KARADENİZ TECHNICAL UNIVERSITY
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

DISTRIBUTED CACHING FOR WIRELESS IOT

MASTER THESIS

Mustafa NAJMULDEEN

AUGUST 2020
TRABZON

KARADENİZ TECHNICAL UNIVERSITY
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

DISTRIBUTED CACHING FOR WIRELESS IOT

Mustafa NAJMULDEEN
ORCID : 0000-0002-1649-0337

This thesis is accepted to give the degree of
"MASTER OF SCIENCE"
By
The Graduate School of Natural and Applied Sciences at
Karadeniz Technical University

The Date of Submission	: 03 / 07 /2020
The Date of Examination	: 21 / 08 /2020

Supervisor	: Asst. Prof. Dr. Sedat GÖRMÜŞ
ORCID	: 0000 - 0003 - 2172 - 2144

Trabzon 2020

ACKNOWLEDGEMENT

This thesis was submitted to the Karadeniz Technical University, Institute of Science, as the final part of my computer engineering master's program and it was carried out between September 2017 and December 2019. Moreover, it accounts for 30 ECTs towards the completion of my MSc. in Engineering.

In view of this thesis, an uncommon dispersed cache solution was proposed to reduce the IoT traffic impact on the Internet backbone. In this solution, the generated data is processed and stored within the IoT network securely via a distributed storage solution. This solution utilizes the properties of Maximum Distance Separable codes (MDS) to create an agreement between the network bandwidth and storage area. This solution will enable storing the sensor data at the network edge trying to tackle the scalability challenges imposed on the Internet by billions of IoT devices. In this study, low power wireless device networks are partitioned into disjoint cliques with minimum overlap to reliably store the generated sensor data within these disjoint cliques in a distributed manner. Furthermore, the proposed wireless storage solution creates optimal clique coverage to boost the accuracies of the caching system.

Above all else, I might want to communicate my sincere thanks toward my regarded thesis advisor, Asst. Prof. Dr. Sedat GÖRMÜŞ, for insightful conversations and observations, as well as constructive feedback and recommendations during the process of this thesis. This thesis would never have come into being without his help.

I might want to give my most profound gratitude to my family for all the sacrifices they have made for me and their endless support. I might likewise want to thank my friends which, throughout my studies, they provide me so much support, faith and confidence.

Thank you very much
Mustafa NAJMULDEEN

Trabzon 2020

DECLARATION

This is to proclaim that the research work in this proposition entitled “Distributed Caching for Wireless IoT" is a record of a credible examination work done by me, Mustafa NAJMULDEEN. Neither this thesis nor any part of it has been acknowledged to any other Institute or University for the honor of any degree or certificate, except where due reference acknowledgement has been given in the text. This thesis is defended in 21/08/2020.

Mustafa NAJMULDEEN

CONTENT

Master Thesis

SUMMARY

Distributed Caching for Wireless IoT

Mustafa NAJMULDEEN

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Computer Engineering Graduate Program
Supervisor: Asst. Prof. Dr. Sedat GÖRMÜŞ
2020, 37 Pages

The future Internet will be connecting billions of low power embedded devices. This globally connected network of small embedded devices is thought to be generating most of the traffic carried over the future Internet backbone. This new Internet containing billions of small embedded devices are the Internet of Things (IoT). Furthermore, significant portion of the information data generated by the IoT networks will need to be stored in data centers contributing to the scalability issues already affecting the Internet. New and novel solutions need to be created to prevent this data explosion's impact on the Internet infrastructure.

In view of this thesis, an uncommon dispersed cache solution was proposed to reduce the impact of the IoT traffic on the Internet backbone. In this solution, the generated data is processed and stored within the IoT network securely via a distributed storage solution. This solution utilizes the properties of Maximum Distance Separable codes (MDS) to create an agreement between the network bandwidth and storage area. This solution will enable storing the sensor data at the network edge trying to tackle the scalability challenges imposed on the Internet by billions of IoT devices. In this study, low power wireless device networks are partitioned into disjoint cliques with minimum overlap to reliably store the generated sensor data within these disjoint cliques in a distributed manner. Furthermore, the proposed wireless storage solution creates optimal clique coverage to boost the accuracies of the caching system.

Key Words: Internet of Things, MDS Codes, Distributed Storage, Cliques and RPL

Yüksek Lisans Tezi

ÖZET

Kablosuz IoT için Dağıtılmış Önbellekleme

Mustafa NAJMULDEEN

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Sedat GÖRMÜŞ
2020, 37 Sayfa

Gelecekte Milyonlarca düşük enerjili gömülü aygıtlar internet ağına bağlanacaktır. Yeni global olan bu ağ ve buna bağlanan çok sayıdaki küçük aygıtlar sınırsız olarak veri oluşturacağı düşünülmektedir. Söz konusu olan bu yeni bağlantı şebekesine Nesneler İnterneti (IoT) adı verilmiştir. Nitekim Nesneler İnterneti adını taşıyan bu yeni ağlardan üretilen bu sınırsız verileri, veri merkezlerinde saklanması gerekecektir. Ancak yaşanacak olan veri patlaması internetin trafiğine olumsuz etkileyerek yoğunluğa sebep olacaktır. Söz konusu olan bu olumsuz etkinin yol açacağı problemleri ortadan kaldırılmasına yönelik çözümler üretilmesi gerekir.

Bu çalışmada, IoT'nin yol açtığı yoğun trafiğinin internet üzerindeki olan olumsuz etkinin giderilmesi için yeni dağıtılmış bir depolama çözümü önerilmiştir. Bu çözümde, üretilen veriler IoT ağı içinde dağıtılmış depolama yöntem ile güvenli bir şekilde saklanır ve işlenecektir. Nitekim söz konusu olan bu çözümün asıl dayandığı temel, depolama alanı ile iletişim bant genişliği arasında denge oluşturmak suretiyle maksimum mesafe ayrılabilir özelliklerine sahip kodlardır. Çalışmada önerilen bu çözüm, milyarlarca IoT aygıtı tarafından internete getirilen ölçeklenebilirlik zorluklarının üstesinden gelmeye çalışırken, sensör verilerinin ağ kenarında depolanmasını sağlayacaktır. Bu çalışmada, düşük güçlü kablosuz aygıt ağları, üretilen sensör verilerini bu ayrık kliklerin içinde dağıtılmış bir şekilde güvenilir olarak saklamak için minimum örtüşme ile ayrık kliklere bölünmüştür. Ayrıca, önerilen kablosuz depolama çözümü, depolama sisteminin güvenilirliğini artırmak için en uygun klik kapsama alanını bu çalışmada oluşturulmuştur.

