into closed expressions such as $[\forall p \in P: Type(Ip)) = C(p)_{MS}]$

The graphical representation of Petri net comprises of rings representing Places, rectangles denoting Transitions, arrows symbolizing Arcs.

A coloured Petri net is composed of variables, values, and expressions. CPN objects are described using colour domain that comprises variables, data values, operators, a syntax for expressions, and typing rules. An abstract colour domain consists of : Data values $\mathbb{D}$, Variables $\mathbb{V}$ and Expressions ($\mathbb{E}$) [137].

$-$ $\mathbb{D}$ *is the set of data values*; These data values include integer values, Boolean values (True and False), and special undefined value $\perp$;

$-$ $\mathbb{V}$ *is a set of variables*, that are represented using single letters $x, y, \ldots$, or as subscribed letters $x_1, y_k, \ldots$

$-$ $\mathbb{E}$ *is the set of expressions*, that are composed of values, variables, and suitable operators.

Variables or values may form a basic expression. Thus, $D \cup V \subset E$. For example, let $e \in E$, the expression $vars(e)$ denotes the set of variables from V involved in e.

A $binding$ is a restricted function $\beta : \mathbb{V} \to \mathbb{D}$. Let $e \in \mathbb{E}$ and β be a binding. $\beta(e)$ represents the evaluation of e under β; if the domain of β does not include $vars(e)$ then $\beta(e) \stackrel{\text{def}}{=} \perp$.

Both sets and multisets of expressions are subjected to binding evaluations.

For example, if $\beta \stackrel{\text{def}}{=} \{x \mapsto 1, y \mapsto 2\}$, we have $\beta\{x + y) = 3$. With $\beta \stackrel{\text{def}}{=} \{x \mapsto 1, y \mapsto 2\}$, according to the colour domain, we may have $\beta(x + y) = \perp$ (no coercion), or $\beta(e + y) =$ "12" (coercion of integer 1 to string "1"), or $\beta(x + y) = 3$ (coercion of string "2" to integer 2), or even other values as defined by the concrete colour domain.

Two expressions $e_1, e_2 \in \mathbb{E}$ are said to be equivalent which is represented as $e_1 \equiv e_2$, iff for all possible binding β, the binding for both expressions are the same $\beta(e_1) = \beta(e_2)$. For example , $x + 1, 1 + x \; and \; 2 + x - 1$ are pairwaise equivalent expressions for the usual integer arithmetic.

Definition 1 (Petri nets). A Petri net is a tuple with several elements such as (S, T, l) where:

$-\; S \; is \; the \; finite \; set \; of \; places$; S is also represented as P

$-\; T, disjoint \; from \; S, is \; the \; finite \; set \; of \; transitions$;

$-\; l \; is \; a \; labelling \; function \; such \; that$:

$for \; all \; s \in S, l(s) \subseteq \mathbb{D} \; is \; the \; type \; of \; s, i.e., the \; set \; of \; values \; that \; s$ is allowed to carry,

$for \; all \; t \in T, \; l(t) \in \mathbb{E} \; is \; the \; guard \; of \; t, i.e.$, a condition for its execution,

$for \; all \; (x, y) \in (S \times T) \cup (T \times S), l(x, y) \; is \; a \; multiset \; over \; \mathbb{E}$ and defines the arc from $x \; toward \; y$.

Definition 2 (Markings and Sequential Semantics) Let $N \stackrel{\text{def}}{=} (S, T, l) be \; a \; Petri \; net$.

A marking M and N is a $function \; on \; S \; that \; maps \; each \; place \; s$ to a $finite \; multiset \; over \; l(s) \; representing \; the \; tokens \; held \; by \; s$.

$A\ transition\ t \in T$ is enabled at $a\ marking\ M\ and\ a\ binding\ \beta, which\ is\ denoted\ by\ M[t,\beta\rangle, iff\ the\ following\ conditions\ hold$:

$-\ M\ has\ enough\ tokens, i.e., for\ all\ s \in S, \beta\big(l(s,t)\big) \leq M$;

$-\ the\ guard\ is\ satisfied, i.e., \beta\big(l(t)\big) =$ True

$-\ place\ types\ are\ respected, i.e., for\ all\ s \in S, \beta\big(l(t,s)\big)\ is\ a\ multiset\ over\ l(s)$.

$If\ t \in T\ is\ enabled\ at\ marking\ M\ and\ binding\ \beta, then\ t\ may\ fire\ and\ yield$ a marking $M' defined\ for\ all\ s \in S\ as\ M'(s) \stackrel{\text{def}}{=} M(s) - \beta\big(l(s,t)\big) + \beta\big(l(t,s)\big)$.

$This\ is\ denoted\ by\ M[t,\beta\rangle\ M'$.

The marking graph $G\ of\ a\ Petri\ net\ marked\ with\ M\ is\ the\ smallest\ labelled\ graph$ such that:

$-\ M\ is\ a\ node\ of\ G$;

$-\ if\ M' is\ a\ node\ of\ G\ and\ M'[t,\beta\rangle M'' is\ also\ a\ node\ of\ G\ and\ there\ is\ an\ arc\ in\ G\ from\ M'\ to\ M''\ labelled\ by\ (t,\beta)$.

The definition of marking graphs allows the addition of infinitely many arcs between two markings. If $M[t,\beta\rangle$, there might exist infinitely many other enabling bindings that differ from β only on variables not involved in t. Finally only firings $M[t,\beta\rangle$ such that the domain of $\beta\ is\ vars(t) \stackrel{\text{def}}{=} vars(l(t)) \cup \bigcup_{s \in S}(vars\big(l(s,t)\big) \cup vars\big(l(t,s)\big)$ is considered.

Definition 3 (Petri nets with control flow). A Petri net with control flow is a tuple $(S,T,l,\sigma)\ where$:

$-\ (S,T,l) is\ a\ Petri\ net$;

$-\ \sigma\ is\ a\ function\ S \longrightarrow \mathbb{S} that\ provided\ a\ status\ for\ each\ place$;

$-\ every\ place\ s \in S\ with\ \sigma(s) \in \{e,i,x\}\ is\ such\ that\ l(s) = \{\bullet\}$.

Statuses are represented as labels, except for ε that is not represented. However, for entry and exit representation of statuses of place N, N^e *and* N^x *are respectively used for the* status representation.

Flow control compositions are defined in Petri nets using node statuses. Let $\mathbb{S}$ be the set of statuses that consists of:

$-$ e, the status for *entry places*, *i.e.*, those marked in an initial state of a Petri net.

$-$ x, the status for *exit places*, *i.e.*, those marked in a final state of a Petri net.

$-$ i, the status for *internal places*, *i.e.*, those marked in intermediary states of a Petri net.

$-$ $\mathcal{E}$, the status of *anonymous places*, *i.e.*, those with no distinguished status.

$-$ *arbitrary* names like *count* or *var*, for *named places*.

Anonymous and named *places* together are referred to as *data of buffer places*. The *entry, internal, and exit places* are together known as *control flow places*.

Let N_1 *and* N_2 be two Petri nets with control flow. Four control flow operations are considered in such instances.

$-$ *sequential* composition N_1 ; N_2 allows to execute N_1 followed by N_2;

$-$ *choice* composition $N_1 \square N_2$ permits the execution of either N_1 or N_2.

$-$ *interation* composistion $N_1 \circledast N_2$ supports the execution of N_1 repetitively (including zero time), and then N_2 once.

$-$ *parallel* composition $N_1 \| N_2$ permits the concurrent execution of both N_1 and N_2.

Definition 4 (Control flow composition) *Let* $\diamond \in \{;, \square, \circledast, \|\}$ *be a control flow operator and* $N_\diamond$ *the corresponding operator net. Let* N_1 *marked by* M_1 *and* N_2 *marked by* M_2 *be two Petri nets with control flow such that* N_1, N_2 *and* $N_\diamond$ *are pairwaise disjoint. Then,* $N_1 \diamond N_2$ *is defined as the Petri net with control flow resulting from the gluing*

phase followed by the merging phase.

The gluing phase assumes that every transition including an operator net signifies one of the operands of the composition. The correct control flow between transitions, the places in the operator net implement such actions. The flow control places between composing nets are organised using an appropriate syntax in the operator net to produce the corresponding control flow.

In the next subsection, the system modelling is presented. The system modelling consists of the system modelling for the Blockchain-based IoT data authentication system, and the modelling of the blockchain-based consensus.

5.3 System Modelling

5.3.1 System Modelling for the Blockchain-based IoT Data Authentication System

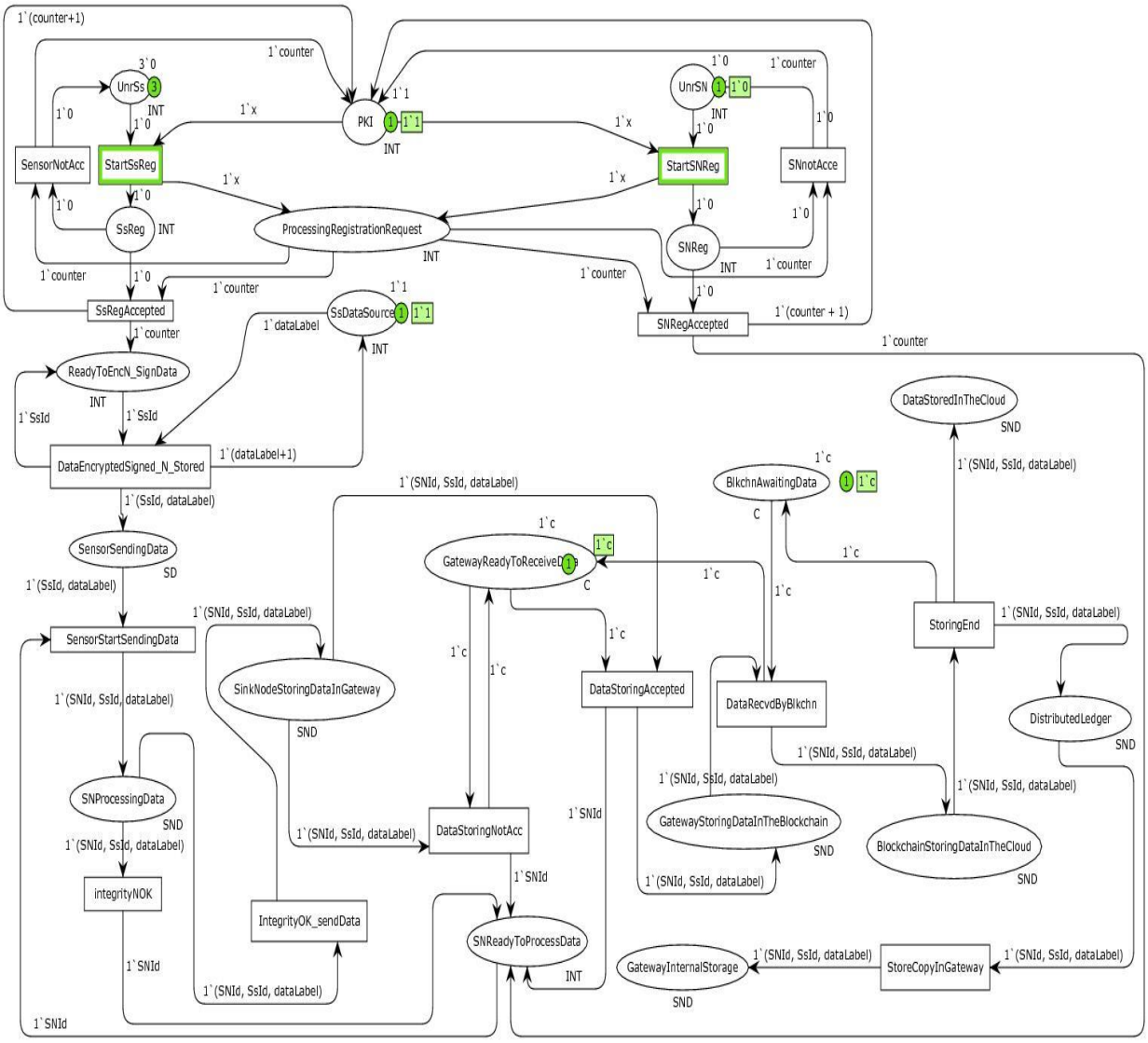

Figure 27: System model for blockchain-based IoT data authentication system with declarations of colour sets and variables.

Figure 27 shows the CPN model of the system with colour sets and types, places, transitions, and variables.

Table 9, 10 and 11 respectively describe the places, transitions, and variables of the model.

Table 9: CPN Places within the model

Place	Description
UnrSs	Unregistered Sensor end device
SsReg	Sensor end device registering.
PKI	Public Key Infrastructure component of the local IoT Network
UnrSN	Unregistered Sink node
SNReg	Sink node registering
ProcessingRegistrationRequest	Public Key Infrastructure component of the local IoT Network
ReadyToEncN_SignData	Registered sensor in a state of collecting data from the data source, encrypting it, and signing the data.
SsDataSource	Sensor data source. It symbolizes the immediate environment of a sensor with several physical measurements (such as temperature, humidity, pressure) to be collected by the sensor.
SensorSendingData	Sensor Ready to Send the ciphertext and the digital signature to a sink node.
SNProcessingData	Sink Node Processing Data by verifying the integrity of a sensor data.
SNReadyToProcessData	Sink Node Ready to process data.
SinkNodeStoringDataInGateway	Sink Node storing validated data into the IoT gateway.
GatewayReadyToReceiveData	IoT Gateway receiving validated data from the sink

Place	Description
	node.
GatewayStoringDataInTheBlockchain	IoT Gateway storing data using smart contract
BlkchnAwaitingData	Blockchain Awaiting Data from the gateway.
BlockchainStoringDataInTheCloud	Blockchain storing data in the cloud.
DataStoredInTheCloud	Data Stored In The Cloud represents remote storage for validated IoT data.
DistributedLedger	Distributed Ledger with updated storage of validated data.
GatewayInternalStorage	Gateway Internal Storage. It is an internal storage that forms part of the physical IoT network. It is used to symbolize persistent storage location of updated distributed ledger for the local IoT.

Table 10: CPN Transitions within the model

Transition	Description
StartSsReg	Start Sensor Registration. This event leads to a state of the sensor initializing registration.
SsRegAccepted	Sensor Registration Accepted
SensorNotAcc	Sensor Registration Not Accepted
DataEncryptedSigned_N_Stored	Ciphertext and digital signature stored
SensorStartSendingData	Sensor starts sending ciphertext and digital signature to sink node.

Transition	Description
StartSNReg	Sink Node Registration is initialized.
SNRegAccepted	Sink Node Registration is successful. This propels the sink node to a state where it can accept ciphertext and verify the digital signature from the sensor.
SNnotAcce	Sink Node Registration not accepted. It symbolizes a failed sink node registration. Such event means the sink node is not available to receive any data from a senor.
DataStoringAccepted	An event that depicts a situation where the integrity checks on a ciphertext after undertaking a signature verification on the digital signature has yielded same digital signature values to validate the genuineness of the data to be stored. The action of the integrity checking is a prior measure to either discard a ciphertext or store the ciphertext.
DataStoringNotAcc	Data Storing Not Accepted. It is an action that describes the situation of a failed integrity check on a ciphertext.
IntegrityOK_sendData	It is an event that resulted from a successful match in the values of the given and derived digital signature that accompanied the ciphertext from a sensor. The integrity check is conducted through a digital signature verification by digital signature algorithm. This allows the removal of one token from the Sink node and a subsequent addition to the IoT Gateway.
IntegrityNOK	Integrity Not OK represented a situation where the

Transition	Description
	digital signature values are not the same. Any difference in the digital signature values will result in discarding the ciphertext for the sink node ID to be released to accept and process the next data from the sensor.
DataRecvdByBlkchn	Data Received By Blockchain. It represented an action where a blockchain virtual network that runs a distributed ledger received data from a sink node and validate the data through consensus to append the data to update the digital ledger.
StoringEnd	Storing Ends. This event depicts the end of the storage of validated data onto the digital ledger. The sink node data is permanently stored on the digital ledger and distributed between the IoT gateway internal storage and the cloud.
StoreCopyInGateway	Store Copy In Gateway. An event that symbolizes the duplication of the updated distributed ledger of validated IoT data on the IoT gateway persistent storage.

Table 11: CPN Arc variables used for the model

Arc label	Description	Location of arc label within the Sub Model
x	It is an integer variable that	Local IoT network

	represents the Id assignment for registration request processing of sensors and sink nodes.	
counter	It counts and tracks the number of sensors or sink nodes that have been registered with the PKI. It is used to assign unique Ids to sensors and sink nodes.	Local IoT network
dataLabel	It symbolizes the sensor data that has been collected.	Local IoT network
SsId	The identity of the sensor	Local IoT network
SNId	The identifier for the sink node	Local IoT network

In the next subsection, the modelling of the blockchain consensus is presented.

5.3.2 System Modelling for the Blockchain-based Consensus.

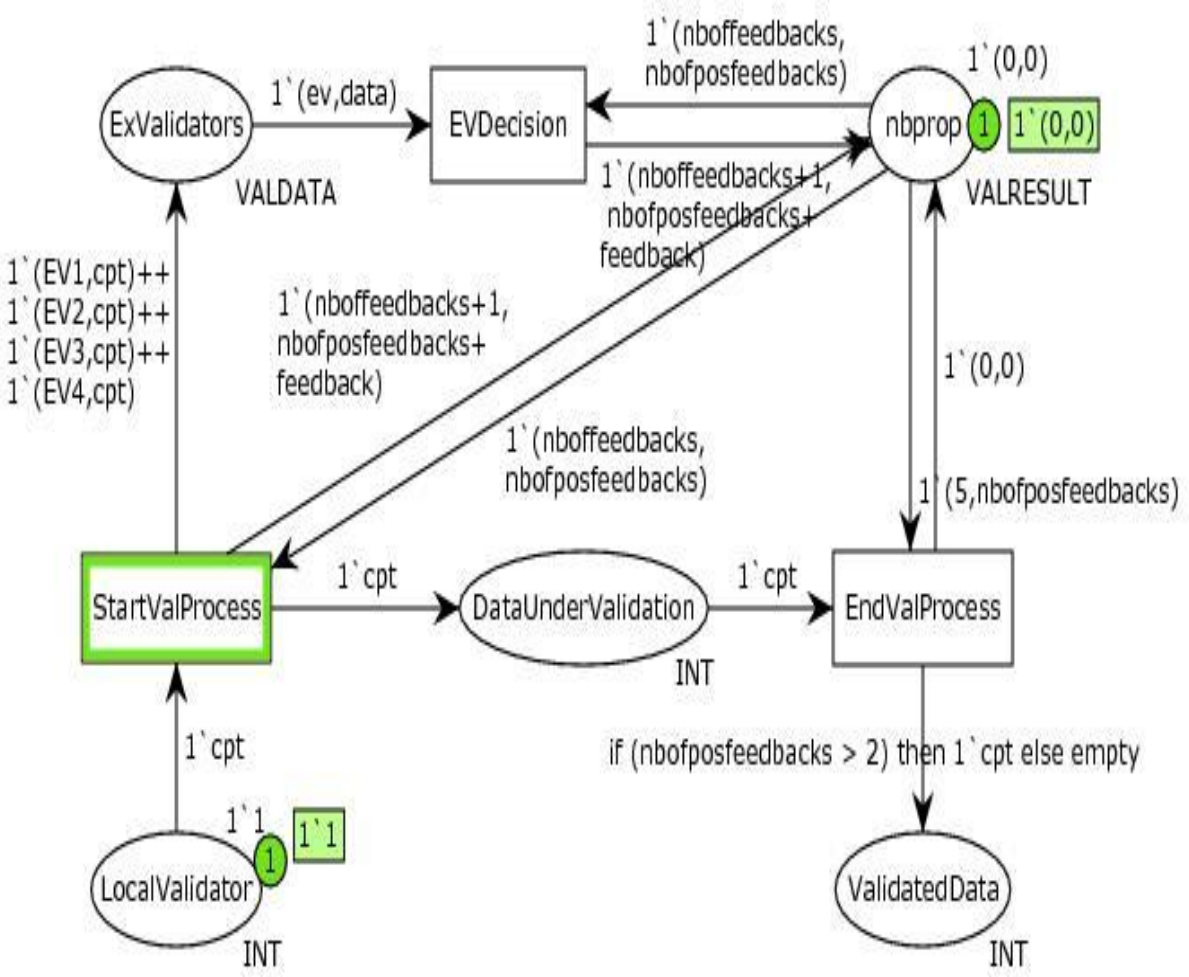

Figure 28: Blockchain-based consensus model

Figure 28 presents the CPN model with colour sets and types, places, transitions, and variables for the blockchain-based consensus.

Table 12 describes the places, transitions, and variables of the model.

Table 12: CPN Modelling features for the blockchain consensus

Abbreviation	Description	CPN Component
LocalValidator	It represents an internal sink node. The container for keeping a sink node data prior to a validation operation	Places
ExValidators	It denotes external sink nodes that form the external validators. These are all the other sink nodes within the hierarchical IoT network. They join the internal validator to reach an agreement on a message through a blockchain consensus.	
nbprop	A container for keeping all the decisions resulting from validators using the consensus rule to vote on a data under validation.	
ValidatedData	It represents the results after the voting decisions undertaken by all the validators have ended. When the number of positive feedbacks where at least 60% of the total decisions by the validators, the data will be moved to a new state of ValidatedData.	
DataUnderValidation	It denotes a place that specifies the	

Abbreviation	Description	CPN Component
	current data being validated is kept. It is represented by the identity of the data which is captured as (`cpt) on the arc inscription.	
StartValProcess	It is an event that signifies the start of the consensus session. The local validator is an input to this event. It fires the data from the local validator to the external validators as well as updates the DataUnderValidation and the "nbprop" places.	Transitions
EVDecision	It is a transition label for the CPN event that fires the decision of each external validator as feedback on a data. The input of the transition is the external validator and the data to be validated based on the consensus rules. The output for this transition is the number of decision feedbacks and the number of positive decision feedbacks.	
EndValProcess	It is a transition to signify the close of a consensus session. It has DataUnderValidation, and nbprop as input. The output of this transition is	

Abbreviation	Description	CPN Component
	the ValidatedData.	
cpt	A token for describing the identity of data under validation. The data under validation is submitted by the local validator to the validators where the consensus mechanism is applied on the data using other established rules in the consensus to vote on the data in validating it.	Variables
EVi (i = 1 .. 4)	The token identifying the external validator i.	
data	A data element representing the data under validation by the external validator.	
nboffeedbacks	It is a counter that records the decisions of voting activities by providing an update on the total number of feedback decisions	
nbofposfeedbacks	It is a counter that records the total number of positive feedback decisions. Both the number of feedbacks and the number of positive feedback decisions are stored as a token in the “nbprop” place and are updated each time an	

Abbreviation	Description	CPN Component
	external validator decision is taken.	