Anahtar Kelimeler: Nesnelerin İnterneti, MDS Kodları, Dağıtılmış Depolama, klikler ve RPL

LIST OF FIGURES

<u>Page Numbers</u>

LIST OF TABLES

ABBREVIATIONS

IoT	Internet of Things
RFID	Radio-Frequency Identification
OSI	Open Systems Interconnection
WSN	Wireless Sensor Network
ADC	Analog-to-Digital Converter
MAC	Media Access Control
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
W3C	World Wide Web Consortium
ETSI	European Telecommunications Standards Institute
CoAP	Constrained Application Layer Protocol
DDS	Data Distribution Service
MQTT	Message Queue Telemetry Transport
XMPP	Extensible Messaging and Presence Protocol
AMQP	Advanced Message Queuing Protocol
IPv6	Internet Protocol version 6
6LoWPAN	IPv6 over Low power Wireless Personal Area Network
TSCH	Time Slotted Channel Hopping
6TiSCH	IPv6 over the TSCH mode
RPL	Routing Protocol for Low-Power and Lossy Networks
CORPL	Cognitive Routing Protocol for Low-Power and Lossy Networks
CARP	Channel-Aware Routing Protocol
EPCglobal	Electronic Product Code global
LTE-A	Long-Term Evolution-Advanced
HTTP	HyperText Transfer Protocol
REST	Representational State Transfer
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
CON	Confirmable
NON	Non-Confirmable

RoLL	Routing over Low-power and Lossy networks
LLN	Low-power and Lossy Networks
DODAG	Destination-Oriented Directed Acyclic Graphs
LBR	Low-power and Lossy Networks Border Router
DIO	DODAG Information Object (DIS)
DIS	DODAG Information Solicitation
DAO	Destination Advertisement Object
ACK	Acknowledgment
MOP	Modes of Operation
LR-WPAN	Low Rate Wireless Private Area Networks
PHY	Physical
DSSS	Direct Sequence Spread Spectrum
CSMA/CA	Carrier-Sense Multiple Access / Collision Avoidance
FFD	Full Function Device
RFD	Reduced Function Device
PAN	Personal Area Network
DSS	Distributed Storage System
GFS	Google File System
MOE	Multi-objective Evolutionary
MORM	Multi-objective Optimized Replication Management
MDS	Maximum Distance Separable
DC	Data Collector
CH	Clique Head
DEC	Decentralized Erasure Codes
FSPL	Free Space Path Loss
SNR	Signal to Noise Ratio
gcd	Greatest Common Divisor
GF	Galois Field
BF	Brute-Force

1. INTRODUCTION

1.1. Internet of Things

Nowadays, Internet of Things (IoT) networks plays an essential role in human life. Over the most recent few years, the Internet of Things term has turned out to be a hot research subject in network fields. IoT networks consists of a globally connected network where sending and receiving data over the Internet is possible without needing any person to person or person to computer interaction [1]. There are two terms for the Internet of Things. The "Internet" term refers to a network oriented manner, while "Things" term refers to the object oriented manner.

The first IoT machine was made in the mid-1980s 1980s. Several programmers created this device at Carnegie Mellon University [2]. The device used to report the machine's contents through the network [3]. The first explanation of the IoT is credited to The Auto-ID Labs [4], which is a massive network of research centers that deal with networked RFID and sensing automation. The Auto-ID Labs in 1999 mentioned, for the first time, the IoT term. Their concept is about the object-oriented approach, and that object is a simple Radio-Frequency Identification (RFID) tag.

The main components of the IoT systems are the sensors, communication interfaces, data analyzing algorithms and the user interface respectively [5], [6]. Sensor device in the IoT networks gather the data from the environment that surround the sensors. Examples of the collected data can be room temperature or live video. In IoT networks, the aggregated data from the sensors are carried to the servers over a backbone communication infrastructure. Different communications mediums are used to deliver the aggregated data to the central network, generally placed in the Internet Cloud. Choosing the best medium, taking the trade-off between power consumption and bandwidth into account, is a momentous decision for such IoT network. The aggregated data are analyzed to define and reduce meaningful information from it, which may be transformed and displayed in a user interface (UI) so the client can cooperate with it. At times, the data from sensors are analyzed locally instead of analyzing it in the server, which may be necessary for scenarios with limited backhaul connectivity.

Many applications may take advantage of IoT networks. Establishing a connection between devices, observing the environment for a superior personal satisfaction, and saving time and money by automating several aspects of both industry and commerce are some of these advantages.

Shortly, the IoT networks are estimated to include billions of network nodes all around the world. This exponential increase in the deployed IoT devices will be responsible for a significant portion of the future Internet traffic. Therefore, the large amount of data from these deployments need to be accommodated with minimal impact on the future Internet infrastructure. Complexity, privacy, and safety are additionally the challenges that should be addressed for making the IoT networks a reality.

1.1.1. Internet of Things generic structure

There is a generic architecture for the IoT networks. This generic architecture is classified into four layers: application layer, middleware layer, network layer and device layer respectively. IoT layers architecture are formed in a manner that can match the need for different application areas [7]. Figure 1 shows the generic architecture for the IoT network. The four layers are described below:

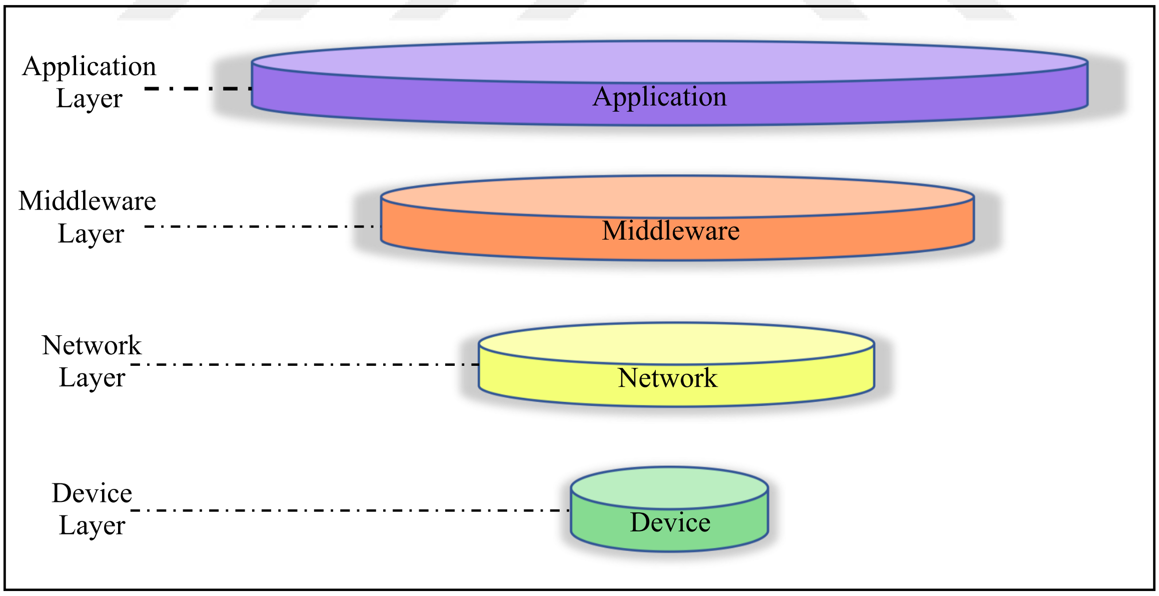


Figure 1. Generic IoT networks stack structure

Device layer: This layer is the principal level in generic IoT structure, representing the OSI model's physical layer [8]. There are various sorts of sensor nodes, such as Zigbee and RFID Infrared. Device layer is responsible for gathering the data (i.e., information) for the sensor nodes. This information can be humidity, wind speed, temperature and dust percentage in the air. The information of this layer is moved upward to the second layer.