In the next subsection, the model simulation results for the entire authentication system, and the model simulation of the blockchain-based consensus are presented.

5.4 System Simulation

This subsection describes the simulation results of the CPN model of the IoT data authentication system as well as the blockchain-based consensus mechanism including the declarations of colour sets and variables.

5.4.1 The Simulation of the Blockchain-based IoT Data Authentication System

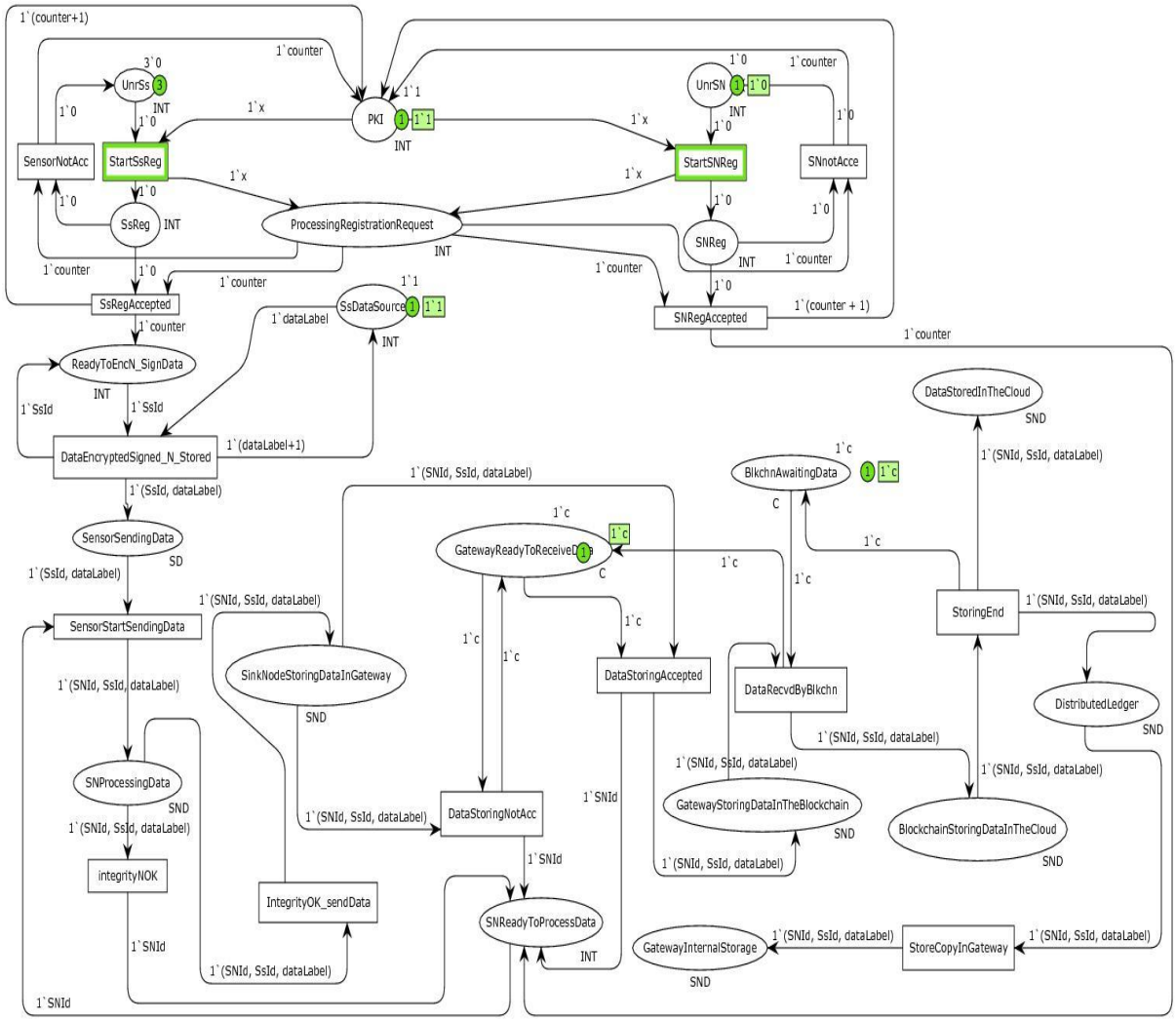

Figure 29: Network Diagram for the IoT Data Authentication System with marked Tokens

Figure 29 represents a fully marked CPN design of the network diagram with token elements. This figure shows the initial state of the system. In the initial state, 3 sensors are available and not yet registered (see place UnSs). Indeed, an unregistered sensor has the dummy Id “0”. There is a Public Key Infrastructure (PKI) place that has been set up with an initial colour marking of 1`1. The value of the token in this place is the Id that will be assigned to the next component (sensor or sink node) that will be registered. The initial value of 1 means that the first component registered will have the Id “1”. The unregistered sink node place (UnrSN) has an initial colour marking of 1`0. Indeed, an unregistered sink node has a dummy Id of

"0". There is a data source place (SsDataSource) with initial colour marking of 1`1. This means that the first sensor data collected will have the label "1". The GatewayReadyToReceiveData and the BlkchnAwaitingData both have initial coloured token of 1`c. This means that in the initial state, the gateway is ready to receive data from the sink node and the blockchain is awaiting data from the gateway.

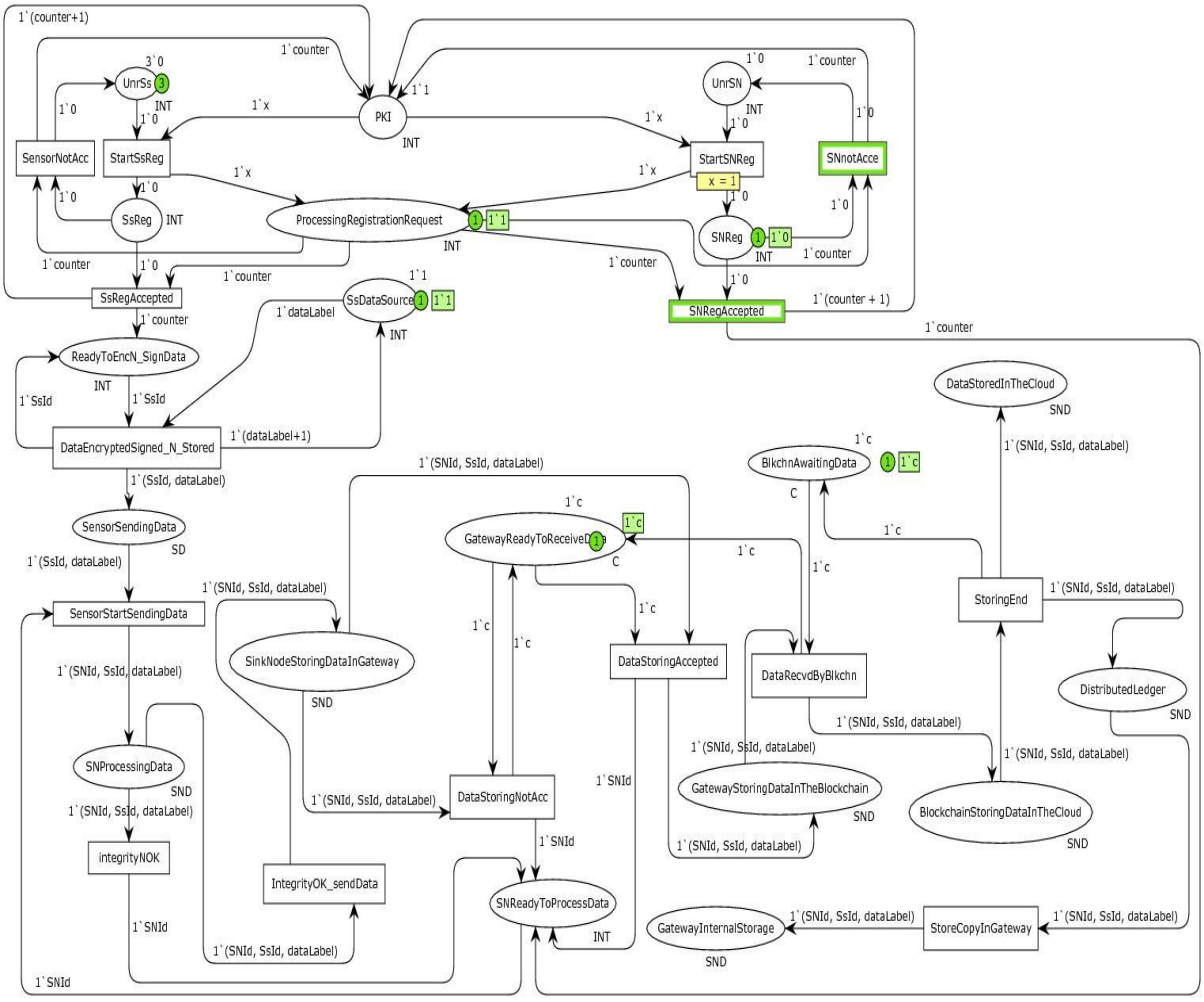

Figure 30: Sink node registering

Figure 30 shows the data flow that demonstrates the sink node registration event. In the registration phase of the sink node, the PKI assigns appropriate digital certificate tokens and other credentials to enable it accept sensors to be connected to it to communicate secure data between them. During the sink node registration, the PKI checks for appropriate credentials

specific to a local IoT before approving the sink node with the requisite digital certificate tokens. A sink node during the registering phase is either accepted after it has satisfied the enrolment checking by the PKI. The sink node can also be refused or not accepted for registration.

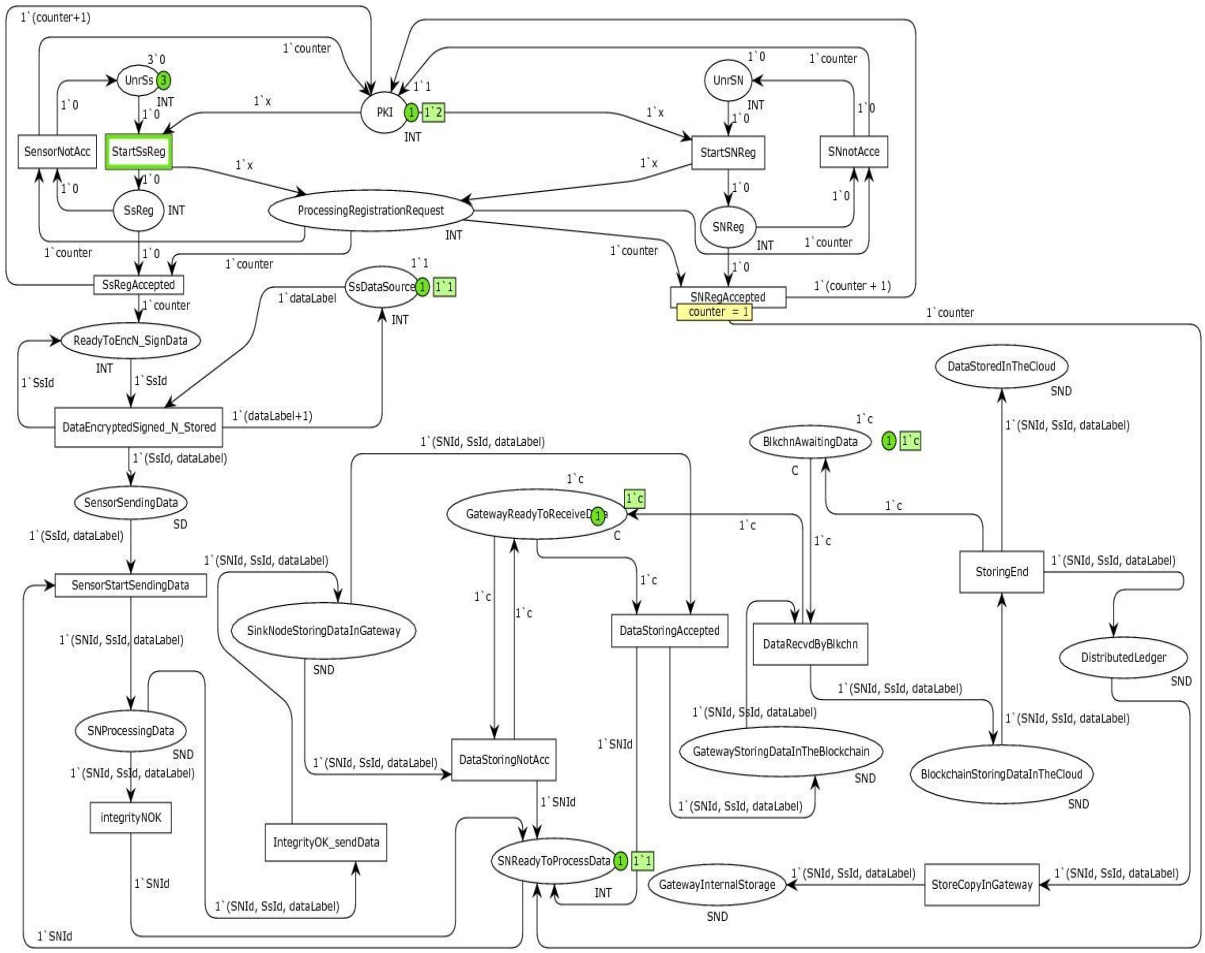

Figure 31: Sink node registration accepted

Figure 31 shows that a sink node registration has been successfully accepted. A successfully registered sink node now moves to the next state where it is active and can accept and process data from sensors. The sink node's successful registration allows it to decrypt and verify the digital signature that accompanied ciphertext from sensors.

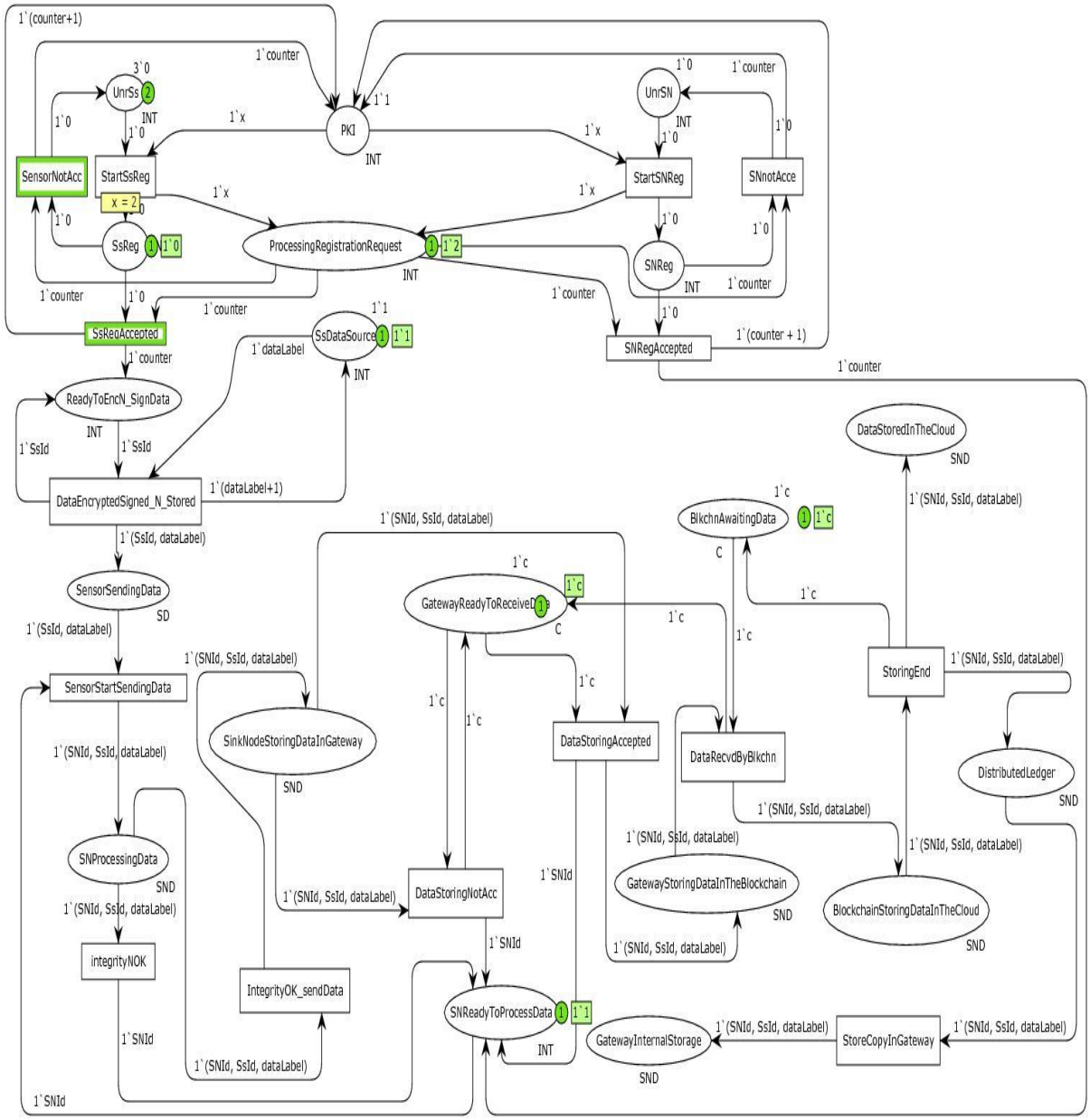

Figure 32: Sensor registering

Figure 32 represents the data flow diagram that illustrates the registration phase of a sensor in the local internet of things (IoT). The PKI ensures that sensors are enrolled and authenticated with a sink node to enable the sensor and the sink node to run algorithms that encrypt and sign data on the sensor, and decrypt and verify the signature of data from the sensor for subsequent storing and sharing of that data from the sensor. During the registration of the sensor, there are two outcomes for the registration event. The assumption is that a sensor is expected to have certain unique credentials to enable it pass for registration enrolment

through digital certificate verification with the PKI. The sensor is deemed successfully registered after satisfying all the registration requirements or it is considered not accepted when the sensor fails to satisfy the condition necessary for a successful registration. The transitions for *SensorNotAcc* and *SsRegAccepted* are both enabled to fire one of the token elements from the *UnrSs place*. The transition for the commencement of the sensor registration is the *StartSsReg* (Start sensor registering). An unregistered sensor moves from its initial state to a new state of *ProcessingRegistrationRequest.*

The initial colour token marking of 3 for the *UnrSs place* symbolizes a multiset element for the sensor *place*. The commencement of the sensor registration fires 1 of the colour elements in the token marking leaving the colour elements at 2.

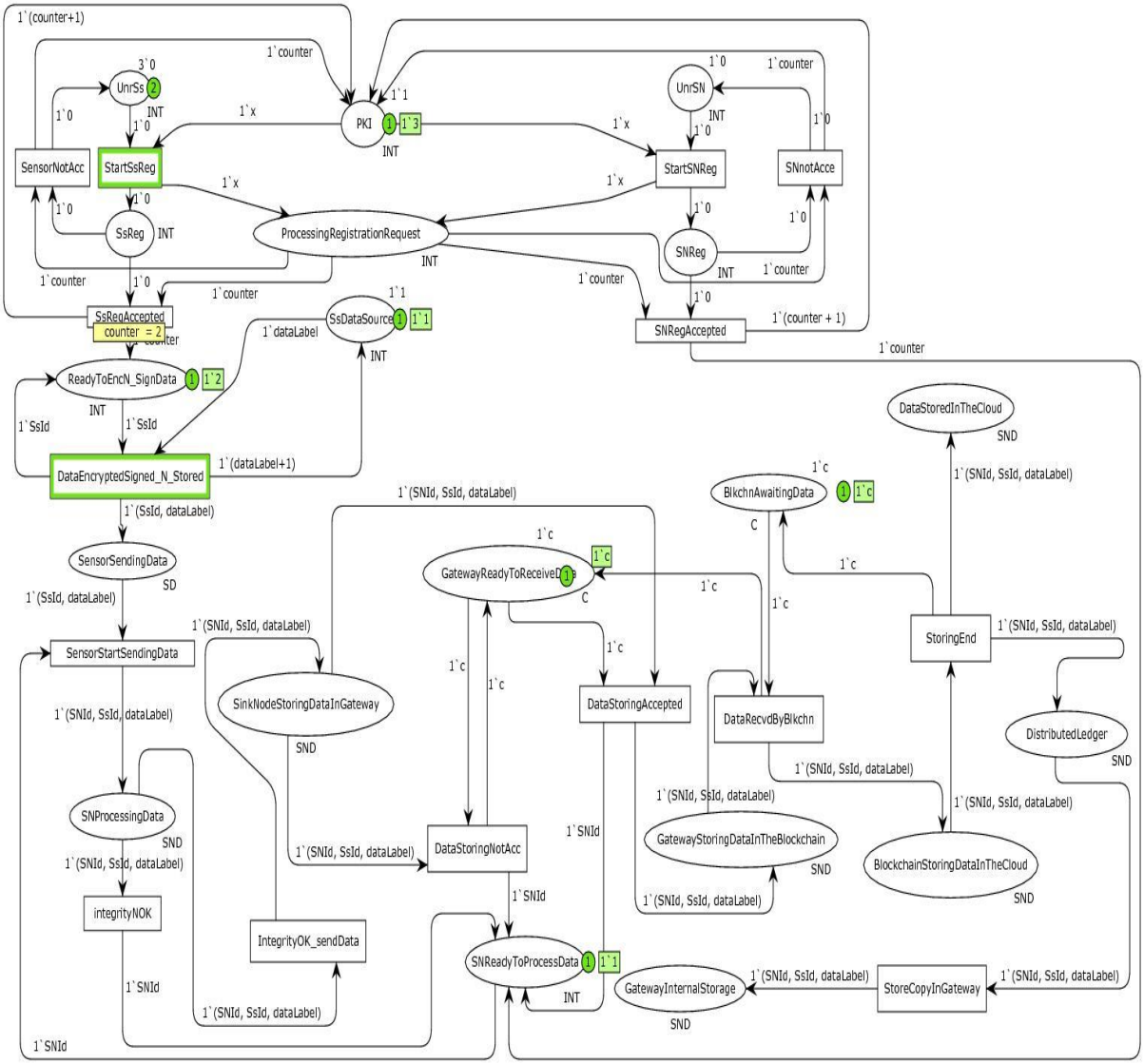

Figure 33: Sensor registration accepted - sensor is ready to encrypt, sign and send data

Figure 33 illustrates the state of the sensor after the acceptance of the registration request. This state enables the sensor to initiate the collection of data, encryption of the data, signing of the data, and sending the encrypted data and the digital signature.