Network layer: This layer takes a vital responsibility in the IoT network architecture [8]. It is in charge of deciding the route the data information to the upper layer [9].

Middleware layer: This layer is lays down between second layer and last layer. It plays as an interface between the network and application layers. It runs in bidirectional mode [9]. The middleware layer has many functions; the main two functions are service management and storing the collected information from the device layer into a database [8]. This layer is capable to recover and compute the data information.

Application layer: This layer is the last layer of the IoT structure, and its task is act as a user-interface between the client and the IoT network.

1.1.2. Internet of Thongs Application

The IoT networks an enormous number of utilizations in different scenarios. These enormous number of utilizations can be used in the public and the private sectors. These applications can be ranged from consumer IoT to industrial IoT networks. IoT applications can be classified into four main domains. These domains are transportation domain, healthcare domain, smart domain and social and personal domain [1]. These four domains are described below:

Transportation domain: Modern vehicles like cars, buses, trains and also bicycles comes with many sensors and processing units. The sensors in modern vehicles are used to collect critical information about the vehicle itself or even collect information about the vehicles surrounding environment. Besides the vehicles, roads are also instrumented with sensors and tags. Sensing the pressure level of the tire and collision avoidance systems are simple examples of the transportation domain.

Healthcare domain: The IoT systems provide many benefits for this domain. The healthcare domain applications can be classified into automatically sensing and collecting vital data for patients, tracking of the patient's status, identification, and authentications. Patients surveillance is the most straightforward application if IoT technology.

Smart domain: This domain can be partitioned further into subdomains. These subdomains are the smart animal farming, smart agriculture, smart metering and smart environment [10]. In the smart domain, the IoT systems can provide many applications to protect the environment and make it better. For example, IoT solution can define alert zone by monitoring the burning gases, control the factories CO2 release, detect the earthquake early, monitor the level of water, oil, and gas in storage tanks, increase the production quality of the fruits and vegetables and also IoT solution can track the location of the animal.

Social and personal domain: The applications offered by the IoT networks for this domain can help a person to get in touch with other people for building social relationships [11]. The most straightforward application for the domain is locating lost objects and updating the social status information in the social networks.

1.1.3. Network of Wireless Sensors

IoT networks can only be formed by connecting objects to each other to retrieve information about the environment. Many types of devices are used to make this vision reality. Sensors play a vital role in many IoT applications. These sensors must have a communication interface to hand over the sensed information data to the network through a medium wirelessly. Wireless Sensor Networks (WSNs) provide the necessary interface for all the "Things" to be an IoT device. Wireless Sensor Networks might be described as a group of devices in the network that can deliver the data through wireless connections [12].

WSNs are made out of devices with information processing, data sensing and communicating components [13]. WSNs might have various sorts of sensors, for example, infrared, radar, seismic and thermal. These different types of these sensors can keep an eye on various kinds of environmental data like temperatures, pressure, noise levels and humidity [14]. WSNs it tends to be utilized in numerous application territories, for example, military, environment, health care and home area networks.

The collaborations between countless devices are the main idea behind the network of sensors. In WSNs, the essential restriction of sensor devices is the low power consumption necessity. In a dense deployment scenario, connecting the devices to a central sink node directly over a single hop can consume more power than the multi-hop communication. The modern WSNs radio layer is mostly dependent on the standard IEEE 802.15.4. Furthermore, this standard supports multi-hop networking to enable low power WSNs.

Many factors should be considered to design a sensor network. These factors that should be carefully planned before deployment are network topology, power consumption, working environment, fault tolerance and manufacturing cost [13]. Two of these elements are quickly depicted below:

Fault tolerance: It is the wireless network's ability for staying operational regardless of whether one or more sensor fails. The fault-tolerance requirement relies on the WSNs application, such as fault tolerance level for a sensor located inside a house is less than a sensor located on the battlefield.

Manufacturing cost: The expense of assembling sensing devices is an extremely challenging issue. WSNs utilize an enormous number of wireless sensor devices, the sensors deployed number could reach hundreds, thousands or even millions in an area. The cost of these devices should be under a certain threshold to make the sensor network cost-justified [13].

1.1.3.1. WSN Hardware Component

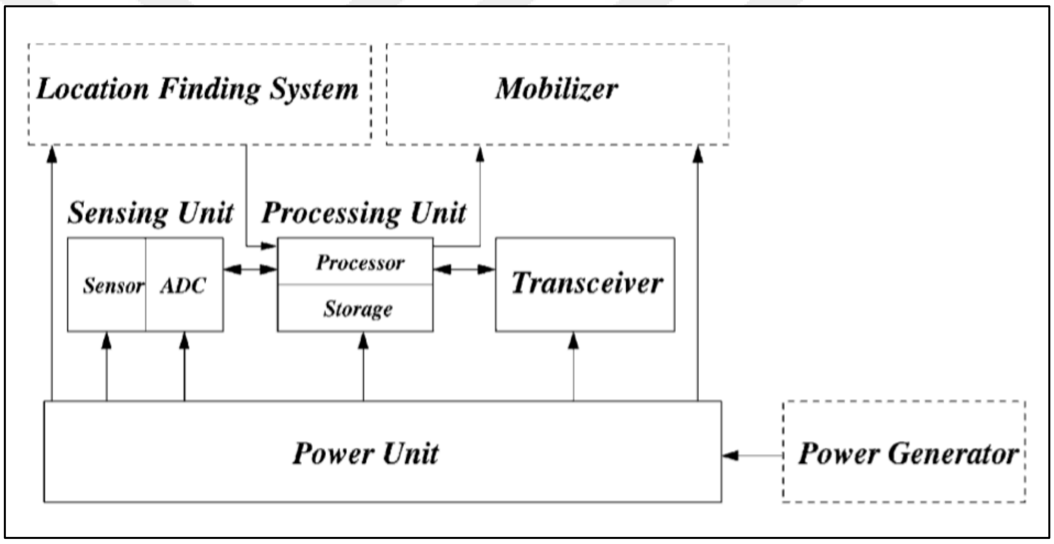


Figure 2. Wireless sensor component [13]

The component of a wireless sensor relies upon the application type. The primary wireless sensor contains four standard components. These are the sensing unit, processing unit, transceiver unit, and power unit. In a few applications, these four parts increase to six components. The other two units are location-finding systems and mobilizer units. Figure 2 shows component of a wireless sensor.

The first unit in the wireless sensor is the sensing unit. It is in general consist of two subunits: analog to digital converters (ADCs) and sensors. The signals that sensed form the sensors are analog signals and they have to be changed to digital signal by using (ADCs) to pass the signal to the processing unit. The sensors are interfaced with other parts of the sensor device via the processing unit. Inside this unit, a small cache space exists. Connecting the sensor device to the network and carrying the sensed information to the central network are done by the transceiver unit. An essential part of the wireless sensor is the power supply. In some applications, the location finding system unit is used to inform the sensor devices about the network routing and the sensing tasks. In different circumstances, the mobilizer unit is utilized to change area of a sensor node when it is expected to accomplish the fundamental tasks [15].

1.1.3.2. WSN Architecture

In Wireless Sensor Networks, the sensors devices can assemble the information from the environment and send the information back to main server (i.e., sink) by taking advantage of multi-hop communication architecture. Figure 3 shows the generic architecture for both the sink and the sensors.

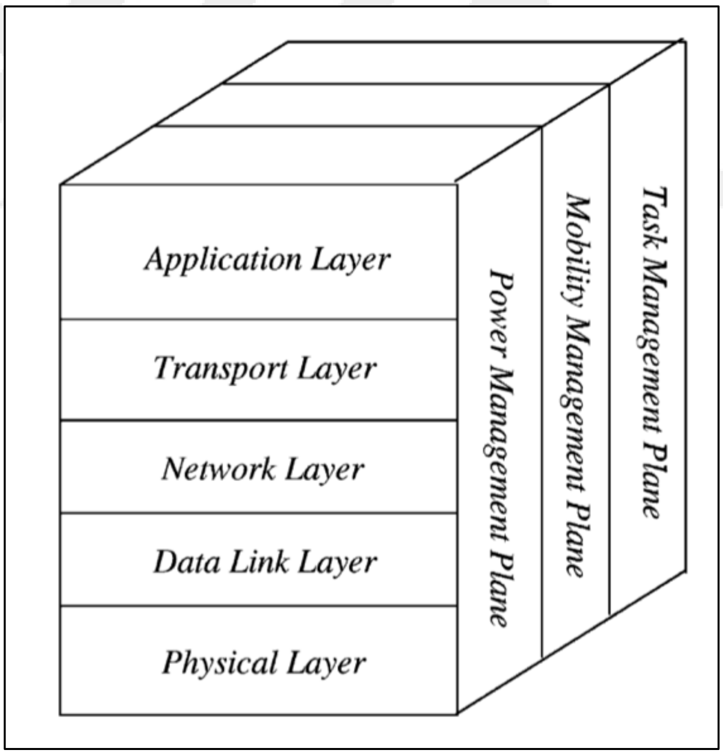


Figure 3. Architecture of WSNs [6]

This architecture defines how the data can be collected and transmitted through the communication medium and how the wireless sensor device should work to minimize its power consumption. WSNs architecture mostly act in accordance with the OSI model [16]. In wireless sensor networks, the architecture is mainly made up by five layers and they are:

the application layer, the transport layer, the network layer, the data link layer and the physical layer. Along with these layers, there are three cross layers of planes. These three cross layers are the task management plane, the mobility management plane and the power management plane [15]. The five layers of WSNs architecture are described below:

Physical layer: The first principal layer of the WSN architecture is liable for transmitting or receiving the sequence of zeros and ones across communication medium, frequency selection and data encryption are other responsibilities of the physical layer [16].

Datalink layer: the second layer of the WSN architecture is accountable for error control, data frame detection and medium access control (MAC). Inside this layer, MAC protocol is needed to reduce the collision with neighboring nodes since the wireless medium is broadcast by nature and the environment of the wireless sensor device is noisy [15].

Network layer: The essential role of the third layer is routing the frames along a selected route. In WSN, the network layer tries to find the reliable path that the data can be transferred through using a fixed scale called metric [16], [17].

Transport layer: This layer is required when the information should convey to different networks. The third layer is utilized to give both blockage avoidance and blockage detection and ensure the delivery of the data.

Application layer: the application layer carry out as an interface between user and sensors data [16]. In this layer, a different software application can produce for different sensor network applications.

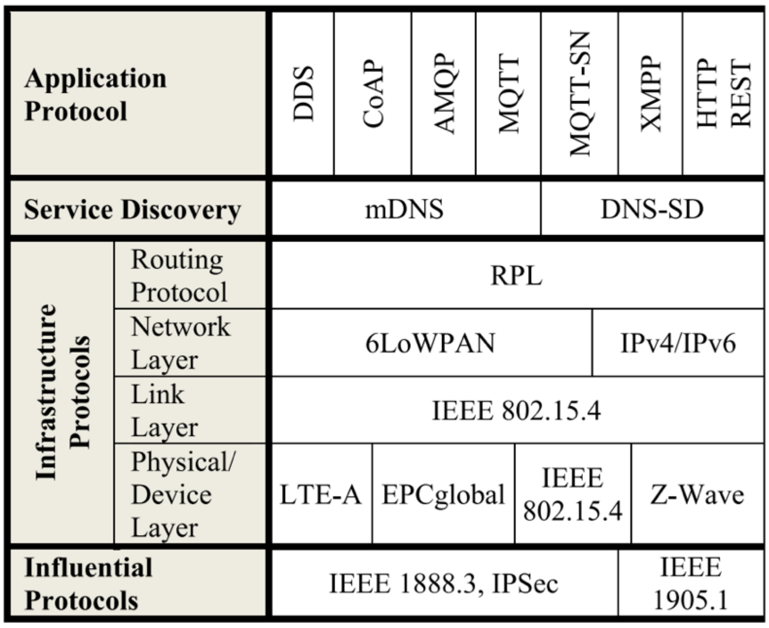
Task management plane, the mobility management plane and the power management planes: These planes are used to raise the efficiency and the network reliability by managing the protocol layers of the device and making them work with each other in collaboration to optimize the sensor performance the desired goal.

1.2. Internet of Thing Protocols

In general, the protocols are a group of rules used for routing and addressing the data packages [18]. In IoT networks, countless devices are relied upon to connect and understand each other without taking the environment or application of the devices into account. A standardized protocol is used to achieve communication between different devices to understand each other [19]. Different numbers of organizations are established to provide protocols for IoT networks. European Telecommunications Standards Institute (ETSI), Internet Engineering Task Force (IETF), Institute of Electrical and Electronics Engineers (IEEE) and World Wide Web Consortium (W3C) are examples for these organizations [20].