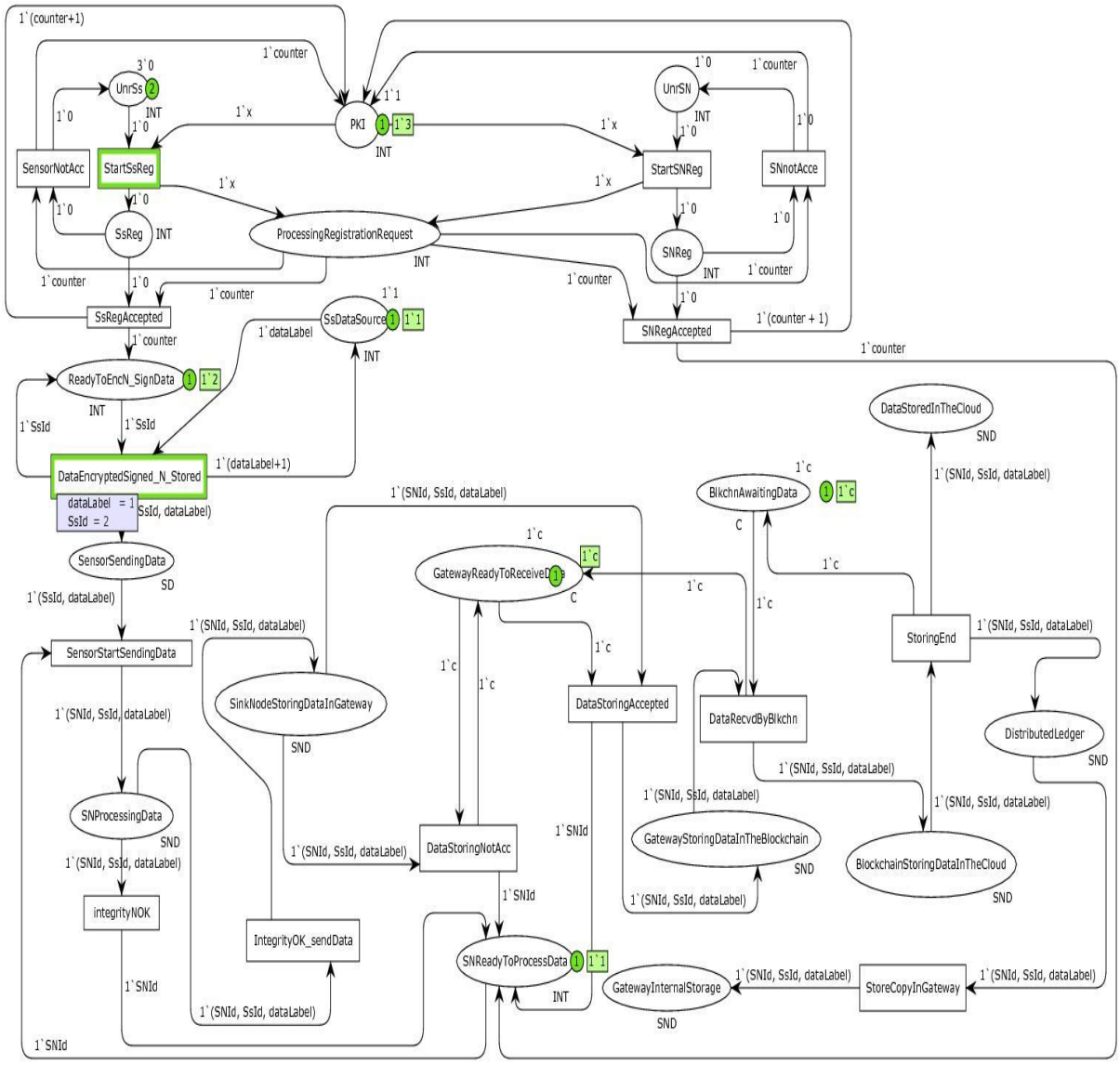

Figure 34: Sensor encrypting and signing the data

Figure 34 denotes a data flow illustration where the sensor encrypts, and signs data.

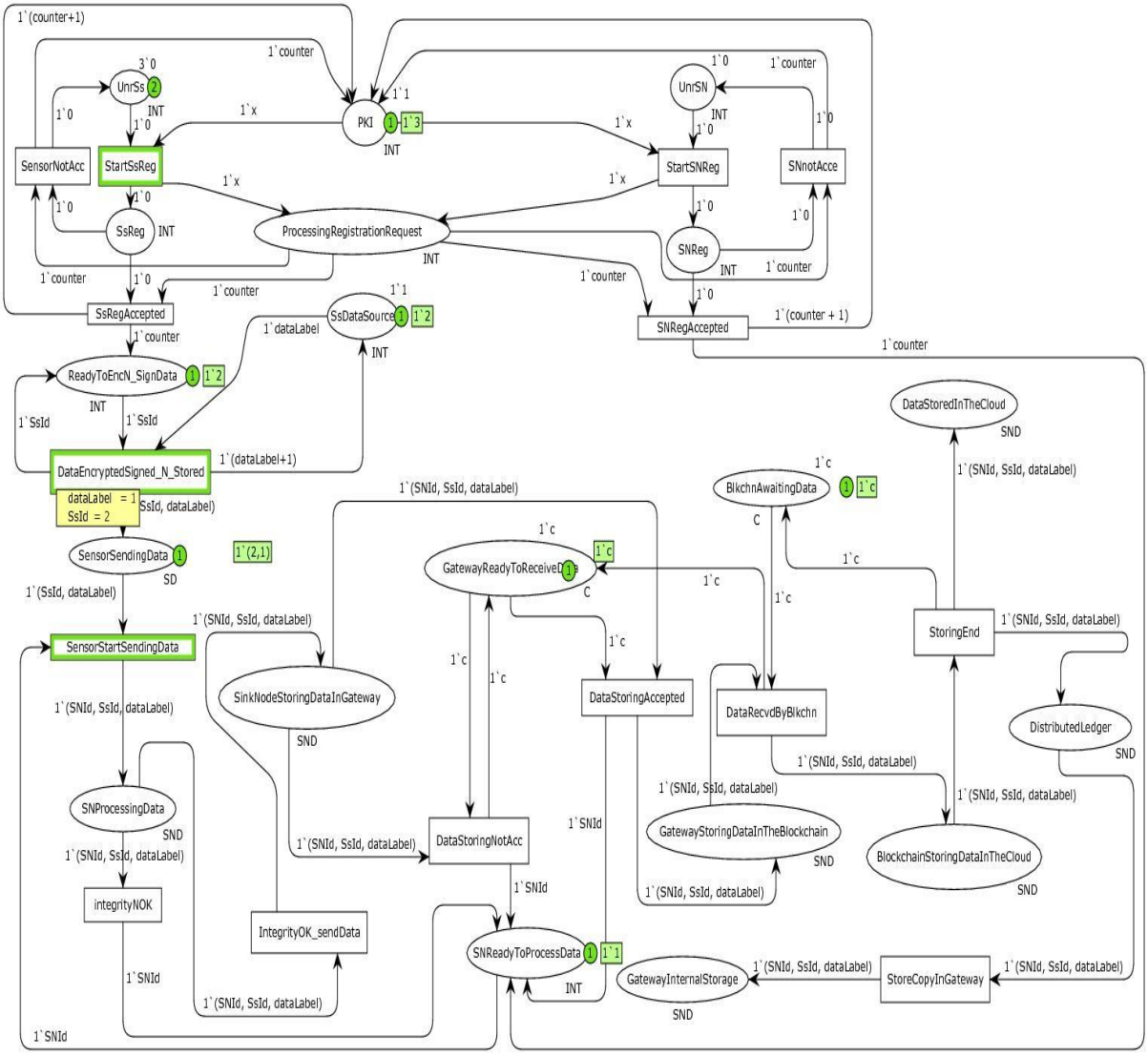

Figure 35: Sensor ready to send encrypted and signed data

Figure 35 represents the state where the sensor is ready to send encrypted and signed data to the sink node. The data is the concatenation of ciphertext and the digital signature for data values collected from the data source. The concatenation string is represented as dataLabel. The data from the sensor is sent to the sink node after this state.

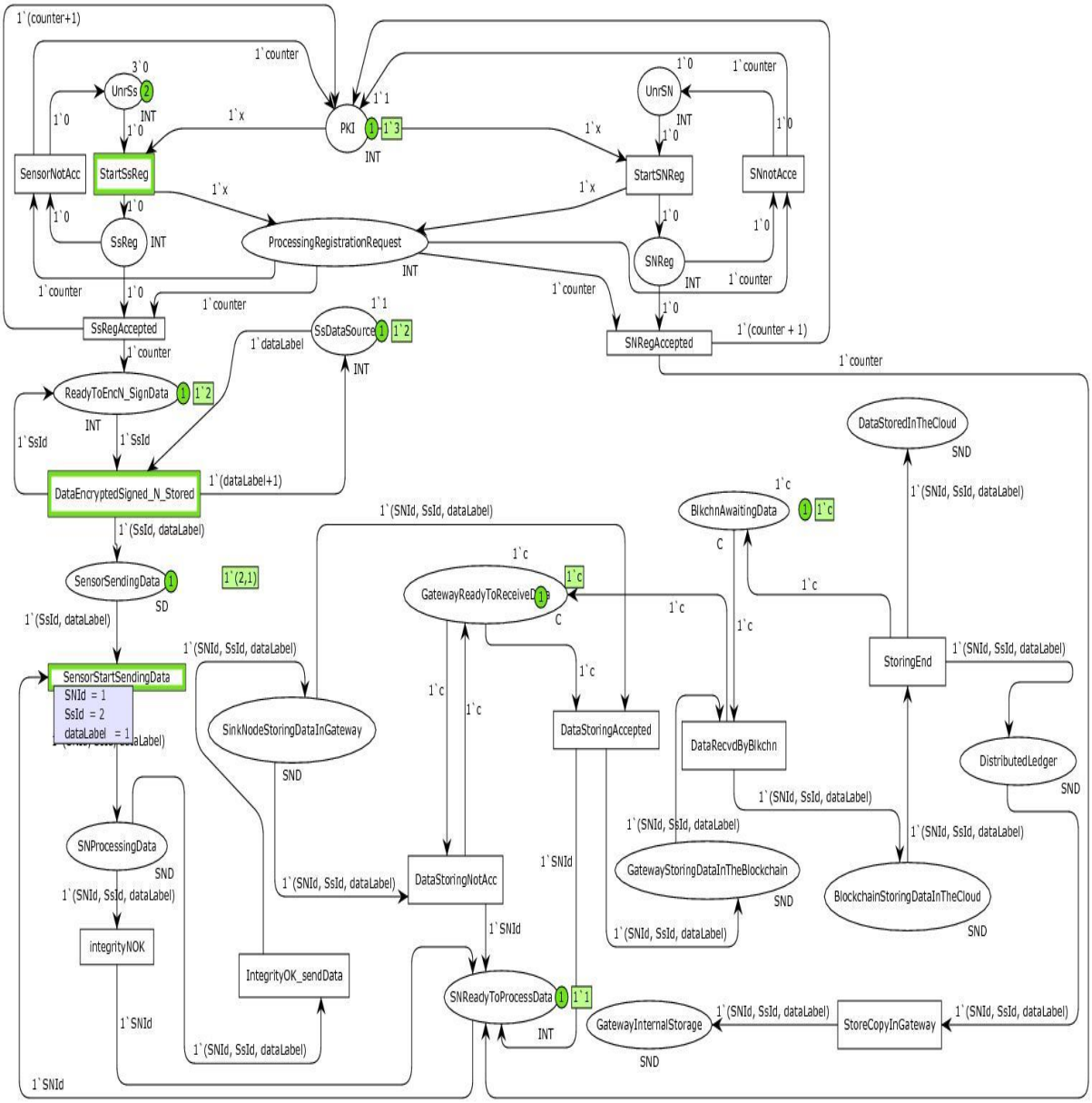

Figure 36: Sensor sending data to sink node

Figure 36 provides an illustration demonstrating the firing of the sensor-start-sending-data transition. The data flow shows that the sink node receives the data from the sensor. At this state, the sink node verifies to authenticate the integrity of the data from the sensor. The authentication of the data is done by using the digital signature algorithm that the sensor used to digitally sign the plaintext before sending it to the sink node. The sink node uses the digital signature algorithm to verify the digital signature (given digital signature) of the accompanying ciphertext. The verification of the digital signature involves generating a new

digital signature (derived digital signature) based on the received ciphertext. The two digital signatures (both the given and derived) are compared for character matching of the digital signature values. Same character sets symbolize that the integrity of the data is intact that is the IntegrityOKsendData transition is accordingly fired, and the data is forwarded to the next state where it is to be stored on the IoT gateway. Any difference in the character set of the derived digital signature in respect to the given digital signature will result in the firing of the IntegrityNOK and the sink node will discard the ciphertext. After this phase, the next sensor is commenced with registration to collect and send the data to the sink node. A sink node is actively involved in validating its content to be appended onto a blockchain.

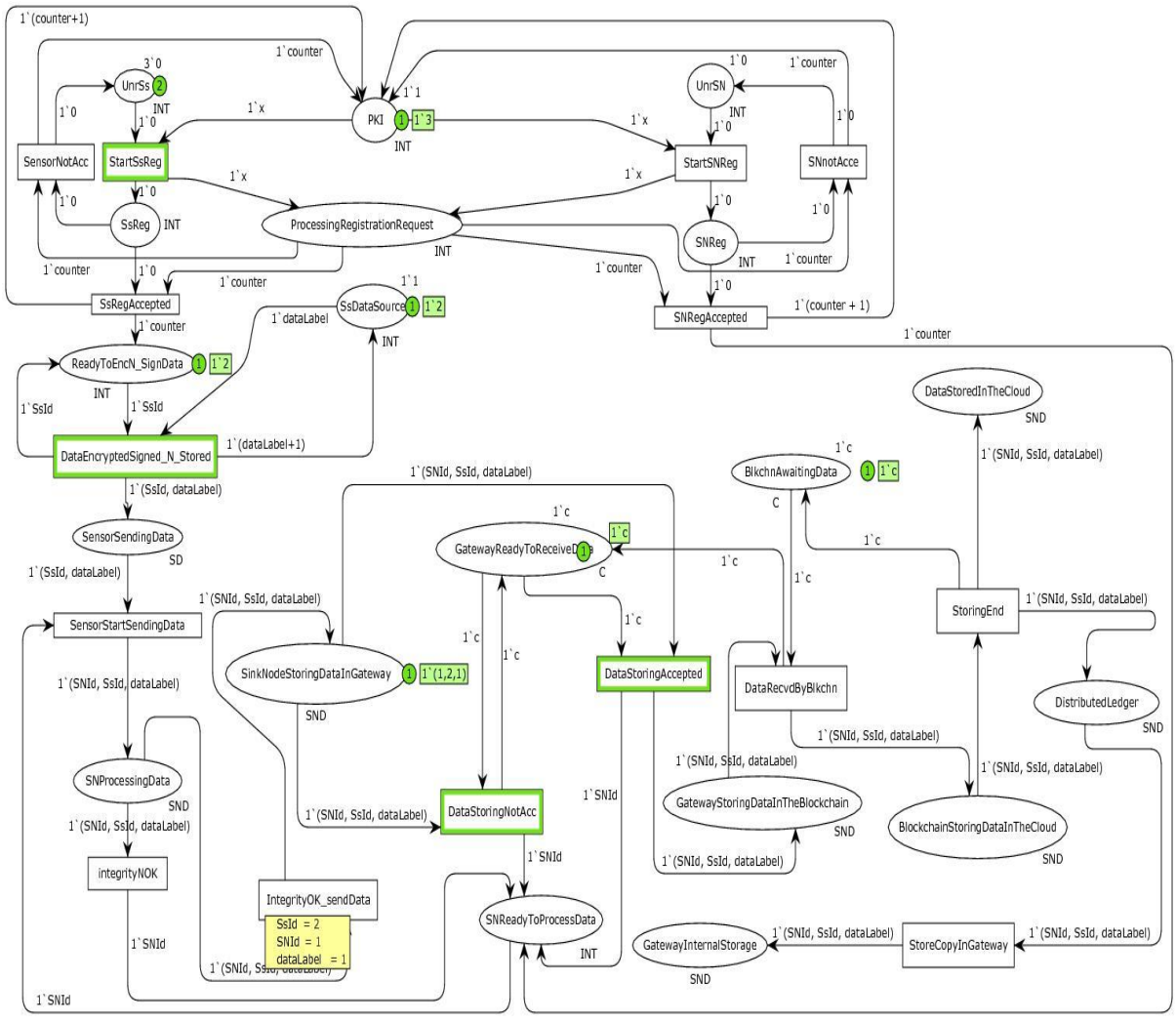

Figure 37: Sink node verifying and storing the sensor data in the gateway

Figure 37 shows the simulation result for firing the IntegrityOKsendData transition that moved the data on the sink node to a new place. The sink node data is a concatenation string that is composed of the sink node identity (SNId), sensor identity of the sensor that collected the data (SsId), and the ciphertext which is represented as dataLabel.

The sink nodes form consensus nodes to that blockchain consensus algorithms to validate data from sink nodes before appending the data from the sink nodes onto the distributed ledger. Any data appending request will be validated by the consensus nodes through a consensus algorithm. A validated data from a sink node will be appended on a distributed ledger with the updated digital ledger shared between the cloud and the IoT gateway persistent storage. A validly approved data from a sink node will result in the firing of the DataStoringAccepted transition to append the data to the blockchain. Any sink node that fails to validate the data that it is carrying will necessitate the firing of the DataStoringNotAcc transition where the sink node will be available to process the next data from the same sensor.

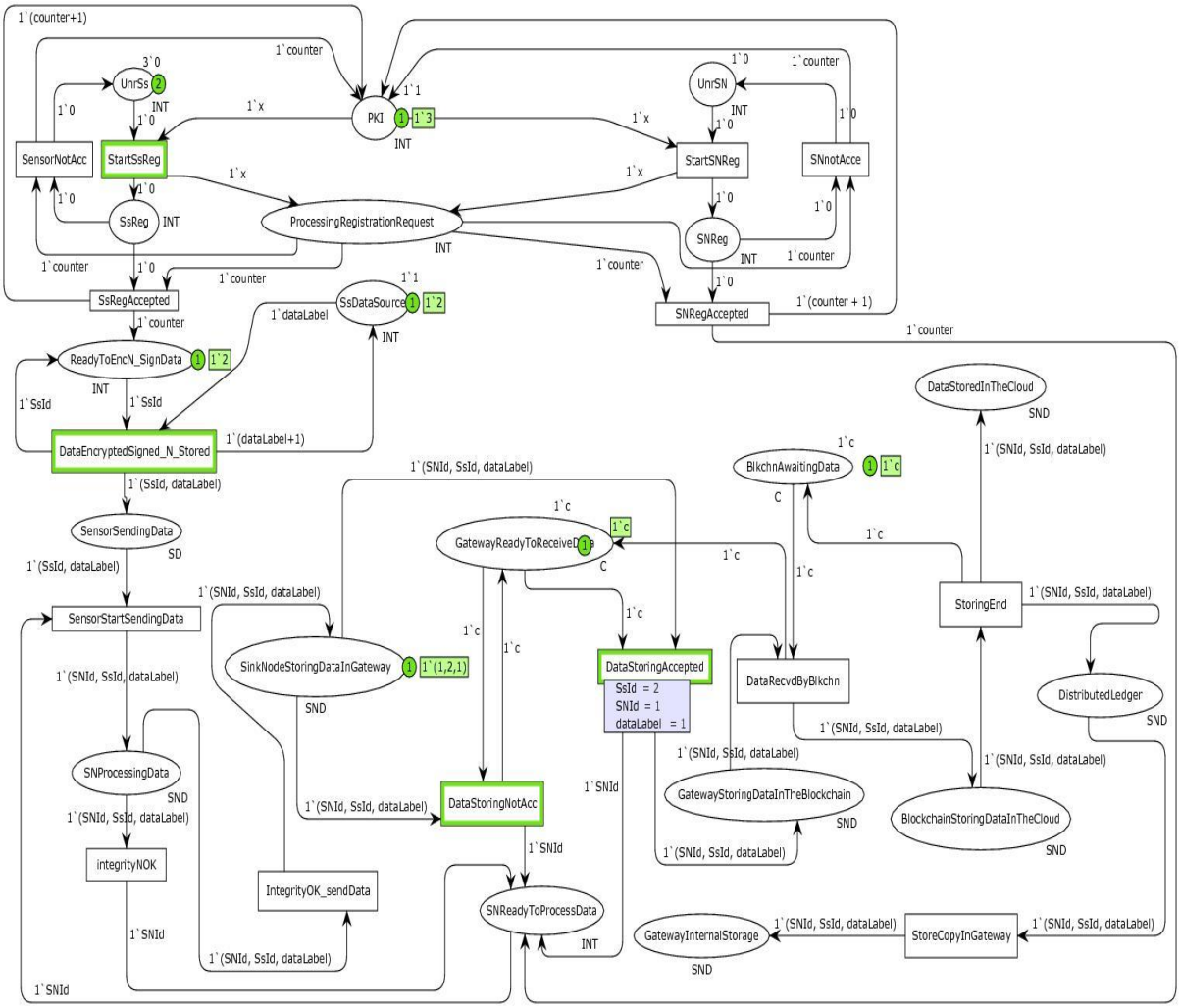

Figure 38: Gateway accepting data from the sink node

Figure 38 highlights the dataflow diagram illustrating the new state for an accepted storage on a distributed ledger from sink node data. The distributed ledger shares its updated digital ledger between the cloud and the IoT gateway persistent storage.

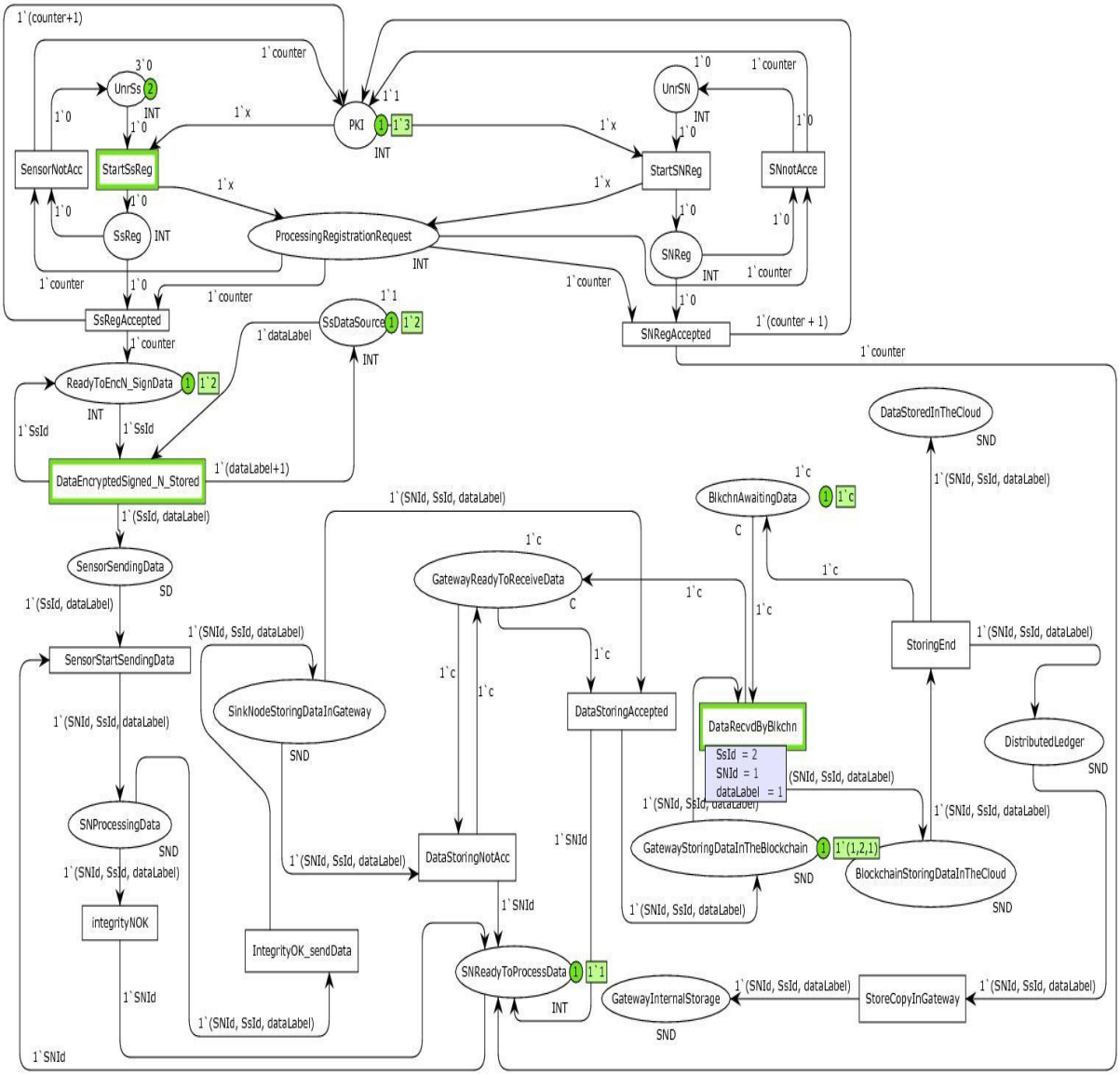

Figure 39: Blockchain receiving data from gateway

Figure 39 represents the new state of the sink node data after the DataReceivedByBlockchain transition has been fired. The firing of the transition causes the data to be finally stored onto the distributed ledger.

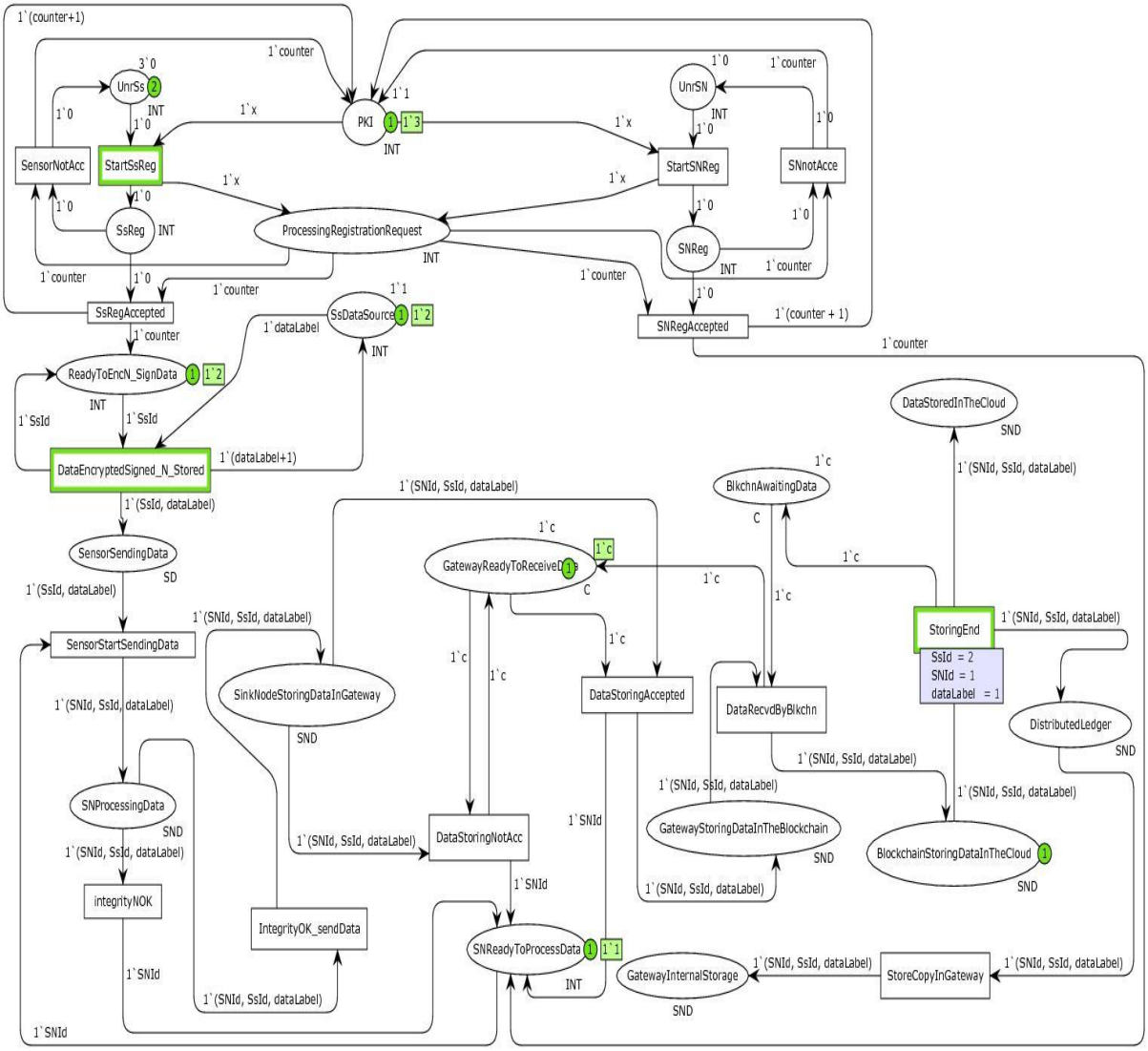

Figure 40: Appending data to the distributed ledger and storing the data in the cloud

Figure 40 represents the data flow illustration of the sink node data after it has successfully been appended to the distributed ledger. The distributed ledger is shared between the IoT gateway persistent storage and the cloud storage. The storage of a sink node data onto the distributed ledger with copies in the cloud completes the transmission of data from the sensor through the sink node, IoT gateway to its destination being the cloud. The Cloud storage triggers the firing of the DistributeLedger transition to share a copy of the updated state of digital ledger with the IoT gateway persistent storage.

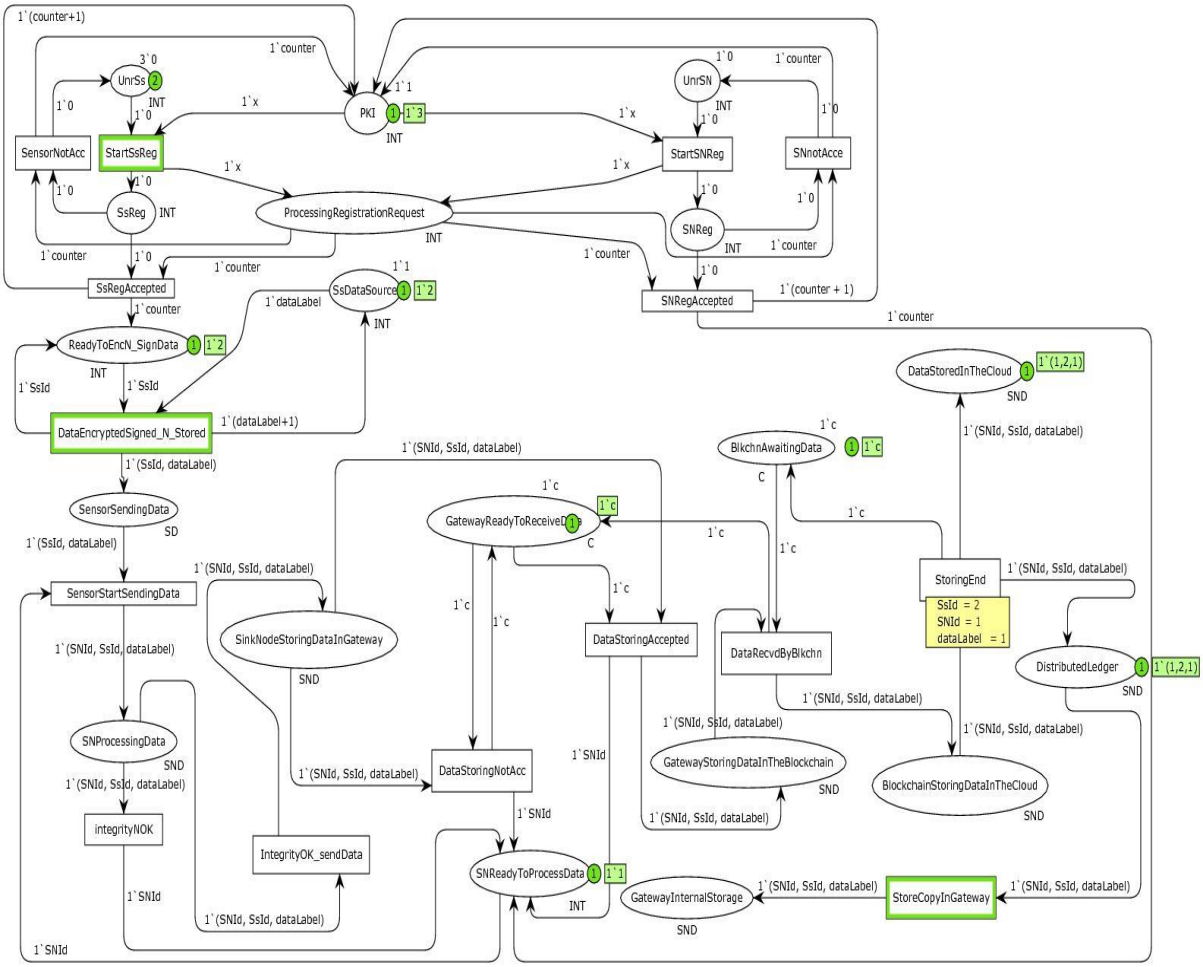

Figure 41: sensor data storage in the cloud

Figure 41 illustrates the data flow for the storage of the sensor data in the cloud. The firing of the StoringEnd transition removed the dataLabel, the SNId, and SsId details to the cloud for remote storage.

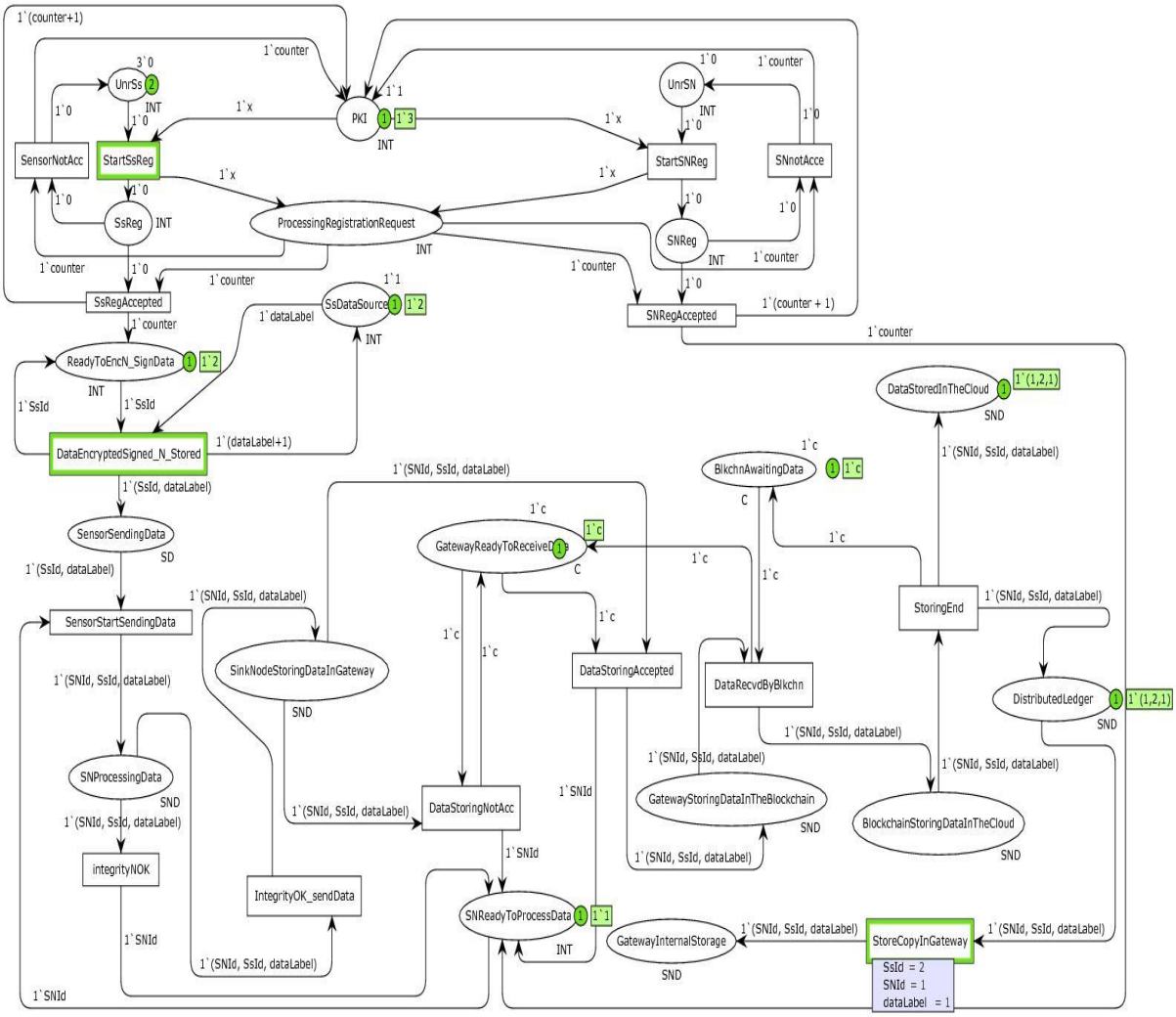

Figure 42: Duplicating the digital ledger in the gateway internal storage

Figure 42 illustrates the duplication of the digital ledger in the gateway internal storage.

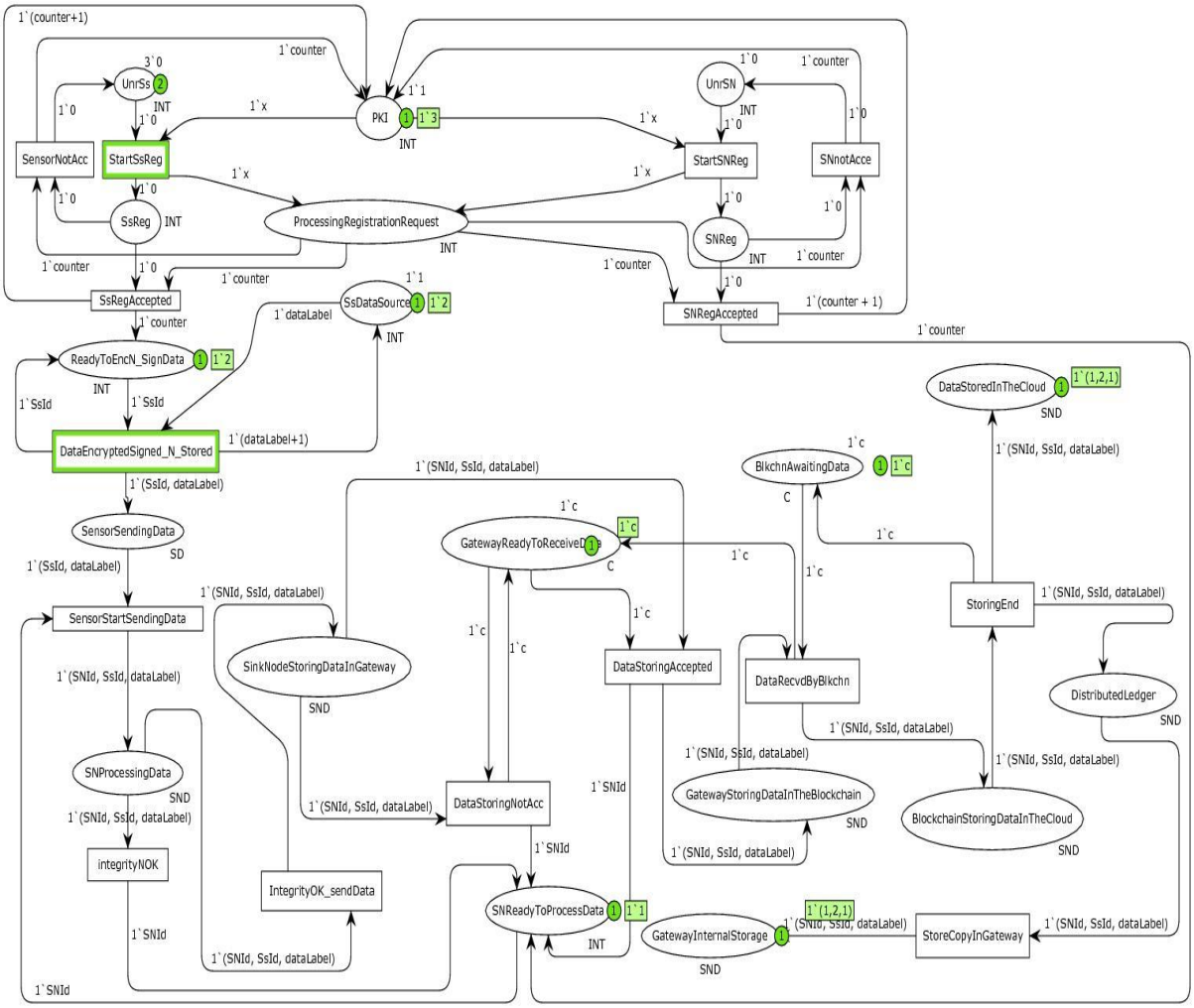

Figure 43: Distributed digital ledger between the cloud and the gateway internal storage

Figure 43 presents a data flow which illustrates the distribution of the updated state of the digital ledger on the IoT gateway persistent storage (IoT gateway internal storage) and the cloud storage. The structure and order of data storage is maintained for both the gateway and the cloud.