IoT systems can be classified into two protocols: application protocols and infrastructure protocols. The application protocols consist of Extensible Messaging and Presence Protocol (XMPP), HTTP RESET, Constrained Application Layer Protocol (CoAP), Data Distribution Service (DDS), Advanced Message Queuing Protocol (AMQP) and Message Queue Telemetry Transport (MQTT) [20].

The infrastructure protocols are the network protocols and device layer protocols. These network protocols can be separated into two subdivisions, and they are encapsulation and routing protocols [21]. IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) and IPv6 over Low power Wireless Personal Area Network (6LoWPAN) protocols are encapsulation protocol [22]. While the routing protocols in the network section are Channel-Aware Routing Protocol (CARP), Cognitive Routing Protocol for Low-Power and Lossy Networks (CORPL) and Routing Protocol for Low-Power and Lossy Networks (RPL) [21]. On the other hand, the protocols that exist in the device layer are Electronic Product Code global (EPCglobal), IEEE 802.15.4 and Long-Term Evolution-Advanced (LTE-A).



Application Protocol		DDS	CoAP	AMQP	MQTT	MQTT-SN	XMPP	HTTP REST
Service Discovery		mDNS				DNS-SD		
Infrastructure Protocols	Routing Protocol	RPL						
	Network Layer	6LoWPAN				IPv4/IPv6		
	Link Layer	IEEE 802.15.4						
	Physical/ Device Layer	LTE-A	EPCglobal	IEEE 802.15.4	Z-Wave			
Influential Protocols		IEEE 1888.3, IPSec					IEEE 1905.1	

Figure 4. Internet of Things protocols [20]

Figure 4 reveals the IoT protocols [20]. Constrained Application Layer Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), IPv6 over Low power Wireless Personal Area Network (6LoWPAN), Routing Protocol for Low-Power and Lossy Networks (RPL) and standard of IEEE 802.15.4 protocol are detailed in the upcoming sections.

1.2.1. Constrained Application Layer Protocol (CoAP)

The CoAP protocol is one of the IoT network's application protocols developed for web transfer by the Internet Engineering Task Force (IETF) [23]. CoAP protocol is utilized amongst a limited network like a constrained environment, such as WSN nodes. The CoAP protocol is designed to make a version of the HTTP protocol to meet the Internet of Things requirement for operating in the lossy and noisy environments and for supporting the low power consumption [24]. This IoT network protocol rely upon REpresentational State Transfer (REST) [20], and it supports basics of the HTTP protocol, such as to delete, put, post and get methods. The REST method is a procedure used to ensure a secure, fault-tolerant and scalable system. This protocol takes advantage of the User Datagram Protocol (UDP) to create a secure communication between endpoints [25]. The UDP is beneficial for this protocol because the UDP gives the ability to transmit the data to multiple endpoints while it maintains the communication speed and low bandwidth usage. Fig. 5 illustrates the functionality of the CoAP [20]

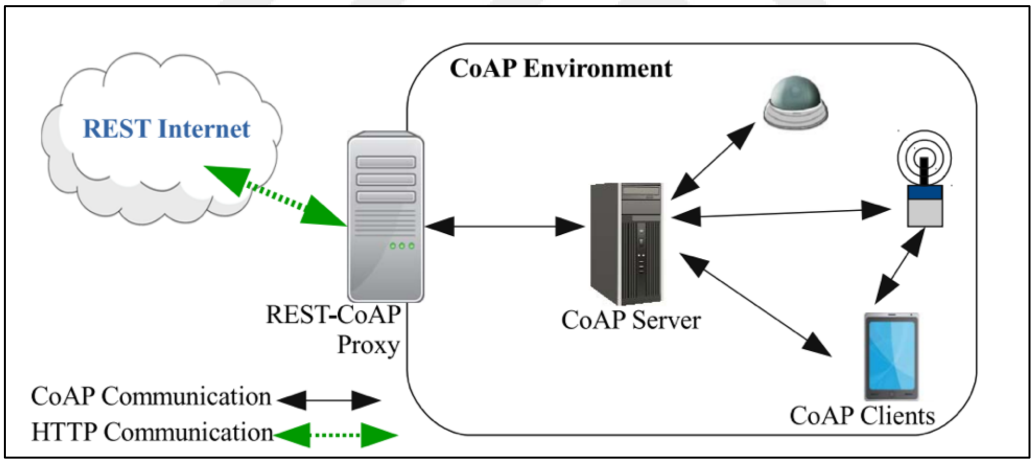


Figure 5. The functionality of the CoAP protocol [20]

The CoAP protocol can be separated into two split layers. These layers are the message and response/request layer. The message layer of the CoAP is in charge of controlling the message transmissions between two endpoints. While the response/request layer tracks the request and response codes to bypass issues like the arrival of the messages and lost or duplicated messages. Both layers are also used to stop-or-wait retransmissions, duplicate detection, and multicast support.

The message that transmits between two endpoints in the CoAP protocol can be four types. The four types are a piggy-backed response, confirmable (CON), non-confirmable

(NON) and separate response. CoAP protocol reliability is achieved by mixing both types of confirmable and non-confirmable. In confirmable type, the user sends the message to the cloud, and it waits the acknowledgment from the cloud. The non-confirmable type sends a message without waiting for the acknowledgment from the server. The piggy-backed response is when the user transmits the confirmable message to the cloud server, and the server sends the acknowledgment with the response to user at exact time. While in the separate response, the user transmits the confirmable type message to the server, and anyhow the server was not available to acknowledge immediately; the server sends an empty acknowledgment to the user. After some time, when the server is available to respond to the user, it sends the response with a confirmable message to the user, and the user should send back the acknowledgment. Each message in the CoAP has its ID and this is used to discover the duplicate messages. Figure 6 showing the CoAP message types.