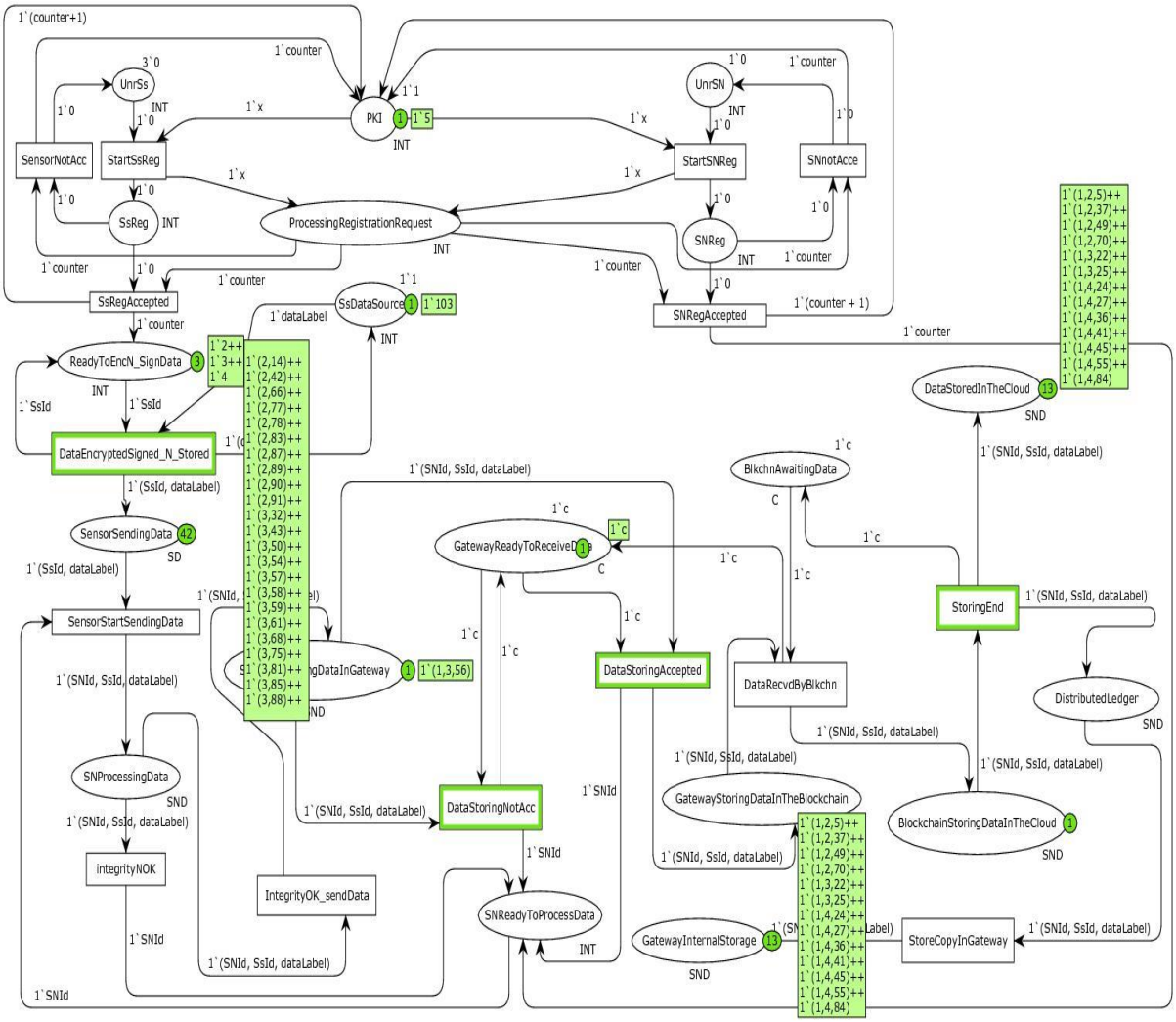

Figure 44: Simulation of 300 transition steps.

Figure 44 demonstrates the simulation results for 300 steps with several sensor data values collected by the sensor, transmitted the data to the sink node, and stored the sink node data on the peer-to-peer distributed ledger shared by the cloud and the internal memory of the IoT gateway (IoT gateway persistent storage). The simulation was not controlled, hence at every simulation step, a validated transition was randomly chosen by the system and fired.

In the next subsection, the second part of the simulation results for the blockchain consensus is presented.

5.4.2 The Simulation of the Blockchain Consensus

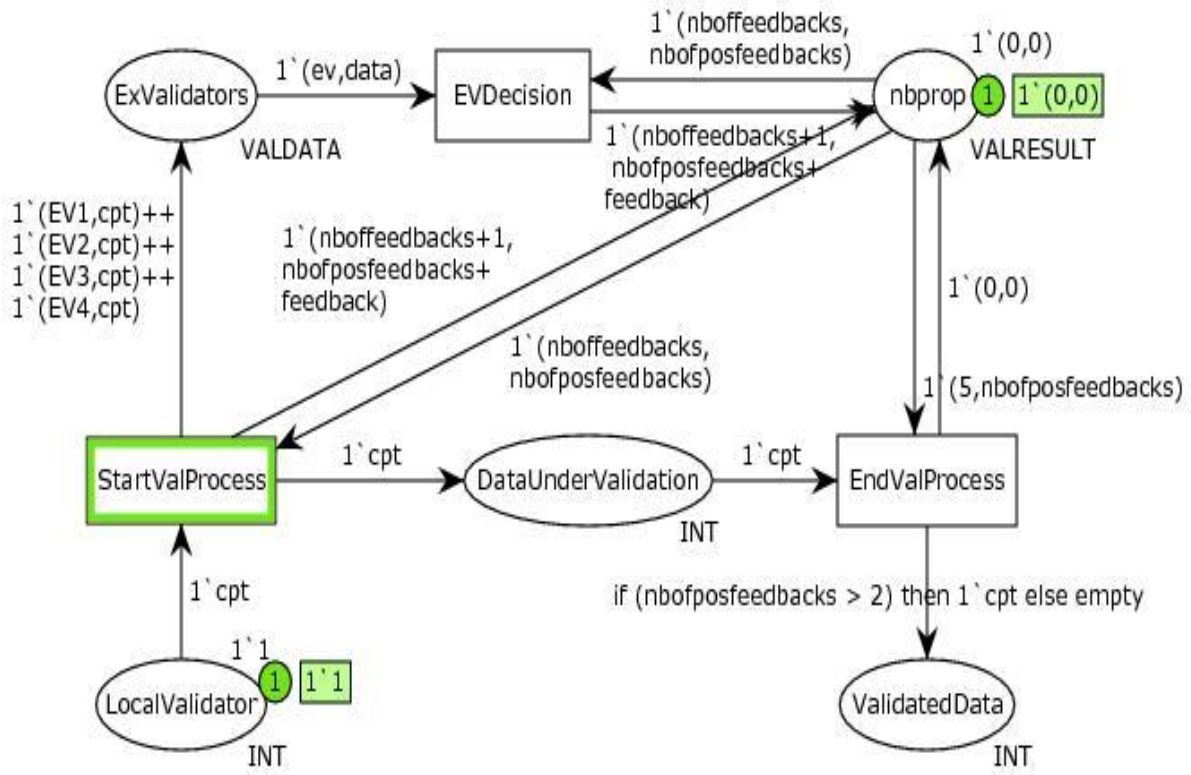

Figure 45: The Blockchain consensus Algorithm for the system

Figure 45 displays the CPN modelling of the blockchain consensus mechanism with validators. There are sink nodes represented as validators and a data to be validated through a blockchain consensus. Validated data is appended onto a distributed ledger. The validators are an amalgamation of the current sink node which presents the data to be validated as well as other external sink nodes within the hierarchical IoT network. The external sink nodes form the external validators. The sink node whose data is to be validated through the consensus assumes the local validator status. The validators (local and external) reach agreement to validate data if and only if the number of the positive feedbacks are more than half of all the total decisions from the validation voting by all the validators. Once a message does not get at least more than half of the total decisions to be positive feedbacks, that message is discarded. A session for the consensus by the validators is considered closed once the decision on a message has been made in accordance with the consensus correctness

criterion of the adopted blockchain consensus. The correctness criterion of the consensus is critical to make the algorithm byzantine fault tolerant.

The decisions of approving a data to be added to a distributed ledger is achieved through voting. A local validator broadcasts a message through the consensus network to all external validators. Each validator is given a stipulated time to decide by voting and sharing its decision from the votes as proposals on the validity of the message using an agreed procedure. These validators use zero-knowledge probabilistic theorem to validate each message from a local validator for a decision to either add that message to the distributed ledger or discard that message.

5.4.3 Interpretation of the Simulation of Blockchain Consensus

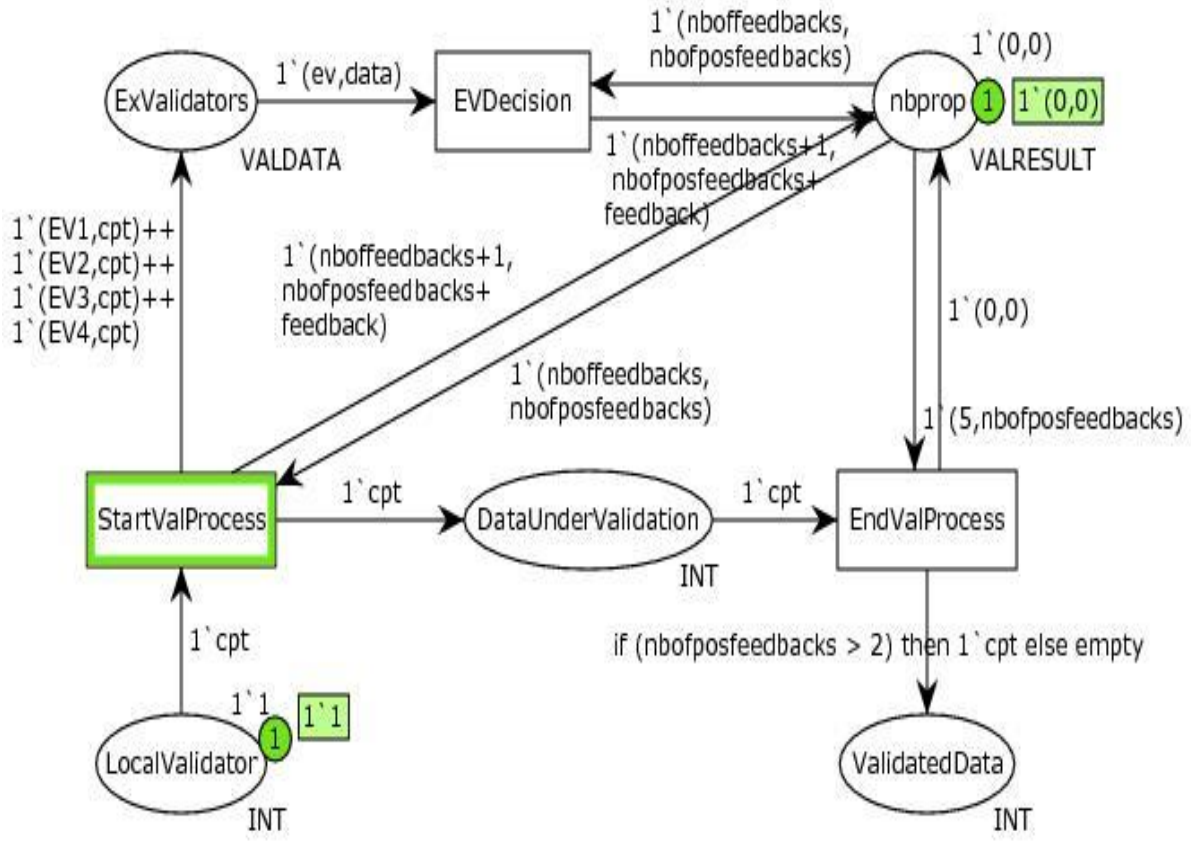

Figure 46: Blockchain consensus diagram

Figure 46 represents the initial state of a CPN modelling of a blockchain consensus process. The CPN diagram shows all the transitions, places, arcs, arc inscriptions, tokens, and the arc inscription with guard condition. The highlighted StartValProcess shows that the voting on the data under validation has not commenced. The token with the value 1`(0,0) on the "nbprop" place that store the number of proposals also validate that none of the validators has voted yet. The guard condition on the number of positive feedback decisions for validating data must be at least 3 positive feedback decisions out of the total number of 5 feedback decisions.

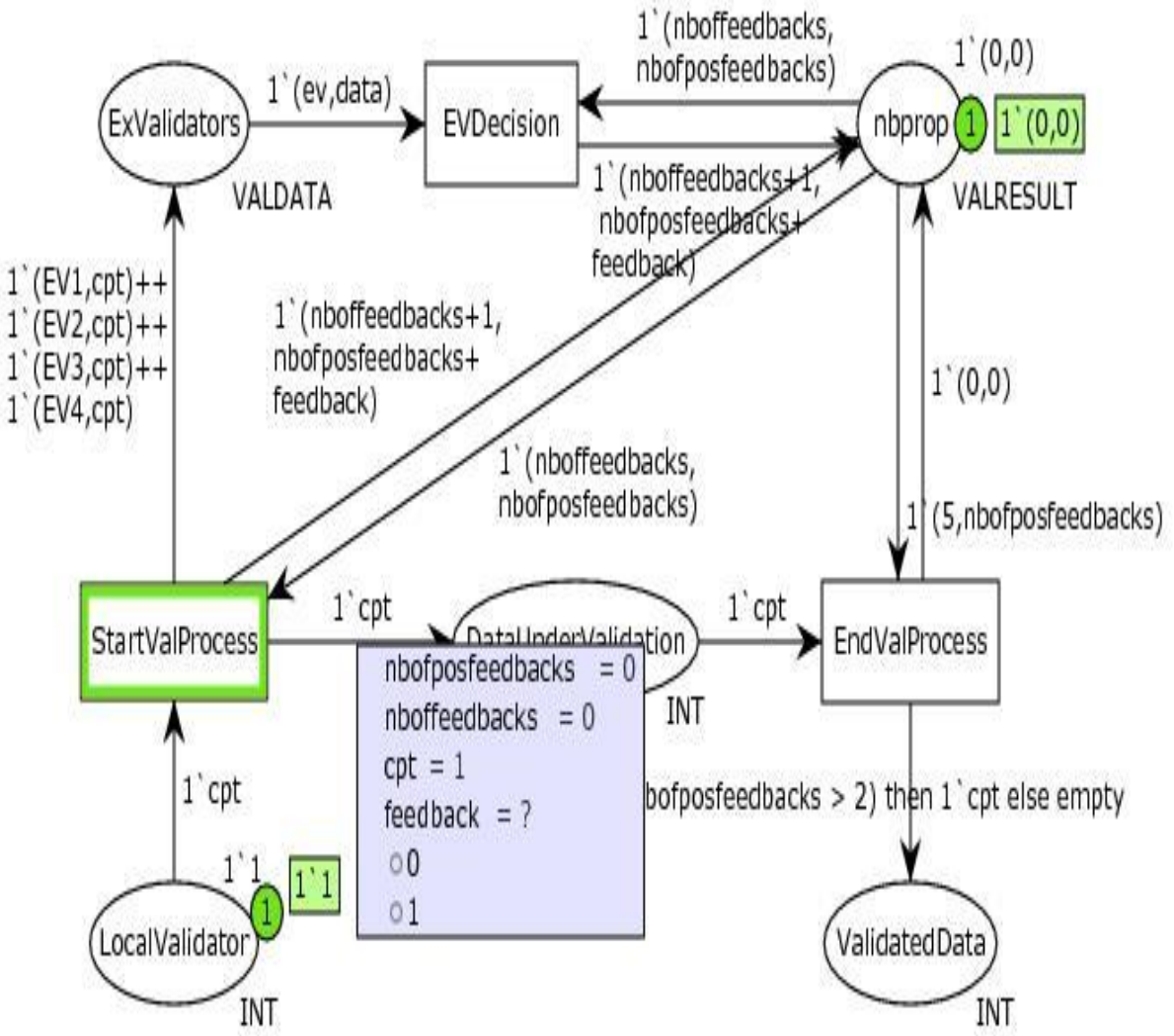

Figure 47: Local validator feedback decision

Figure 47 illustrates the start of the blockchain consensus where the local validator is yet to decide by voting by on the feedback of the data under validation. There are two feedback options (0,1) to be selected by a validator. Option 1 symbolizes positive feedback whereas option -0 denotes non-positive feedback. The start validation transition has not been fired yet. Selecting a choice for the feedback will fire the transition. The token (1`1) on the local validator symbolizes a single node data and the specific data to be validated is 1. The update on the number of proposals “nbprop” of 1`(0,0) shows that voting on the decision feedback on the data under validation has not started (0,0).

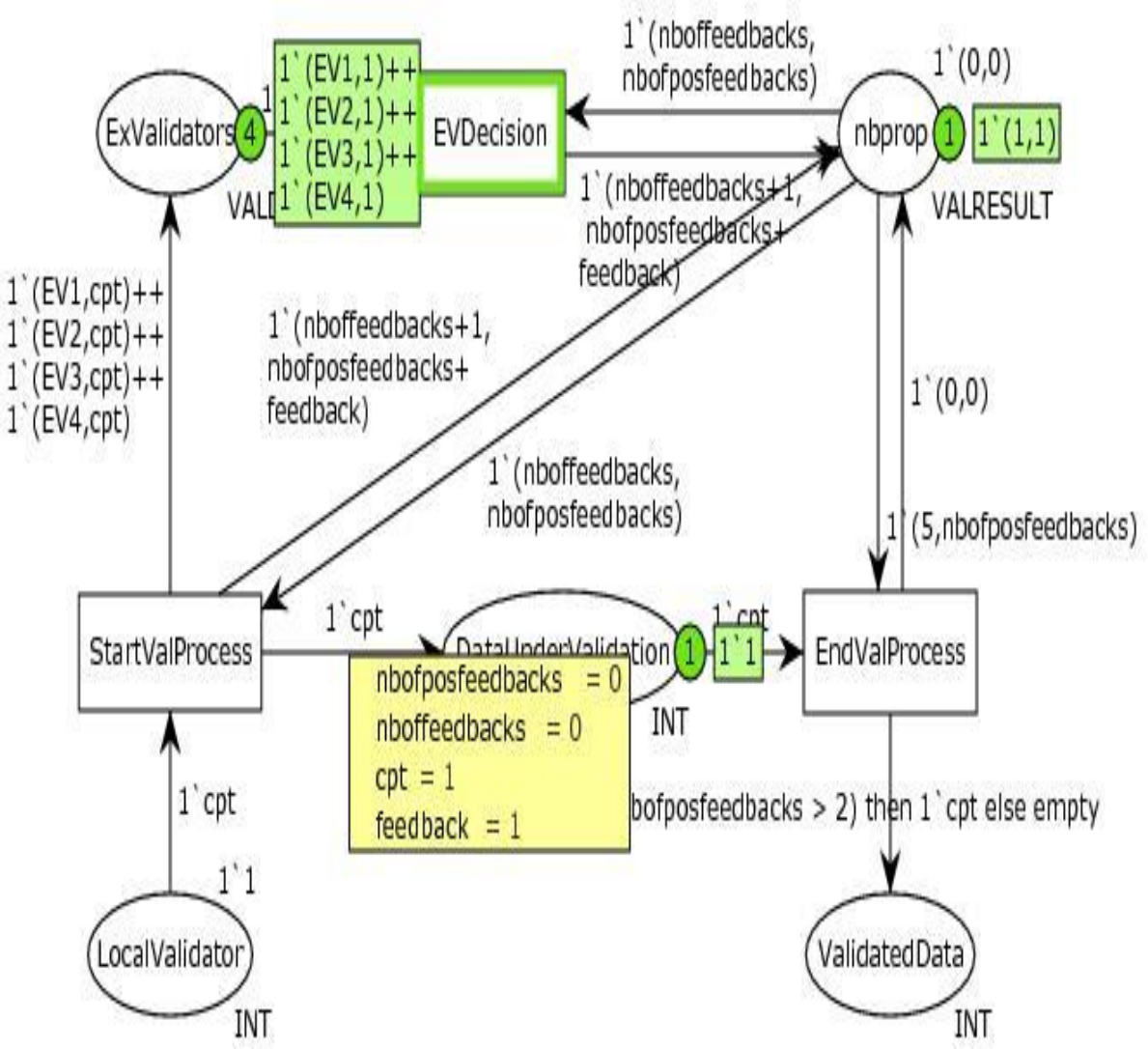

Figure 48: Local validator feedback decision proposal update

Figure 48 shows a feedback decision of '1' on the data under validation. The feedback from the local validation confirms that the "start validation transition" has been fired. The update on the number of proposals "nbprop" of 1`(1,1) shows that voting has started on the decision feedback on the data under validation. That only 1 validator has voted on the decision feedback. That decision is a positive decision (1 – 'Number of decisions',1-'number of positive decisions').

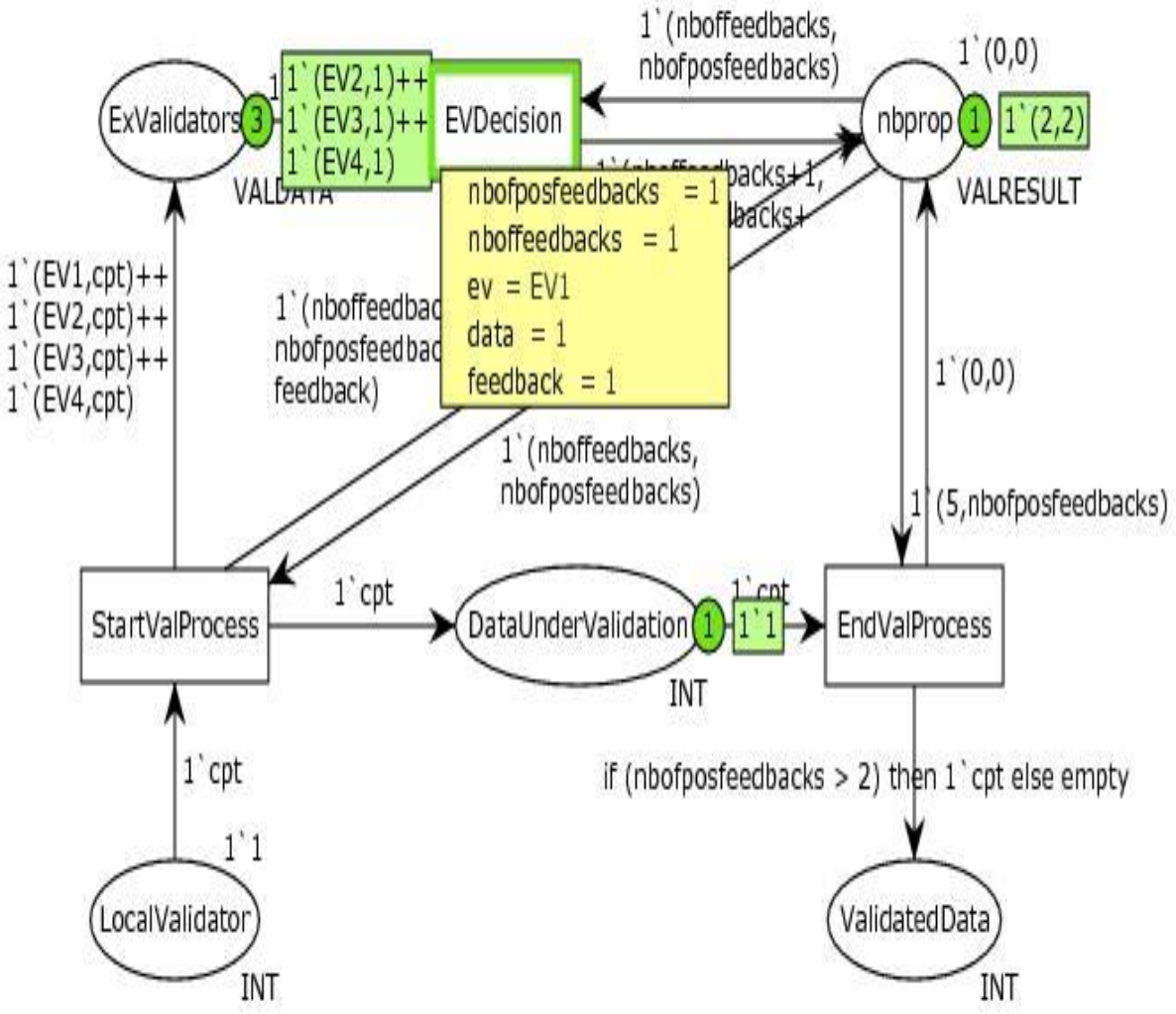

Figure 49: External validator1 feedback decision

Figure 49 depicts the feedback decision from external validator1 on the data under validation.