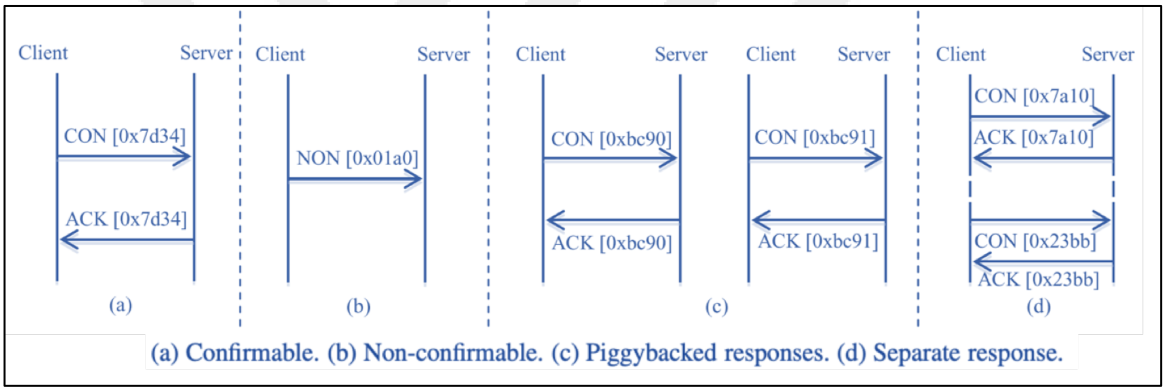
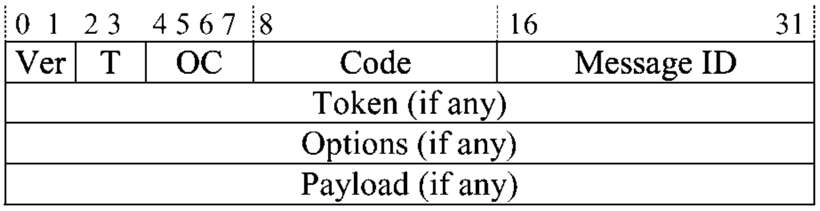


Figure 6. Types of the CoAP message [20]



0 1	2 3	4 5 6 7	8	16 31
Ver	T	OC	Code	Message ID
Token (if any)				
Options (if any)				
Payload (if any)				

Figure 7. CoAP message format [20]

Figure 7 illustrates the format of CoAP message [20]. The message size of the CoAP can be ranged from 10 bytes to 20 bytes [23]. In the format of CoAP message, "OC" is Option Count, "T" field is the type of the message and the "Ver" field is the version of the CoAP.

1.2.2. Message Queuing Telemetry Transport (MQTT)

The MQTT protocol is an Internet of Things application protocol that is made-up by Dr. Andy Stanford-Clark of IBM and Arlen Nipper of Arcom (Eurotech) in 1999, while in 2013 it was standardized at OASIS [26], [27]. This protocol is quite a simple type of messaging protocol. It is designed for restricted IoT devices and high latency or unstable networks. The preferred manner of this protocol is that it minimizes the bandwidth of the network, so the device resource requirements and the power consumption are low. Broker (server), the subscriber (user) and publisher (sensor) model are the base of the MQTT. The sensor's (publisher's) task is to bring together the data and transmit the data information to the user (subscriber) through the server (broker) [25]. The broker in the MQTT protocol is not only used for sending the data between publisher and subscriber, but it also ensures the security by examining the publisher and subscriber's authorization.

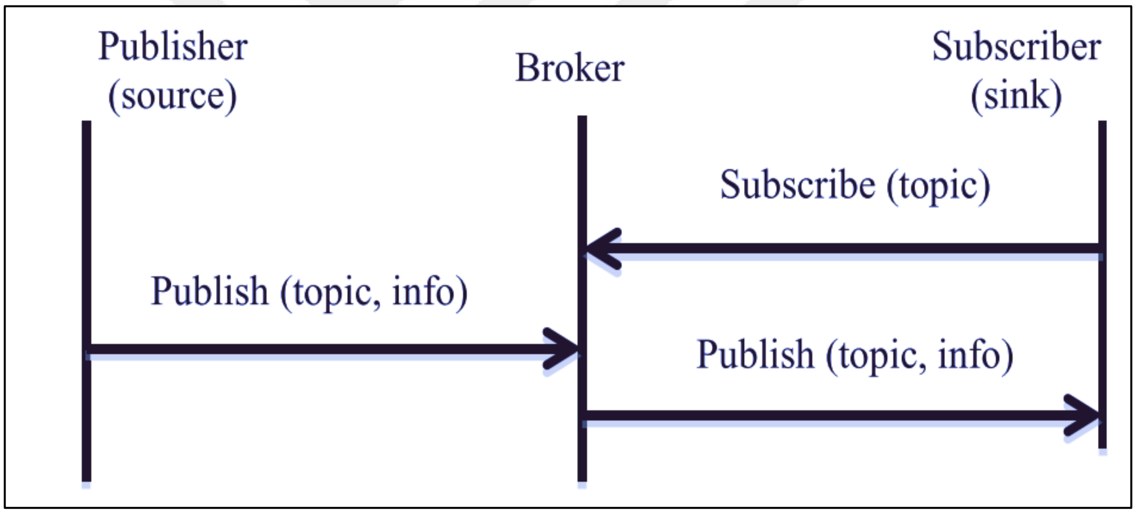


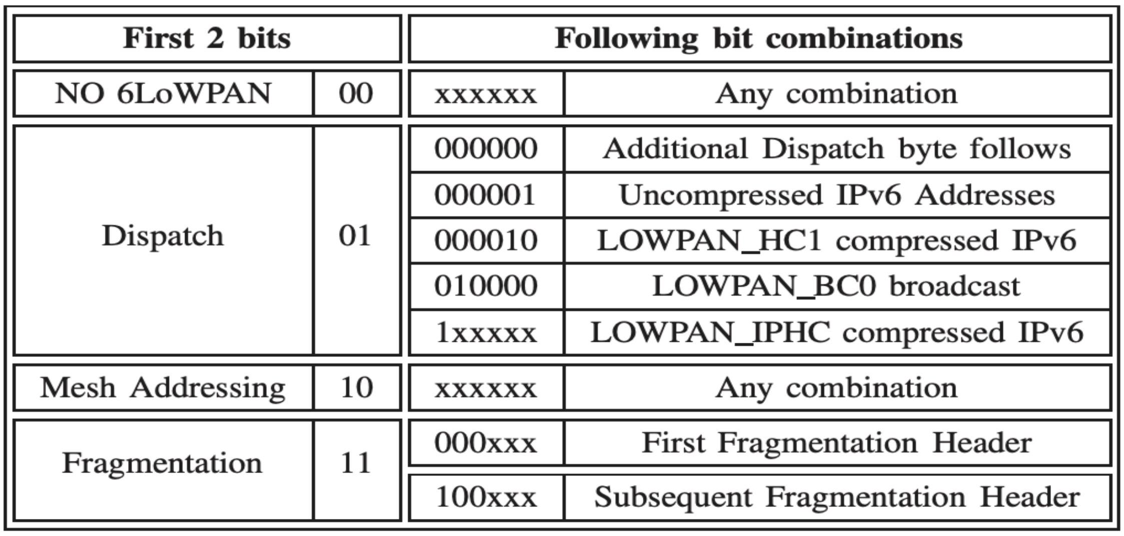
Figure 8. Publisher and subscribe process in the MQTT

The publisher in MQTT has the ability to define the quality of the message. The publisher can choose message quality by sending it with or without the confirmation and with or without taking the duplicated message into account. MQTT protocol uses the TCP protocol as a transport layer to deliver the message [28]. Figure 8 shows the publish and subscribe process in the MQTT [20].

1.2.3. IPv6 over Low power Wireless Personal Area Network (6LoWPAN)

The Low power Wireless Personal Area Networks (LoWPANs) are IoT networks which allow wireless connectivity for applications requiring low power consumption [29]. Low bandwidth, countless number of devices, low power consumption and low cost, Small packet size and supporting many topologies are the LoWPANs characteristics. The LoWPANs are based on the standard of IEEE 802.15.4 protocol, that it has 127 bytes of a frame size [20]. A countless number of wireless nodes in LoWPANs presents the need for ample address space, and IPv6 is the suitable candidate for this need [20]. The minimum fragmentable IPv6 packets size is 1280 bytes, and it is not compatible with the IEEE 802.15.4 protocol frame size [30]. An adjustment layer is needed to make the sizeable IPv6 frame fit within the IEEE 802.15.4 frame. This adjustment layer is called IPv6 over Low power Wireless Personal Area Network (6LoWPAN) standard [20].

In 2007, the IETF has developed the standard 6LoWPAN [20]. The fundamental utilization of the 6LoWPAN is encapsulating the IPv6 frame into IEEE 802.15.4 frames. This protocol is used to minimize the transmission overhead by maintaining the header compression, and it also maintain both multi-hop delivery and the fragmentation to match the small 127-byte packet in IEEE 802.15.4 protocol and multi-hop delivery.



First 2 bits		**Following bit combinations**	
NO 6LoWPAN	00	xxxxxx	Any combination
Dispatch	01	000000	Additional Dispatch byte follows
		000001	Uncompressed IPv6 Addresses
		000010	LOWPAN_HC1 compressed IPv6
		010000	LOWPAN_BC0 broadcast
		1xxxxx	LOWPAN_IPHC compressed IPv6
Mesh Addressing	10	xxxxxx	Any combination
Fragmentation	11	000xxx	First Fragmentation Header
		100xxx	Subsequent Fragmentation Header

Figure 9. Types of the 6LoWPAN header [20]

The 6LoWPAN's encapsulation method implements different dispatch codes. These dispatch codes are represented by two bits and they are (00) NO 6LoWPAN, (01) Dispatch, (10) Mesh Addressing and (11) Fragmentation. The header of (00) is used to discard the data packet if it is not a 6LoWPAN frame. In (01) Dispatch header, the header of IPv6 is compressed. The header of (10) allows the frames of the IEEE 802.15.4 protocol to be delivered to the link-layer. The last header, (11), utilized when the IPv6 frame was not capable to fit into the IEEE 802.15.4 standard frame. Figure 9 describes the 6LoWPAN header types.

In 6LoWPAN, the greater part of the IPv6 overhead is detached in such a manner that small frames of IPv6 can be transmited through a particular IEEE 802.15.4 frame [31], [20].

1.2.4. Routing Protocol for Low-Power and Lossy Networks (RPL)

The RPL protocol is a tree-based routing protocol which is commonly used in IoT networks for routing between network devices [32]. This protocol is established and standardized by the IETF Routing over Low-power and Lossy networks (RoLL) working group. The primary function of the RPL is to hold up the requirement of minimal routing in the Low-power and Lossy Networks (LLNs).

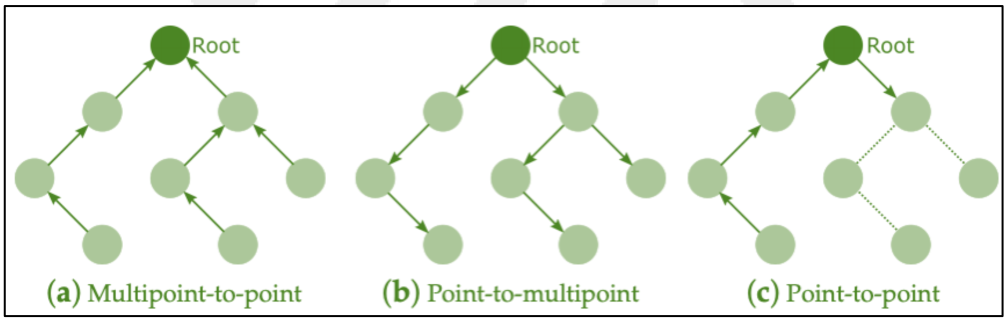


Figure 10. DODAG topology for all types of communication [32]

The RPL protocol supports all communication types, such as multipoint-to-point, point-to-multipoint, and point-to-point. This protocol arranges the network topology in a Destination-Oriented Directed Acyclic Graphs (DODAG). The DODAG graph is cited as a directed acyclic graph that has a particular LLN Border Router (LBR), root or sink node. Figure 10 shows the DODAG topology with all types of communication [32].

Routing metrics (link metrics) and node rank are used by the RPL to build up the DODAG. In the RPL protocol, there are three main variety of messages. These are: Destination Advertisement Object (DAO), DODAG Information Solicitation (DIS) and DODAG Information Object (DIO). These messages contain vital information data, like metrics and rank of the node.

The nodes initially create the DODAG in the RPL protocol. Thus, as the first thing, the DIO message is transmitted by the sink (i.e. root) device to its neighbors [20]. The node will receive the DIO message from the root node. Then it decides either it joins the DODAG or not according to the link metric. If the network device sorts out to join the DODAG, it computes its rank and rearranges the table of neighbor and selects the approved parent node which this parent will be utilized to send the message to the root node. Regularly the node's parent is the lowest rank node. In case a new node enters the network, it transmits a DIS data message to ask for the DIO data message from nodes that already part of the DODAG.

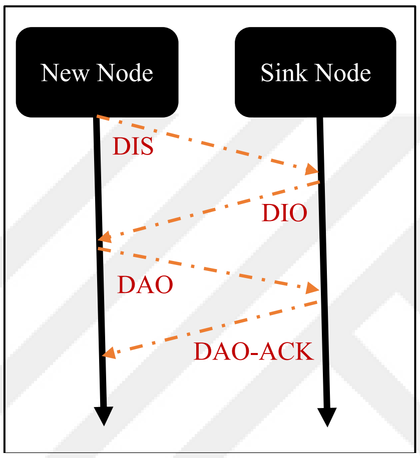


Figure 11. RPL messages [33]

After the node that that already part of the DODAG or the new node receives the DIO from its neighbors, it starts to transmit the DAO data message to request from the root or node's parent to join as a child node. If the root/parent accepts the node's DAO, it will respond with the DOA-ACK message. Figure 11 illustrating RPL messages [33].