The update on the "nbprop" place 1`(2,2) shows that there have been two voting decisions and all the decision are positive decisions.

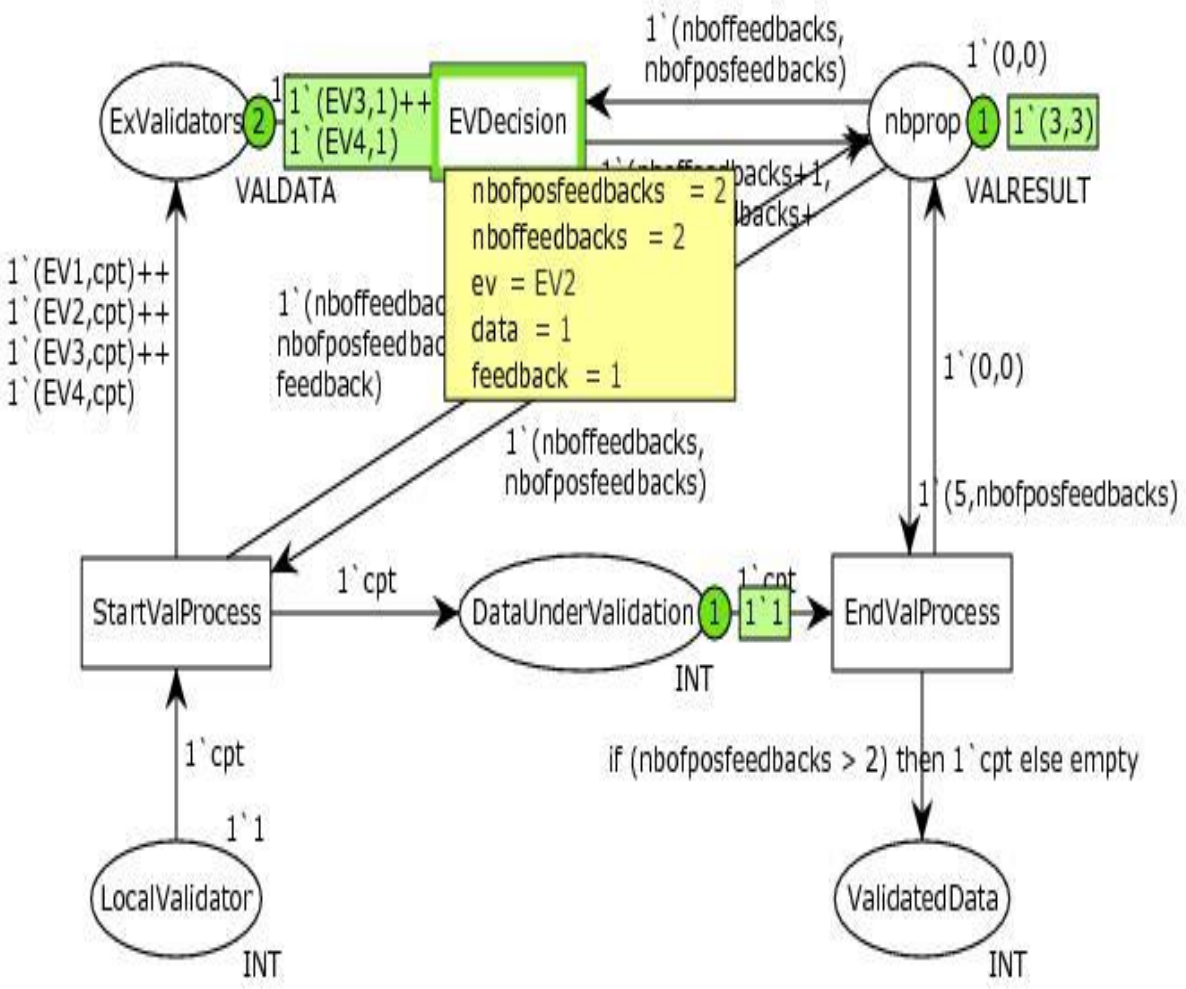

Figure 50: External validator2 feedback decision proposal update

Figure 50 illustrates the feedback decision from external validator2 on the data under validation

The decision feedback voting update on the number of proposals "nbprop" – 1`(3,3) shows that there have been 3 feedback decisions with all 3 being positive feedback decisions.

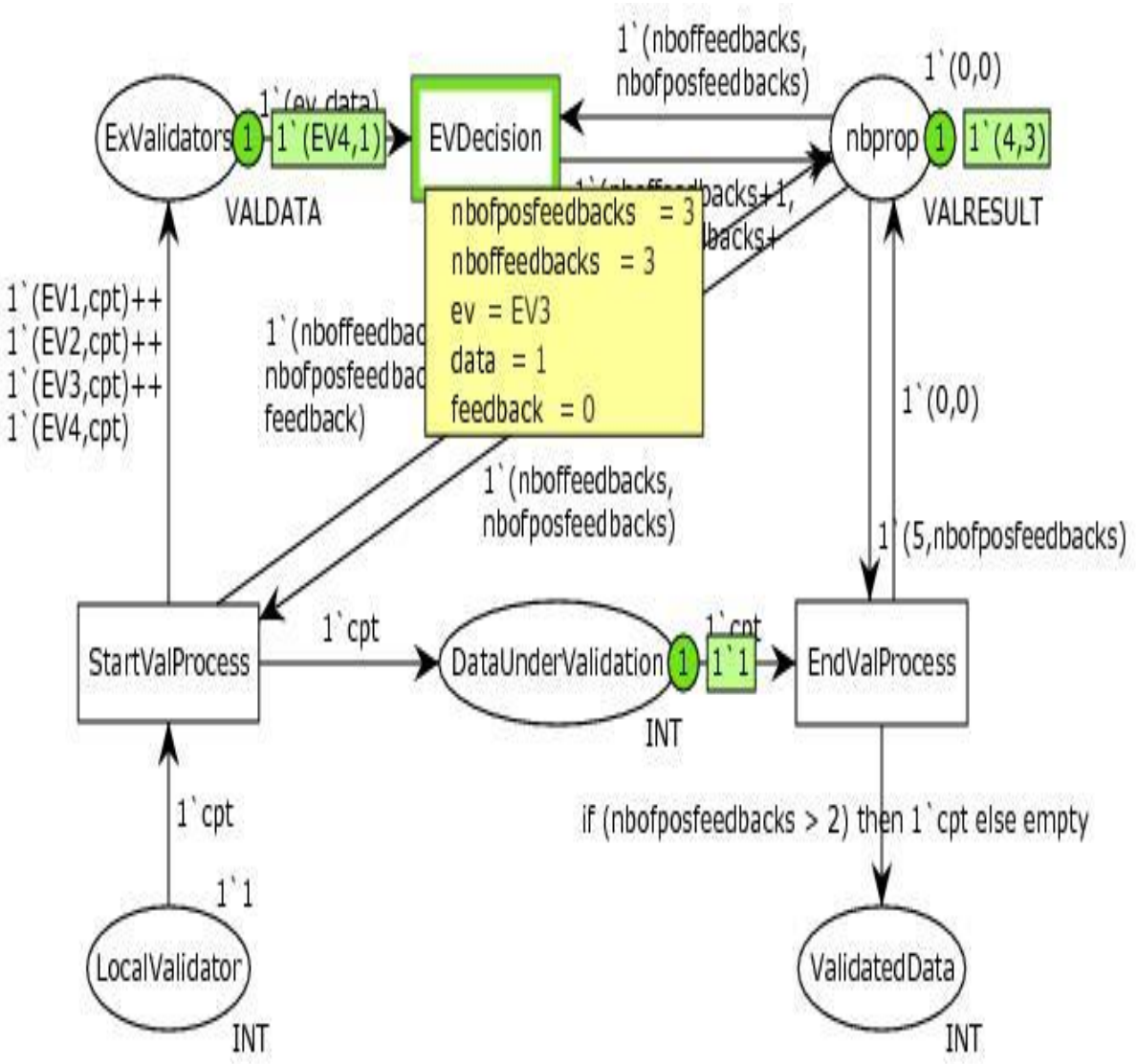

Figure 51: External validator3 feedback decision proposal update

Figure 51 illustrates the feedback decision from external validator3 on the data under validation. The local validator, external validators 1, 2,3 have all voted on the decision and have the feedback updated and stored on the nbprop place. The token value has been updated to 1`(4,3) to show that there have been four votes (local validator, external validators 1,2,3). And that 3 out of the 4 votes are positive feedback decisions.

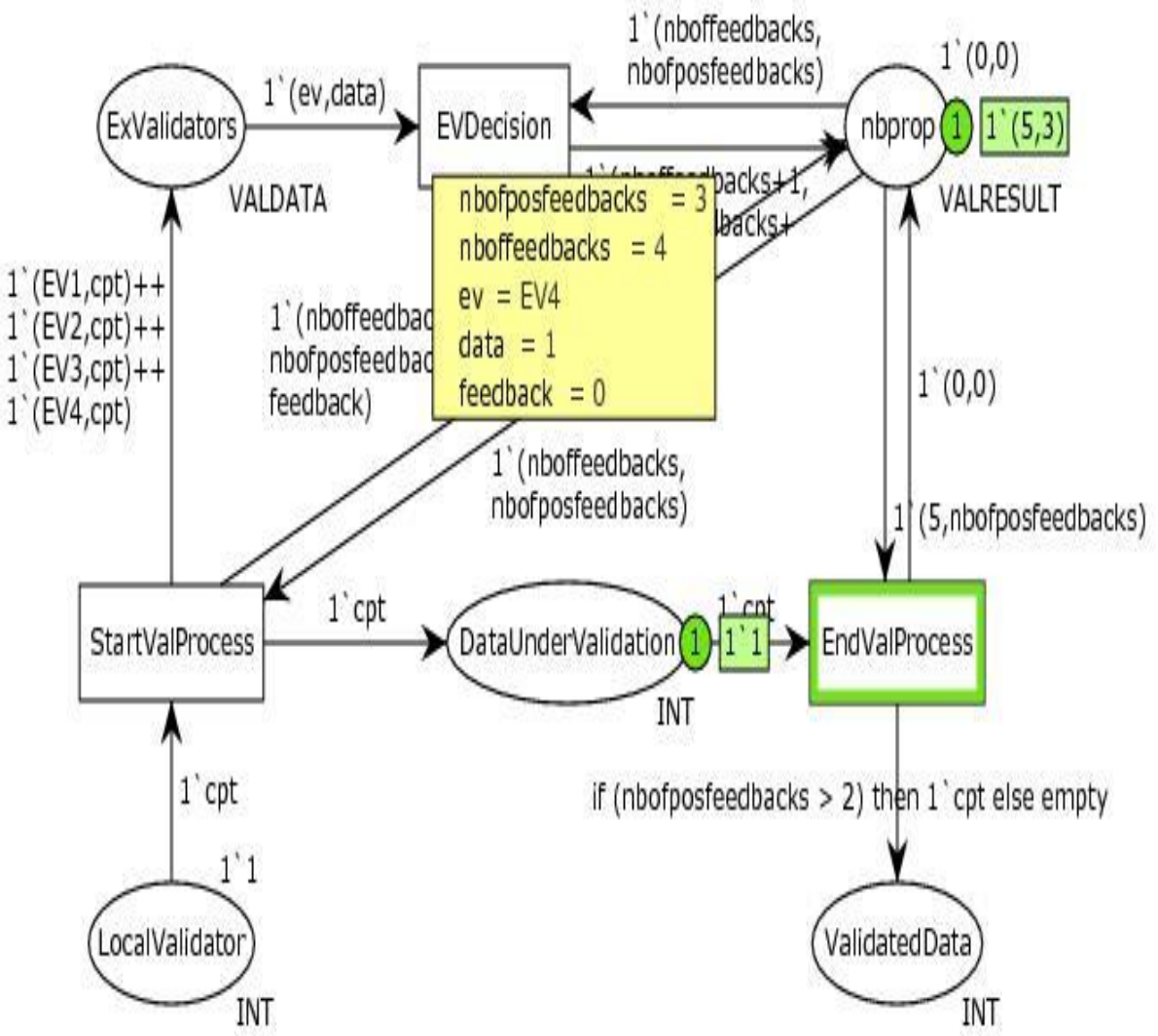

Figure 52: External validator4 feedback decision proposal update

Figure 52 depicts the feedback decision from external validator4 on the data under validation. Additionally, it provides the update as illustrated in the place for the number of proposals “nbprop” for a total of 5 decisions, with 3 positive feedback decisions. The EndValProcess transition is highlighted to show that it is the next action or step to be taken for the simulation.

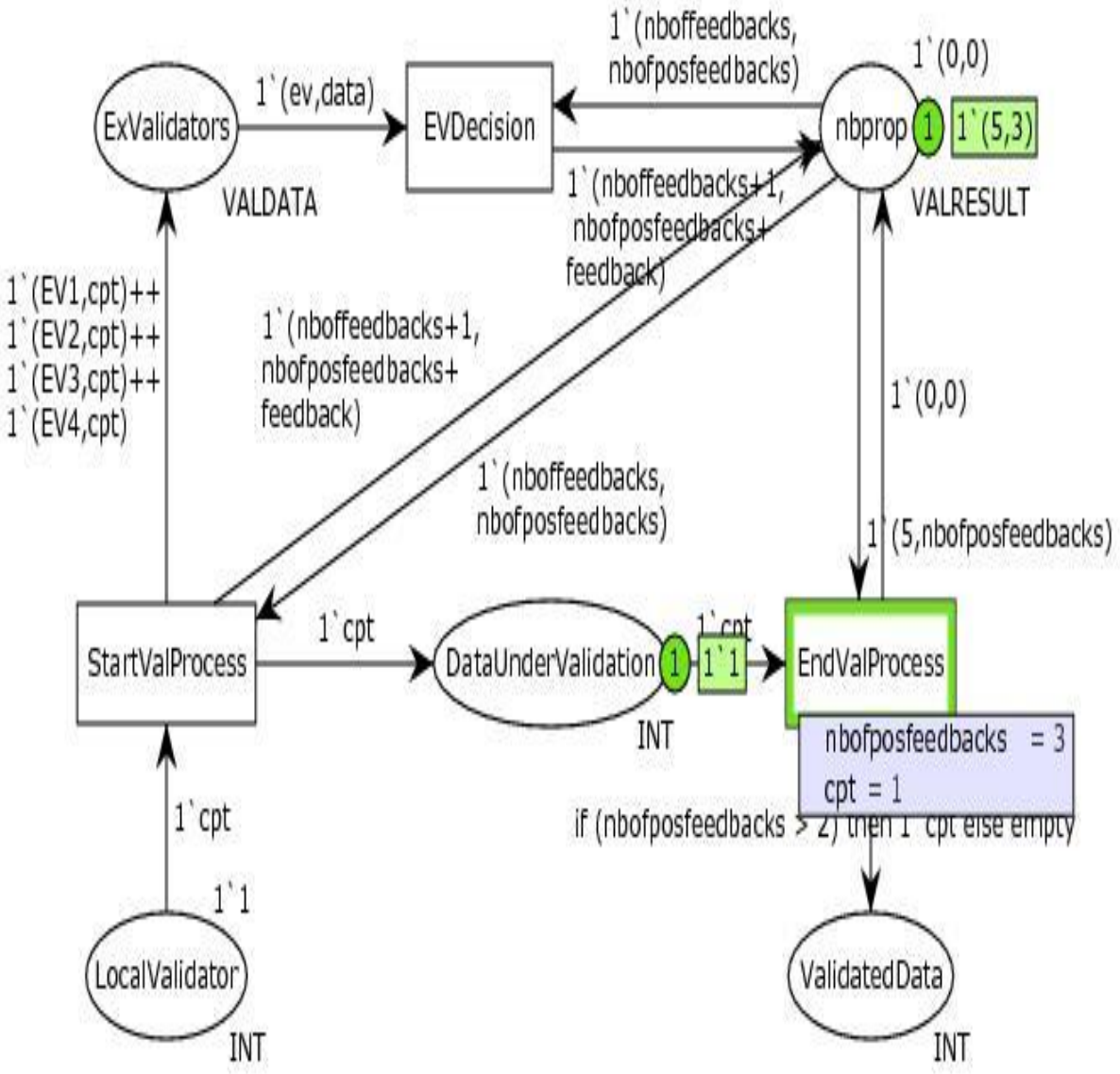

Figure 53: Decision result after the proposals

Figure 53 shows the data flow CPN simulation on the decision feedback results at the end of the decision voting process. The token value on the nbprop 1`(5,3) and the summary information on the transition confirm same that there was one data identity that represented 1 data element to be validated and that there were 3 positive decision feedbacks.

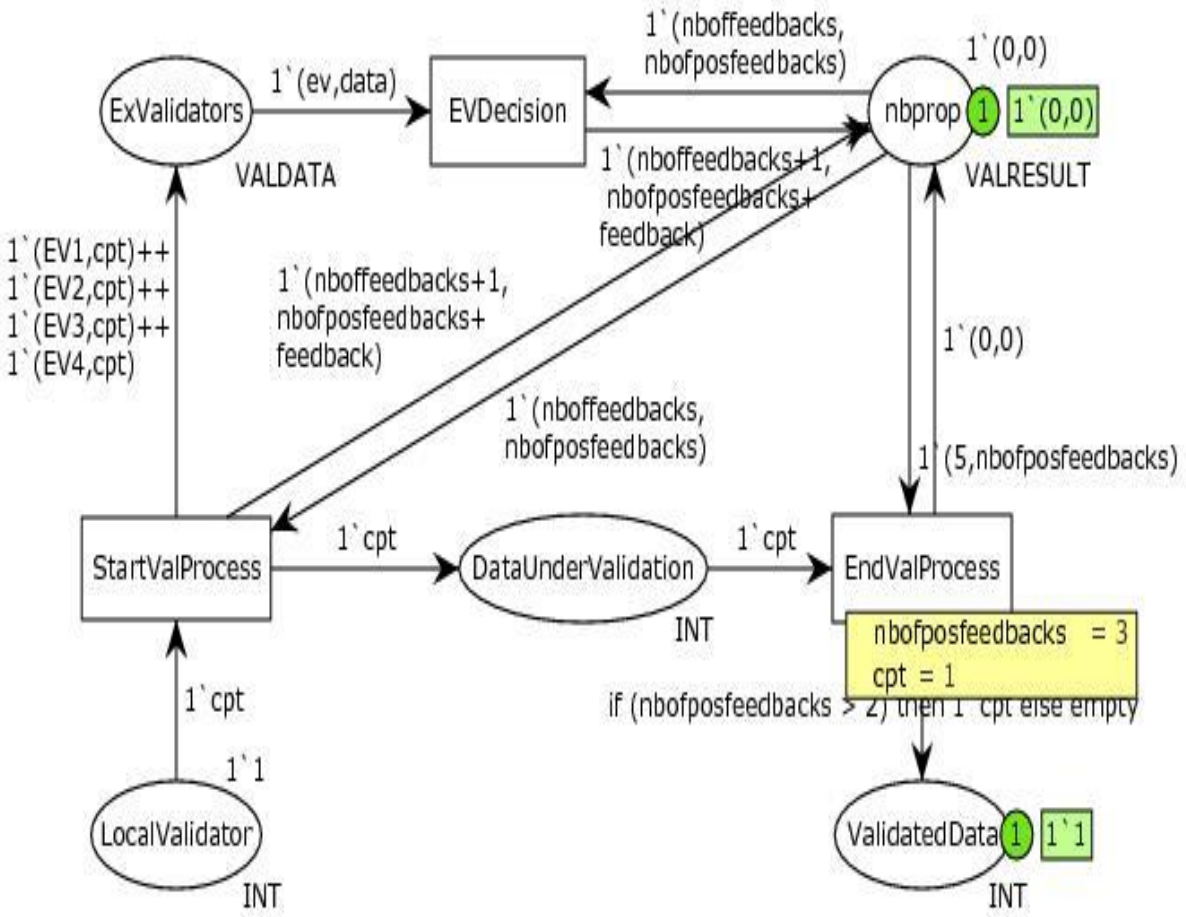

Figure 54: Consensus decision

Figure 54 demonstrates the decision on the data at the end of the consensus process. The initial token element 1`1 on the local validator has been moved to the place for the ValidatedData. Additionally, the consensus session is closed and the nbprop token element is reset to 1`(0,0).

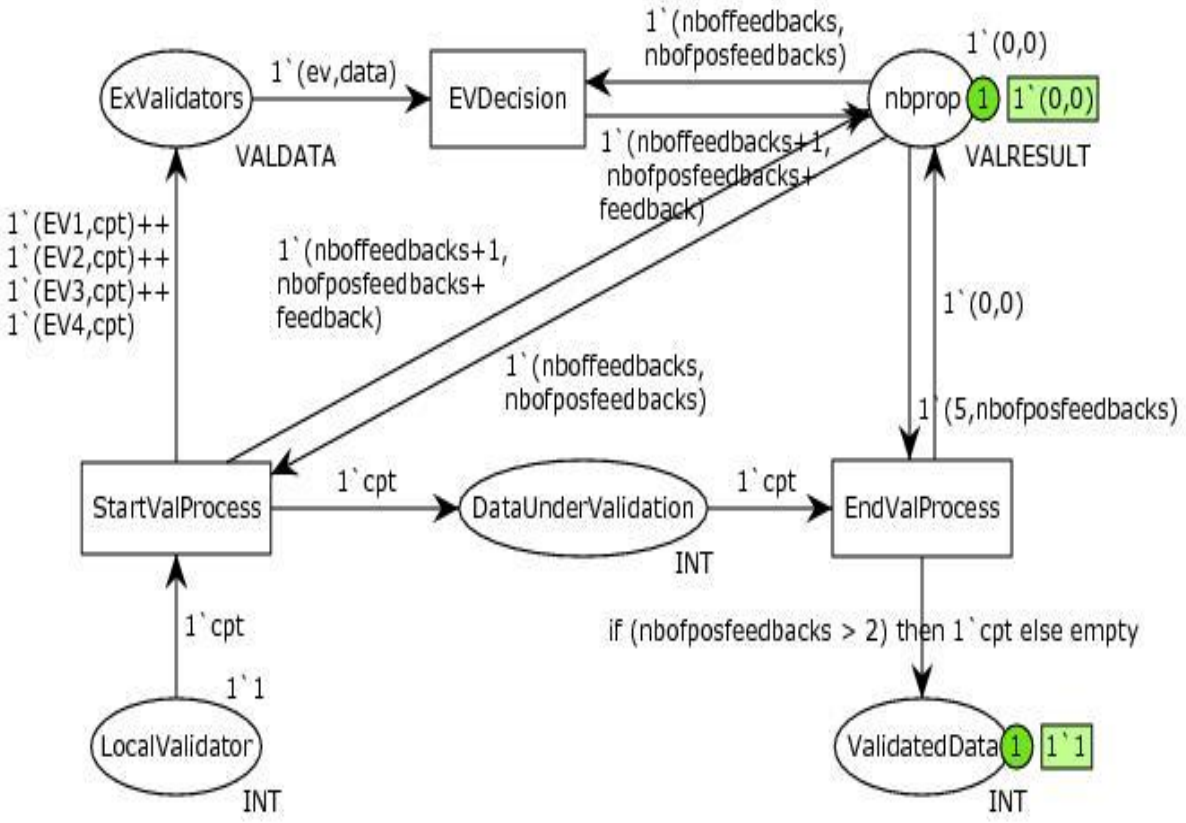

Figure 55: Validated data

Figure 55 shows the decision on the data under validation using the blockchain consensus. The data has been validated.

In the next subsection, the conclusion on the validation of the blockchain-based IoT data authentication system is presented.

5.5 Conclusion on the Validation of the Blockchain-based IoT Data Authentication System

To validate the proposed architecture, the entire system was modelled, simulated, and verified using Coloured Petri net and CPN tools.

The modeling, simulation, and validation of the security properties of the blockchain-based authentication mechanism was done by transferring the expected design specifications and

properties of the system into a CPN design for simulation to verify the unique attributes of authenticating data. The CPN features including places, transitions, arcs, expressions, and token markings were used to represent the entities or physical attributes of the system as well as the design decisions of the system. The design decisions and the dynamic nature expected of the distributed ledger system were represented using places, transitions, arc, and tokens.

In the next chapter, the general conclusion and future research directions are presented.

Chapter 6: General Conclusion and Future Works

The chapter summarises the main contributions of the thesis and presents future works.

In the next subsection, the conclusion is presented.

6.1 Conclusion

The widespread adoption and use of IoT by small and large enterprise networks backed by the recent report of massive investments and budget allocations in cyber security by these networks to address the soaring incidences of cyber-attacks in these wireless sensor networks is an indicative gesture of the acceptance of the benefits that IoT offers in scaling and growing their network operations. Although there seems to be a parallel match in the rise of IoT network compromises and cyber-attacks on IoT data around the same period, the demand for an enhanced cryptographic-based solution to assure the privacy, security, and auditing of cyber activities within the information technology infrastructure has reached an all-time high. Authentication schemes have evolved from the classical simple password based to an improved cryptographic based solutions that operates using the peer-to-peer network approach. Existing classical IoT architectures serve as a major obstacle to the adoption of blockchain-based solution. Similarly, since classical blockchain-based authentication solution is based on peer-to-peer networking with heavy computational overhead, directly adopting the authentication approach used in a blockchain for existing classical IoT architecture where the main actors have energy, storage and computational processing constraints become challenge for an efficient implementation.

The primary contribution of the thesis is the design of an architecture that employ a blockchain-based authentication mechanism for IoT node data. Classical IoT architecture depend on a centralized authentication mechanism that introduces a single point of failure situation where the centralized mechanism functions as trusted third-party system for node

data authentication. The design of the architecture was based on a distributed ledger technology for a cyber-physical peer-to-peer security implementation that involved the IoT gateway sink node and the cloud storage. The design of a hierarchical IoT architecture appropriate for cyber-physical network provides a good reference point for system designers, network architects and the scientific community for future works.

Secondly, the thesis proposed a cryptographic algorithm that is based on the Feistel cryptographic approach for data authentication. The proposed lightweight cryptographic solution involved cryptographic mechanism and digital signature algorithm. The proposed cryptographic solution offered enhanced security capabilities whiles maintaining a lightweight structure that is appropriate for IoT environments since IoT devices including sensors and sink nodes have computational processing, storage, and power or energy limitations. The proposed cryptographic approach is useful for decentralized authentication for node data validation in a hierarchical network. Centralized authentication mechanisms introduce a single point of failure in the network. Decentralized authentication scheme that adopts blockchain-based authentication mechanism eliminate the situation of single point of failure in the network. In effect, the thesis designed and used a blockchain-based authentication for validating node data in a hierarchical network that include onsite network which is also known as the local IoT network and offsite network that represents a remote network component.

Finally, the thesis contributed to the design of a formal model for a generic blockchain architecture for internet of things systems with several use case applications.

The features in CPN tools were used for validating the security properties in the proposed blockchain-based IoT architecture for data authentication. The CPN modelling of the blockchain-based consensus mechanism provided a visualization platform for validating the critical features of the digital ledger. The unique features of transparent transaction,

permanent storage, and decentralized authentication. The authenticated IoT data is stored on a distributed ledger whose storage is synchronized between the IoT gateway persistent storage (on-site network) and the cloud storage network (remote network).

In the next subsection, the future research direction is presented.

6.2 Future Research Directions

Although the thesis made significant use of available tools and platforms to model and simulate security solution that assured the privacy, availability, and integrity of node data using blockchain-based cryptographic techniques, our future work is planned to analyse the complexity of the entire system with respect to operational, time and space complexities in the context of constrained IoT devices. Additionally, we will develop a smart contract using available blockchain smart contract platforms to implement the storage of node data onto a distributed ledger with direct involvement and use of existing cloud-based blockchain technologies like the Hyperledger.

References

[1] R. Geissbauer, J. Vedso, and S. Schrauf, “Industry 4.0: Building the Digital Enterprise. 2016,” *Accessed Online Novemb.*, vol. 10, 2018.

[2] S. H. Mahmud, L. Assan, and R. Islam, “Potentials of internet of things (IoT) in Malaysian construction industry,” *Ann. Emerg. Technol. Comput. AETiC Print ISSN*, pp. 2516–0281, 2018.

[3] V. G. Cruz, “Global number of connected IoT devices 2015-2025,” *Stat. Mar*, 2021.

[4] F. Daneshgar, O. Ameri Sianaki, and P. Guruwacharya, “Blockchain: a research framework for data security and privacy,” in *Workshops of the International Conference on Advanced Information Networking and Applications*, 2019, pp. 966–974.

[5] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, “DDoS in the IoT: Mirai and other botnets,” *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[6] B. T. Asare, K. Quist-aphetsi, and L. Nana, *Towards a Secure Communication of Data in IoT Networks : A Technical Research Report*, vol. 4. Springer International Publishing, 2020. doi: 10.1007/978-3-030-60248-2.

[7] A. Haouzia and R. Noumeir, “Methods for image authentication: a survey,” *Multimed. Tools Appl.*, vol. 39, no. 1, pp. 1–46, 2008.

[8] I. J. Cox, J. Kilian, F. T. Leighton, and T. Shamoon, “Secure spread spectrum watermarking for multimedia,” *IEEE Trans. Image Process.*, vol. 6, no. 12, pp. 1673–1687, 1997.

[9] M. Conoscenti, A. Vetro, and J. C. De Martin, “Blockchain for the Internet of Things: A systematic literature review,” in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, 2016, pp. 1–6.

[10] A. Lohachab, A. Lohachab, and A. Jangra, “A comprehensive survey of prominent cryptographic aspects for securing communication in post-quantum IoT networks,” *Internet Things*, vol. 9, p. 100174, 2020.

[11] R. E. Klima, R. Klima, N. P. Sigmon, and N. Sigmon, *Cryptology: classical and modern*. Chapman and Hall/CRC, 2018.

[12] A. Machiavelo and R. Reis, “Automated Ciphertext-Only Cryptanalysis of the Bifid Cipher,” *Cryptologia*, vol. 31, no. 2, pp. 112–124, Apr. 2007, doi: 10.1080/01611190601134222.

[13] N. Kopal, “Solving classical ciphers with CrypTool 2,” 2018, no. 149, pp. 29–38.

[14] D. Rachmawati, S. M. Hardi, and R. P. Pasaribu, “Combination of columnar transposition cipher caesar cipher and lempel ziv welch algorithm in image security and compression,” in *Journal of Physics: Conference Series*, 2019, vol. 1339, no. 1, p. 012007.

[15] P. K. Pradhan, S. Rakshit, and S. Datta, “Lattice based cryptography: Its applications, areas of interest & future scope,” in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, 2019, pp. 988–993.

[16] U. Menon, A. Hudlikar, and D. Panda, “Scytale--An Evolutionary Cryptosystem,” *ArXiv Prepr. ArXiv200805290*, 2020.

[17] S. K. Pal and S. Mishra, “Revolutionary Change in Cryptography,” *Invertis J. Renew. Energy*, vol. 9, no. 2, pp. 43–54, 2019.

[18] T. M. Aung and N. N. Hla, “A Complex Polyalphabetic Cipher Technique Myanmar Polyalphabetic Cipher,” in *2019 International Conference on Computer Communication and Informatics (ICCCI)*, 2019, pp. 1–9.

[19] R. M. Marzan and A. M. Sison, “An enhanced key security of Playfair cipher algorithm,” in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, 2019, pp. 457–461.

[20] M. Ahmid, O. Kazar, S. Benharzallah, L. Kahloul, and A. Merizig, “An intelligent and secure health monitoring system based on agent,” in *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, 2020, pp. 291–296.

[21] S. Rubinstein-Salzedo, “The vigenere cipher,” in *Cryptography*, Springer, 2018, pp. 41–54.

[22] A. J. Abd and S. Al-Janabi, “Classification and identification of classical cipher type using artificial neural networks,” *J. Eng. Appl. Sci.*, vol. 14, no. 11, pp. 3549–3556, 2019.

[23] J. C. T. Arroyo and A. J. P. Delima, “Caesar cipher with goldbach code compression for efficient cryptography,” *Int J Emerg Trends Eng Res*, 2020.

[24] A. Singh and S. Sharma, “Enhancing data security in cloud using Split algorithm, Caesar cipher, and Vigenere cipher, homomorphism encryption scheme,” in *Emerging Trends in Expert Applications and Security*, Springer, 2019, pp. 157–166.

[25] G. Lasry, *A methodology for the cryptanalysis of classical ciphers with search metaheuristics*. kassel university press GmbH, 2018.

[26] D. Buell, “Modern Symmetric Ciphers—DES and AES,” in *Fundamentals of Cryptography: Introducing Mathematical and Algorithmic Foundations*, Cham: Springer International Publishing, 2021, pp. 123–147. doi: 10.1007/978-3-030-73492-3_9.

[27] E. Valea, M. Da Silva, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, “Stream vs block ciphers for scan encryption,” *Microelectron. J.*, vol. 86, pp. 65–76, 2019.

[28] J. Kelsey, B. Schneier, and D. Wagner, “Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES,” in *Advances in Cryptology — CRYPTO ’96*, Berlin, Heidelberg, 1996, pp. 237–251.

[29] K. Quist-Aphetsi, B. T. Asare, and L. Nana, “IoT node-node secure communication using RIPEMD-128 and des,” *Proc. - 2019 Int. Conf. Cyber Secur. Internet Things ICSIoT 2019*, pp. 62–65, 2019, doi: 10.1109/ICSIoT47925.2019.00018.

[30] B. T. Asare, Q. Kester, and L. Nana, “Secure Data Exchange Between Nodes In IoT Using TwoFish and DHE,” pp. 2–5.

[31] B. T. Asare, K. Quist-Aphetsi, and L. Nana, “Using RC4 and whirlpool for the encryption and validation of data in IoT,” *Proc. - 2019 Int. Conf. Cyber Secur. Internet Things ICSIoT 2019*, pp. 114–117, 2019, doi: 10.1109/ICSIoT47925.2019.00027.

[32] N. Li, “Research on diffie-hellman key exchange protocol,” *ICCET 2010 - 2010 Int. Conf. Comput. Eng. Technol. Proc.*, vol. 4, no. 4, pp. V4-634-V4-637, 2010, doi: 10.1109/ICCET.2010.5485276.

[33] P. Rewagad and Y. Pawar, “Use of Digital Signature with Diffie Hellman Key Exchange and AES Encryption Algorithm to Enhance Data Security in Cloud Computing,” *2013 Int. Conf. Commun. Syst. Netw. Technol.*, pp. 437–439, 2013, doi: 10.1109/CSNT.2013.97.

[34] Q. A. Kester, L. Nana, A. C. Pascu, and S. Gire, “A new encryption cipher for securing digital images of video surveillance devices using Diffie-HellMan-MD5 algorithm and RGB pixel shuffling,” *Proc. - UKSim-AMSS 7th Eur. Model. Symp. Comput. Model. Simul. EMS 2013*, pp. 305–311, 2013, doi: 10.1109/EMS.2013.53.

[35] G. Wang and H. Yu, “Improved cryptanalysis on RIPEMD-128,” *IET Inf. Secur.*, 2015, doi: 10.1049/iet-ifs.2014.0244.

[36] H. Dobbertin, A. Bosselaers, and B. Preneel, “RIPEMD-160: A strengthened version of RIPEMD,” in *Fast Software Encryption*, Berlin, Heidelberg, 1996, pp. 71–82. doi: https://doi.org/10.1007/3-540-60865-6_44.

[37] W. Li *et al.*, “Security analysis of the whirlpool hash function in the cloud of things,” *KSII Trans. Internet Inf. Syst.*, 2017, doi: 10.3837/tiis.2017.01.028.

[38] F. Mendel and V. Rijmen, “Cryptanalysis of the tiger hash function,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture*

Notes in Bioinformatics), 2007, vol. 4833 LNCS, pp. 536–550. doi: 10.1007/978-3-540-76900-2_33.

[39] R. Damasevicius, G. Ziberkas, V. Stuikys, and J. Toldinas, "Energy consumption of hash functions," *Elektron. Ir Elektrotechnika*, vol. 18, no. 10, pp. 81–84, 2012, doi: 10.5755/j01.eee.18.10.3069.

[40] K. M. Prabha and P. V. Saraswathi, "Tiger hash kerberos biometric blowfish user authentication for secured data access in cloud," 2019. doi: 10.1109/I-SMAC.2018.8653713.

[41] A. Sharma and S. Mittal, *COMPARATIVE ANALYSIS OF CRYPTOGRAPHIC HASH FUNCTIONS*. 2018.

[42] M. Lavanya and V. Natarajan, "LWDSA: light-weight digital signature algorithm for wireless sensor networks," *Sādhanā*, vol. 42, no. 10, pp. 1629–1643, 2017.

[43] A. A. Yavuz and M. O. Ozmen, "Ultra Lightweight Multiple-time Digital Signature for the Internet of Things Devices," *IEEE Trans. Serv. Comput.*, 2019, doi: 10.1109/TSC.2019.2928303.

[44] P. Gupta, A. Sinha, P. Kr. Srivastava, A. Perti, and A. K. Singh, "Security Implementations in IoT Using Digital Signature," in *Innovations in Electrical and Electronic Engineering*, vol. 661, M. N. Favorskaya, S. Mekhilef, R. K. Pandey, and N. Singh, Eds. Singapore: Springer Singapore, 2021, pp. 523–535. doi: 10.1007/978-981-15-4692-1_40.

[45] D. R. M. N. Sindhu, "Secure Elliptic Curve Digital Signature Algorithm for Internet of Things," *Glob. J. Comput. Sci. Technol.*, 2016, [Online]. Available: https://computerresearch.org/index.php/computer/article/view/1376

[46] G. M. Abdullah, Q. Mehmood, and C. B. A. Khan, "Adoption of Lamport signature scheme to implement digital signatures in IoT," in *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, Sukkur, Mar. 2018, pp. 1–4. doi: 10.1109/ICOMET.2018.8346359.

[47] L. Chen, D. Moody, A. Regenscheid, and K. Randall, “Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters,” National Institute of Standards and Technology, 2019.

[48] J. Brendel, C. Cremers, D. Jackson, and M. Zhao, “The provable security of ed25519: theory and practice,” in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 1659–1676.

[49] D. L. K. Chuen and R. H. Deng, *Handbook of Blockchain, Digital Finance, and Inclusion: Cryptocurrency, FinTech, InsurTech, Regulation, ChinaTech, Mobile Security, and Distributed Ledger*. Academic Press, 2017.

[50] S. Nakamoto, “Bitcoin : A Peer-to-Peer Electronic Cash System,” 2009.

[51] E. Barker, L. Chen, and R. Davis, “Recommendation for key-derivation methods in key-establishment schemes,” *NIST Spec. Publ.*, vol. 800, p. 56C, 2018.

[52] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, “Blockchain challenges and opportunities: A survey,” *Int. J. Web Grid Serv.*, vol. 14, no. 4, pp. 352–375, 2018.

[53] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, “Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab,” in *International conference on financial cryptography and data security*, 2016, pp. 79–94.

[54] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, “An Overview of Blockchain Technology : Architecture , Consensus , and Future Trends,” 2017, doi: 10.1109/BigDataCongress.2017.85.

[55] R. Beck, “Beyond bitcoin: The rise of blockchain world,” *Computer*, vol. 51, no. 2, pp. 54–58, 2018.

[56] Y. Yuan and F.-Y. Wang, “Blockchain and cryptocurrencies: Model, techniques, and applications,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 48, no. 9, pp. 1421–1428, 2018.

[57] I. Grishchenko, M. Maffei, and C. Schneidewind, “A semantic framework for the security analysis of ethereum smart contracts,” in *International Conference on Principles of Security and Trust*, 2018, pp. 243–269.

[58] E. Hildenbrandt *et al.*, “Kevm: A complete formal semantics of the ethereum virtual machine,” in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, 2018, pp. 204–217.

[59] X. Liang, S. S. Shetty, D. Tosh, L. Njilla, C. A. Kamhoua, and K. Kwiat, “ProvChain: Blockchain-based Cloud Data Provenance,” in *Blockchain for Distributed Systems Security*, John Wiley & Sons, Ltd, 2019, pp. 67–94. doi: https://doi.org/10.1002/9781119519621.ch4.

[60] H. Okada, S. Yamasaki, and V. Bracamonte, “Proposed classification of blockchains based on authority and incentive dimensions,” in *2017 19th international conference on advanced communication technology (ICACT)*, 2017, pp. 593–597.

[61] M. Walker, *Front-to-Back Designing and Changing Trade Processing Infrastructure*. Risk Books, 2018.

[62] “embracing-disruption.pdf.” Accessed: Mar. 08, 2022. [Online]. Available: https://www.dtcc.com/~/media/Files/Downloads/WhitePapers/embracing-disruption.pdf

[63] “WEF_Digital_Assets_Distributed_Ledger_Technology_2021.pdf.” Accessed: Mar. 08, 2022. [Online]. Available: https://www3.weforum.org/docs/WEF_Digital_Assets_Distributed_Ledger_Technology_2021.pdf

[64] A. Gaul, I. Khoffi, J. Liesen, and T. Stüber, “Mathematical Analysis and Algorithms for Federated Byzantine Agreement Systems,” pp. 1–42, 2019.

[65] X. Wang, H. Su, X. Wang, and G. Chen, “Fully distributed event-triggered semiglobal consensus of multi-agent systems with input saturation,” *IEEE Trans. Ind. Electron.*, vol. 64, no. 6, pp. 5055–5064, 2016.

[66] R. Di Pietro, X. Salleras, M. Signorini, and E. Waisbard, “A blockchain-based trust system for the internet of things,” in *Proceedings of the 23rd ACM on symposium on access control models and technologies*, 2018, pp. 77–83.

[67] J. Yang, Z. Lu, and J. Wu, “Smart-toy-edge-computing-oriented data exchange based on blockchain,” *J. Syst. Archit.*, vol. 87, pp. 36–48, 2018.

[68] A. Baliga, “Understanding blockchain consensus models,” *Persistent*, vol. 4, pp. 1–14, 2017.

[69] A. Kashevnik and N. Teslya, “Blockchain-oriented coalition formation by cps resources: Ontological approach and case study,” *Electronics*, vol. 7, no. 5, p. 66, 2018.

[70] Z. Li, A. V. Barenji, and G. Q. Huang, “Toward a blockchain cloud manufacturing system as a peer-to-peer distributed network platform,” *Robot. Comput.-Integr. Manuf.*, vol. 54, pp. 133–144, 2018.

[71] M. Ruta, F. Scioscia, S. Ieva, G. Capurso, and E. Di Sciascio, “Semantic blockchain to improve scalability in the internet of things,” *Open J. Internet Things OJIOT*, vol. 3, no. 1, pp. 46–61, 2017.

[72] A. Kiayias and D. Zindros, “Proof-of-work sidechains,” in *International Conference on Financial Cryptography and Data Security*, 2019, pp. 21–34.

[73] V. Gramoli, “From blockchain consensus back to Byzantine consensus,” *Future Gener. Comput. Syst.*, vol. 107, pp. 760–769, 2020.

[74] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, “Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities,” *IEEE Access*, vol. 7, pp. 85727–85745, 2019.

[75] D. Wang, C. Jin, H. Li, and M. Perkowski, “Proof of Activity Consensus Algorithm Based on Credit Reward Mechanism,” in *Web Information Systems and Applications: 17th*

International Conference, WISA 2020, Guangzhou, China, September 23–25, 2020, Proceedings, Berlin, Heidelberg, 2020, pp. 618–628. doi: 10.1007/978-3-030-60029-7_55.

[76] B. Chase and E. MacBrough, "Analysis of the XRP ledger consensus protocol," *ArXiv Prepr. ArXiv180207242*, 2018.

[77] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab," in *Financial Cryptography and Data Security*, vol. 9604, J. Clark, S. Meiklejohn, P. Y. A. Ryan, D. Wallach, M. Brenner, and K. Rohloff, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 79–94. doi: 10.1007/978-3-662-53357-4_6.

[78] T. Kramp, R. van Kranenburg, and S. Lange, *Introduction to the internet of things*. 2013. doi: 10.1007/978-3-642-40403-0_1.

[79] K. P. Lakshmi, K. Archana, Y. Prasanthi, and V. Padma, "A Study on Internet of Things with Blockchain Technology," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, Apr. 2019, pp. 372–376. doi: 10.1109/ICOEI.2019.8862509.

[80] A. Rayes and S. Salam, "Internet of things (IoT) overview," in *Internet of Things from hype to reality*, Springer, 2019, pp. 1–35.

[81] A. H. Mohd Aman, E. Yadegaridehkordi, Z. S. Attarbashi, R. Hassan, and Y.-J. Park, "A Survey on Trend and Classification of Internet of Things Reviews," *IEEE Access*, vol. 8, pp. 111763–111782, 2020, doi: 10.1109/ACCESS.2020.3002932.

[82] X. Li, Y. Lu, X. Fu, and Y. Qi, "Building the Internet of Things platform for smart maternal healthcare services with wearable devices and cloud computing," *Future Gener. Comput. Syst.*, vol. 118, pp. 282–296, 2021, doi: https://doi.org/10.1016/j.future.2021.01.016.

[83] "Minoli et al. - 2017 - IoT Considerations, Requirements, and Architecture.pdf."

[84] D. Minoli and B. Occhiogrosso, “Blockchain mechanisms for IoT security,” *Internet Things*, vol. 1–2, pp. 1–13, 2018, doi: https://doi.org/10.1016/j.iot.2018.05.002.

[85] A. Ali, G. A. Shah, M. O. Farooq, and U. Ghani, “Technologies and challenges in developing Machine-to-Machine applications: A survey,” *J. Netw. Comput. Appl.*, vol. 83, pp. 124–139, 2017, doi: https://doi.org/10.1016/j.jnca.2017.02.002.

[86] M. O. Ojo, S. Giordano, G. Procissi, and I. N. Seitanidis, “A Review of Low-End, Middle-End, and High-End Iot Devices,” *IEEE Access*, vol. 6, pp. 70528–70554, 2018, doi: 10.1109/ACCESS.2018.2879615.

[87] T. Muhammed, R. Mehmood, A. Albeshri, and I. Katib, “UbeHealth: A Personalized Ubiquitous Cloud and Edge-Enabled Networked Healthcare System for Smart Cities,” *IEEE Access*, vol. 6, pp. 32258–32285, 2018, doi: 10.1109/ACCESS.2018.2846609.

[88] M. Raza, N. Aslam, H. Le-Minh, S. Hussain, Y. Cao, and N. M. Khan, “A Critical Analysis of Research Potential, Challenges, and Future Directives in Industrial Wireless Sensor Networks,” *IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 39–95, 2018, doi: 10.1109/COMST.2017.2759725.

[89] O. B. Sezer, E. Dogdu, and A. M. Ozbayoglu, “Context-Aware Computing, Learning, and Big Data in Internet of Things: A Survey,” *IEEE Internet Things J.*, vol. 5, no. 1, pp. 1–27, 2018, doi: 10.1109/JIOT.2017.2773600.

[90] T. M. Fernández-Caramés and P. Fraga-Lamas, “A Review on Human-Centered IoT-Connected Smart Labels for the Industry 4.0,” *IEEE Access*, vol. 6, pp. 25939–25957, 2018, doi: 10.1109/ACCESS.2018.2833501.

[91] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, “The industrial internet of things (IIoT): An analysis framework,” *Comput. Ind.*, vol. 101, pp. 1–12, Oct. 2018, doi: 10.1016/j.compind.2018.04.015.

[92] Y. A. Qadri, A. Nauman, Y. B. Zikria, A. V. Vasilakos, and S. W. Kim, “The Future of Healthcare Internet of Things: A Survey of Emerging Technologies,” *IEEE Commun. Surv. Tutor.*, vol. 22, no. 2, pp. 1121–1167, 2020, doi: 10.1109/COMST.2020.2973314.

[93] T. A. A. Ali, V. Choksi, and M. B. Potdar, “Precision Agriculture Monitoring System Using Green Internet of Things (G-IoT),” in *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, 2018, pp. 481–487. doi: 10.1109/ICOEI.2018.8553866.

[94] I. Chew, D. Karunatilaka, C. P. Tan, and V. Kalavally, “Smart lighting: The way forward? Reviewing the past to shape the future,” *Energy Build.*, vol. 149, pp. 180–191, 2017.

[95] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, “A comprehensive survey on fog computing: State-of-the-art and research challenges,” *IEEE Commun. Surv. Tutor.*, vol. 20, no. 1, pp. 416–464, 2017.

[96] T. M. Fernández-Caramés and P. Fraga-Lamas, “A Review on the Use of Blockchain for the Internet of Things,” *Ieee Access*, vol. 6, pp. 32979–33001, 2018.

[97] Z. Ashfaq *et al.*, “A review of enabling technologies for internet of medical things (IOMT) ecosystem,” *Ain Shams Eng. J.*, vol. 13, no. 4, p. 101660, 2022.

[98] A. Antony and S. S., “A Review on IoT Operating Systems,” *Int. J. Comput. Appl.*, vol. 176, no. 24, pp. 33–40, May 2020, doi: 10.5120/ijca2020920245.

[99] “Naik - 2017 - Choice of effective messaging protocols for IoT sy.pdf.”

[100] N. Naik, “Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP,” in *2017 IEEE international systems engineering symposium (ISSE)*, 2017, pp. 1–7.

[101] J. Sidna, B. Amine, N. Abdallah, and H. El Alami, “Analysis and evaluation of communication Protocols for IoT Applications,” in *Proceedings of the 13th international conference on intelligent systems: theories and applications*, 2020, pp. 1–6.

[102] “Sidna et al. - 2020 - Analysis and evaluation of communication Protocols.pdf.”

[103] C. Sharma and N. K. Gondhi, “Communication protocol stack for constrained IoT systems,” in *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, 2018, pp. 1–6.

[104] I. Lee, “The Internet of Things for enterprises: An ecosystem, architecture, and IoT service business model,” *Internet Things*, vol. 7, no. 2019, p. 100078, 2019, doi: 10.1016/j.iot.2019.100078.

[105] R. Ross, “Guide for Conducting Risk Assessments.” Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, Sep. 17, 2012. doi: https://doi.org/10.6028/NIST.SP.800-30r1.

[106] I. Butun, P. Österberg, and H. Song, “Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures,” *IEEE Commun. Surv. Tutor.*, vol. 22, no. 1, pp. 616–644, 2019.

[107] S. N. G. Gourisetti, M. Mylrea, and H. Patangia, “Cybersecurity vulnerability mitigation framework through empirical paradigm: Enhanced prioritized gap analysis,” *Future Gener. Comput. Syst.*, vol. 105, pp. 410–431, 2020.

[108] U. NCSC, “Understanding Vulnerabilities.” https://www.ncsc.gov.uk/search (accessed Sep. 05, 2022).

[109] M. Premkumar and T. V. P. Sundararajan, “DLDM: Deep learning-based defense mechanism for denial-of-service attacks in wireless sensor networks,” *Microprocess. Microsyst.*, vol. 79, p. 103278, 2020.

[110] F. Abdel-Fattah, K. A. Farhan, F. H. Al-Tarawneh, and F. AlTamimi, “Security challenges and attacks in dynamic mobile ad hoc networks MANETs,” in *2019 IEEE jordan international joint conference on electrical engineering and information technology (JEEIT)*, 2019, pp. 28–33.

[111] M. Tahir, M. Sardaraz, Z. Mehmood, and S. Muhammad, “CryptoGA: a cryptosystem based on genetic algorithm for cloud data security,” *Clust. Comput.*, vol. 24, no. 2, pp. 739–752, Jun. 2021, doi: 10.1007/s10586-020-03157-4.

[112] M. Sethi, A. Peltonen, and T. Aura, “Misbinding Attacks on Secure Device Pairing and Bootstrapping,” in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, Auckland New Zealand, Jul. 2019, pp. 453–464. doi: 10.1145/3321705.3329813.

[113] M. Tahir, M. Sardaraz, Z. Mehmood, and S. Muhammad, “CryptoGA: a cryptosystem based on genetic algorithm for cloud data security,” *Clust. Comput.*, vol. 24, no. 2, pp. 739–752, 2021.

[114] S. D. Galbraith, “CRYPTREC Review of EdDSA (Executive summary),” *CRYPTREC Rep. 2020*, p. 77, 2021.

[115] M. Elhoseny and K. Shankar, “Reliable data transmission model for mobile ad hoc network using signcryption technique,” *IEEE Trans. Reliab.*, vol. 69, no. 3, pp. 1077–1086, 2019.

[116] M. Tahir, M. Sardaraz, Z. Mehmood, and S. Muhammad, “CryptoGA: a cryptosystem based on genetic algorithm for cloud data security,” *Clust. Comput.*, vol. 24, no. 2, pp. 739–752, 2021.

[117] G. Kavallieratos, V. Gkioulos, and S. K. Katsikas, “Threat analysis in dynamic environments: The case of the smart home,” in *2019 15th international conference on distributed computing in sensor systems (DCOSS)*, 2019, pp. 234–240.

[118] “Threat Modeling,” *EC-Council*. https://www.eccouncil.org/threat-modeling/ (accessed Feb. 16, 2022).

[119] D. J. Bodeau, C. D. McCollum, and D. B. Fox, “Cyber threat modeling: Survey, assessment, and representative framework,” MITRE CORP MCLEAN VA MCLEAN, 2018.

[120] P. Aufner, “The IoT security gap: a look down into the valley between threat models and their implementation,” *Int. J. Inf. Secur.*, vol. 19, no. 1, pp. 3–14, Feb. 2020, doi: 10.1007/s10207-019-00445-y.

[121] A. Albalawi, A. Almrshed, A. Badhib, and S. Alshehri, “A survey on authentication techniques for the internet of things,” in *2019 International Conference on Computer and Information Sciences (ICCIS)*, 2019, pp. 1–5.

[122] R. Melki, H. N. Noura, and A. Chehab, “Lightweight multi-factor mutual authentication protocol for IoT devices,” *Int. J. Inf. Secur.*, vol. 19, no. 6, pp. 679–694, Dec. 2020, doi: 10.1007/s10207-019-00484-5.

[123] A. S. Patil, R. Hamza, H. Yan, A. Hassan, and J. Li, “Blockchain-PUF-Based Secure Authentication Protocol for Internet of Things,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Dec. 2020, vol. 11945 LNCS, pp. 331–338. doi: 10.1007/978-3-030-38961-1_29.

[124] V. Kumar, M. Ahmad, and A. Kumari, “A secure elliptic curve cryptography based mutual authentication protocol for cloud-assisted TMIS,” *Telemat. Inform.*, vol. 38, pp. 100–117, May 2019, doi: 10.1016/j.tele.2018.09.001.

[125] Q. F. Hassan, A. ur R. Khan, and S. A. Madani, *Internet of Things: Challenges, Advances, and Applications*. CRC Press, 2017.

[126] M. Ramírez Flores, G. Aguilar Torres, G. Gallegos García, and M. Á. García Licona, “Fingerprint verification using computational geometry,” *DYNA*, vol. 83, no. 195, pp. 128–137, Feb. 2016, doi: 10.15446/dyna.v83n195.46323.

[127] A. Esmat, M. de Vos, Y. Ghiassi-Farrokhfal, P. Palensky, and D. Epema, “A novel decentralized platform for peer-to-peer energy trading market with blockchain technology,” *Appl. Energy*, vol. 282, p. 116123, 2021.

[128] K. Košťál, P. Helebrandt, M. Belluš, M. Ries, and I. Kotuliak, “Management and Monitoring of IoT Devices Using Blockchain,” *Sensors*, vol. 19, no. 4, p. 856, Feb. 2019, doi: 10.3390/s19040856.

[129] S. Agrawal and P. Ahlawat, “A survey on the authentication techniques in internet of things,” in *2020 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2020, pp. 1–5.

[130] G. Wang, Z. Shi, M. Nixon, and S. Han, “ChainSplitter: Towards blockchain-based industrial IoT architecture for supporting hierarchical storage,” in *Proceedings - 2019 2nd IEEE International Conference on Blockchain, Blockchain 2019*, Jul. 2019, pp. 166–175. doi: 10.1109/Blockchain.2019.00030.

[131] S. N. G. Gourisetti *et al.*, “Standardization of the Distributed Ledger Technology cybersecurity stack for power and energy applications,” *Sustain. Energy Grids Netw.*, vol. 28, p. 100553, 2021.

[132] S. Josefsson and I. Liusvaara, “Edwards-Curve Digital Signature Algorithm (EdDSA),” *RFC*, vol. 8032, pp. 1–60, 2017.

[133] G. Fuchsbauer, E. Kiltz, and J. Loss, “The Algebraic Group Model and its Applications,” in *Advances in Cryptology – CRYPTO 2018*, Cham, 2018, pp. 33–62.

[134] N. I. N. I. S. Technology, *FIPS PUB 186-4 Digital Signature Standard (DSS): Subcategory: Cryptography*. CreateSpace Independent Publishing Platform, 2013. [Online]. Available: https://books.google.fr/books?id=amNMswEACAAJ

[135] G. Ayoade, V. Karande, L. Khan, and K. Hamlen, “Decentralized IoT data management using blockchain and trusted execution environment,” 2018, pp. 15–22.

[136] K. Jensen and L. M. Kristensen, *Coloured Petri nets: modelling and validation of concurrent systems*. Springer Science & Business Media, 2009.

[137] C. Gaucherel and F. Pommereau, “Using discrete systems to exhaustively characterize the dynamics of an integrated ecosystem,” *Methods Ecol. Evol.*, vol. 10, no. 9, pp. 1615–1627, 2019, doi: https://doi.org/10.1111/2041-210X.13242.

Publications

International Journal Publications

1. Bismark Tei Asare, Laurent Nana and Kester Quist-Aphetsi, “Modeling and Simulation of a Blockchain Consensus for IoT Node Data Validation” International Journal of Advanced Computer Science and Applications(IJACSA), 13(12), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0131204

2. Asare B.T., Quist-Aphetsi, K, and Nana, L. (2021) "Node-Node Data Exchange in IoT Devices Using Twofish and DHE", Advances in Science, Technology and Engineering Systems Journal, vol. 6, no. 2, pp. 622-628 (2021). DOI: 10.25046/aj060271

3. Asare, B.T., Quist-Aphetsi, K., & Nana, L. (2021). “A Cryptographic Technique for Communication among IoT Devices using Tiger192 and Whirlpool”. International Journal of Advanced Computer Science and Applications –ASTESJ. DOI:10.14569/ijacsa.2021.0120853

International Conference Publications

1. Asare B.T., Quist-Aphetsi K., Nana L. (2019), "Using RC4 and Whirlpool for the Encryption and Validation of Data in IoT," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 114-117, doi: 10.1109/ICSIoT47925.2019.00027

2. Asare, B.T., Quist-Aphetsi, K., & Nana, L. (2019). “Nodal Authentication of IoT Data Using Blockchain”. 2019 International Conference on Computing, Computational Modelling and Applications (ICCMA), 125-1254, doi: 10.1109/ICCMA.2019.00028.

3. Asare B.T., Quist-Aphetsi K., Nana L. (2019), "Secure Data Exchange Between Nodes in IoT Using TwoFish and DHE," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 101-104, doi: 10.1109/ICSIoT47925.2019.00024

4. Quist-Aphetsi K, Asare B. T., and Nana L. (2019), "Centralised Nodal Block Ledger for Remote Edge Devices in IoT," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 19-23, doi: 10.1109/ICSIoT47925.2019.00010

5. Quist-Aphetsi K, Asare B. T., and Nana L. (2019), "IoT Node-Node Secure Communication Using RIPEMD-128 and DES," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 62-65, doi: 10.1109/ICSIoT47925.2019.00018

6. Asare B.T., Quist-Aphetsi K., Nana L. (2020) “Towards a Secure Communication of Data in IoT Networks: A Technical Research Report”. In: Qiu M. (eds) Algorithms and Architectures for Parallel Processing. ICA3PP 2020. Lecture Notes in Computer Science, vol 12454. Springer, Cham. https://doi.org/10.1007/978-3-030-60248-2_39

7. Asare, B. T., Quist-Aphetsi, K., & Nana, L. (2019). “A Hybrid Lightweight Cryptographic Scheme For Securing Node Data Based On The Feistel Cipher And MD5 Hash Algorithm In A Local IoT Network”. In 2019 International Conference on Mechatronics, Remote Sensing, Information Systems, and Industrial Information Technologies (ICMRSISIIT) (Vol. 1, pp. 1-5). IEEE. 10.1109/ICMRSISIIT46373.2020.9405869

8. Asare, B.T., Quist-Aphetsi, K., & Nana, L. (2019). "Secure Data Exchange Between Nodes in IoT Using TwoFish and DHE," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), 2019, pp. 101-104, doi: 10.1109/ICSIoT47925.2019.00024.

9. B. T. Asare, K. Quist-Aphetsi, L. Nana and G. Simpson, "A nodal Authentication IoT Data Model for Heterogeneous Connected Sensor Nodes Within a Blockchain Network," 2021 International Conference on Cyber Security and Internet of Things (ICSIoT), 2021, pp. 65-71, doi: 10.1109/ICSIoT55070.2021.00021.

DOCTORAT BRETAGNE LOIRE / MATHSTIC

Université de Bretagne Occidentale

GCTU

Titre : Une approche cryptographique pour l'authentification de données multimédia dans l'Internet des Objets basée sur les blockchains.

Mots clés : Internet des Objets (IoT), Blockchain, Cryptographie à clé publique, Authentification, Modélisation, Réseaux de Petri.

Résumé : Les cyberattaques hautement sophistiquées et les occurrences de failles de sécurité sur l'IoT (Internet des Objets) croissent de jour en jour. La technologie des blockchains de par ses atouts, a suscité un grand intérêt de la part des industriels face aux cyberattaques visant ces systèmes. Contrairement au modèle classique d'identité basé sur un système central dédié qui a l'inconvénient d'être un point singulier de défaillance, elle offre un mécanisme d'authentification basé sur une technologie de registres distribués évitant d'avoir un point singulier de défaillance. Malgré les atouts des blockchains, les solutions existantes ne prennent pas suffisamment en compte certaines spécificités de l'IoT telles que les contraintes en termes de capacités de stockage et de calculs. Par ailleurs, les primitives cryptographiques utilisées n'offrent pas le niveau de sécurité nécessaire. Dans cette thèse, des solutions ont été proposées pour répondre à ces problèmes. Ses principales contributions peuvent se décliner en 3 volets principaux: la proposition d'une architecture basée sur la blockchain pour la sécurité de l'IoT, la proposition d'approches cryptographiques pour le renforcement de la sécurité dans les architectures basées sur la blockchain pour l'IoT et, l'élaboration d'un modèle basé sur une approche formelle, à savoir les réseaux de Petri, pour la simulation, l'analyse et la vérification d'architectures basées sur la blockchain pour l'IoT.

Title : A cryptographic technique for authentication of multimedia data in internet-of-things using blockchain.

Keywords : Internet of Things, Blockchain, Public-Key Cryptographic, Authentication, Modeling, Petri nets.

Abstract : High profile cybersecurity attacks on networks and the occurrences of critical security breaches on existing wireless sensor networks grow by the day. The distributed ledger technology provided for by blockchain with all its benefits has shown great industrial prospects in the face of the rampant occurrences of cyber thefts of critical data from compromised networks. The centralized identity model relies on centralized authentication mechanisms that create a single point of failure for such database systems. Again, the centralized data is a honeypot for cyber-attacks. Blockchain as a database structure adopts an authentication mechanism that depends on the distributed ledger technology to eliminate the incidences of single point of failure. Although Blockchain holds security benefits for ensuring enhanced privacy and confidentiality for stored data, the computational overhead in classical blockchain technology makes it a challenge for full use in a network where its devices are resource constrained. In this thesis, solutions have been proposed to solve the aforementioned problems. The main contributions of the PhD thesis are the following: the proposal of a blockchain-based architecture for IoT security, the proposal of cryptographic approaches for security reinforcement in blockchain-based architectures, and the elaboration of a model based on a formal approach, namely Petri nets, for simulation, analysis and verification of blockchain-based architectures for IoT.