In the RPL protocol, there are two types of modes of operation (MOP) that can be routing protocol (RPL) run on it [34]. These two types of modes are: non-storing mode and storing mode. In the storing mode, the node has the capability to keep a complete routing path for its sub DODAG nodes. When the MOP is storing mode, the node will forward the received message to the target device, if the target node route is accessible in its routing table. If the responsible node does not find the target node in its routing table, the message will be forwarded to its parent.

In non-storing mode, the sink (i.e. root) node stores the information of the topology for the entire DODAG. Root node makes use of DAO messages to collect this information. It calculates the path to the target node according to the collected data, and it places this path of the routing inside the message header and then forwards it to the next hop. Each node that located in the routing path will transmit the data message to the next hop according to the routing path in the header until it is delivered.

Right now, The RPL protocol does not bolster a mixed-mode where some LLN device is working in non-storing mode, and the other is working in storing mode. If all the LLN devices are working in a non-storing mode in the network, they do not need to keep and know the routing path about the DODAG. So, each message needs to be sent up to the root to obtain the source path and then sent the data message down to the target node. Figure 12 shows the non-storing mode and storing mode for the routing protocol [34].

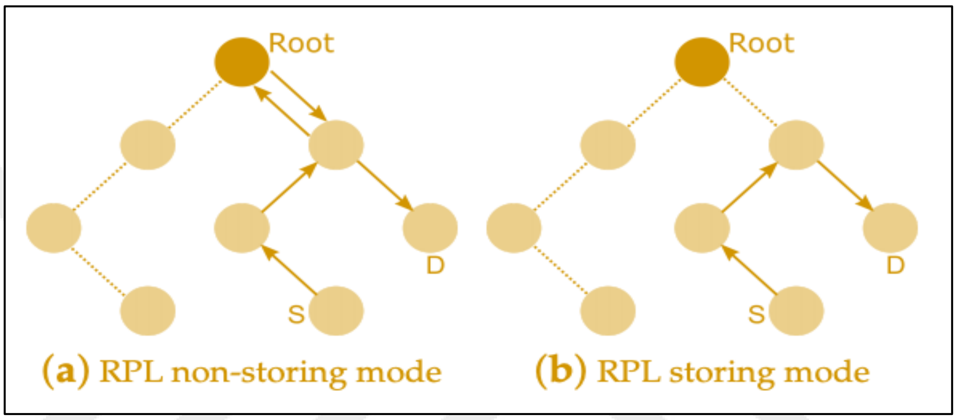


Figure 12. Non-storing mode and storing mode for the routing protocol [34]

1.2.5. IEEE 802.15.4

The standard of IEEE 802.15.4 protocol is produced for the Low Rate Wireless Private Area Networks (LR-WPAN) to specify the physical layer (PHY) and the Medium Access Control (MAC) [35]. This protocol is the most frequently used protocol in IoT networks because of its particulars, such as low cost, low power consumption, high message throughput and low data rate.

IEEE 802.15.4 protocol supports different channel bands for the physical layer. It utilizes a direct sequence spread spectrum (DSSS) approach in the PHY layer. According to these different frequency channel bands, the physical layer can receive and transmit the data at different rates: in 2.4 GHz frequency, the node can transfer data using 250 kbps data rate, in 915 MHz frequency, the data rate is 40-kbps data rate. In 868 MHz frequency, a 20-kbps data rate is typical [20]. Based on these different data rates and frequencies, the larger bands and higher frequencies can provide higher throughput and low latency while lower frequencies can produce the best sensitivity and it can cover broad areas. The IEEE 802.15.4

protocol uses the Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) approach in the MAC layer to reduce collisions in the network.

The IEEE 802.15.4 protocol bolsters two different categories of network devices (i.e., node). These devices are Reduced Function Device (RFD) and Full Function Device (FFD) [34]. The FFD device has the ability to serve as the coordinator of Personal Area Network (PAN). Moreover, RFD is a network device that does not have the capability of serving like FFD.

The RFD device is created for uncomplicated applications such as a passive infrared sensor or a light switch [34]. RFD sends limited amounts of information data, and it only collaborates with a single FFD at a time. Fig. 13 shows the topologies of IEEE 802.15.4 [34].

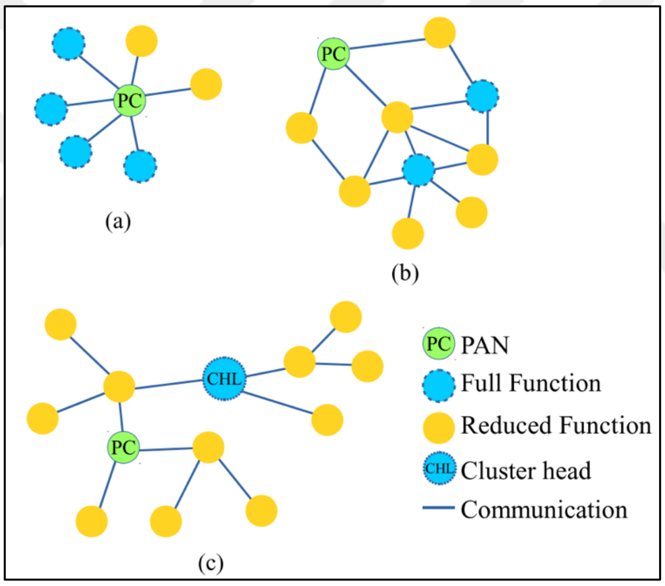


Figure 13. Topologies of IEEE 802.15.4 [20]

1.3. Distributed Storage System

IoT network system is predicted to penetrate the Internet and increase the connected small-scale gadgets to billions by the introduction of a new application. Many of these IoT devices will be responsible for generating information such as patient's vital data. The information generated by IoT devices (i.e., devices deployed in harsh environments) may be lost before being sent to the central server. This risk can be reduced by utilizing a Distributed Storage System (DSS) in IoT networks [36].

The DSS is utilized in IoT Networks to store data belonging to a node in neighboring nodes. This distribution method can be achieved by copying the node's data into different node devices that located in the network.

The DSS also has ability to repair information data when an erasure happens. When the node leaves the network or collapses, a new device node will be imported to network as a newcomer and gets the data from the nodes that survive to regenerate the correct data information and restore the failed device [37].

The uncomplicated approach in distributed storage is replication. Besides the replication method, network coding is also utilized in DSS.

1.3.1. Replication Distributed Storage System

In the last few years, data replication has become popular in many applications such as the sensor networking, peer-to-peer networks, World Wide Web and ad-hoc [38]. This method is used to generate various copies of the same data and store them at different sites. The primary explanation behind utilizing information replication is to guarantee the high accessibility of information and boost the scalability of the system [39]. In data replication, there are two procedures: passive replication and active procedures used to replicate the data [40]. Figure 14 shows these two procedures [39].

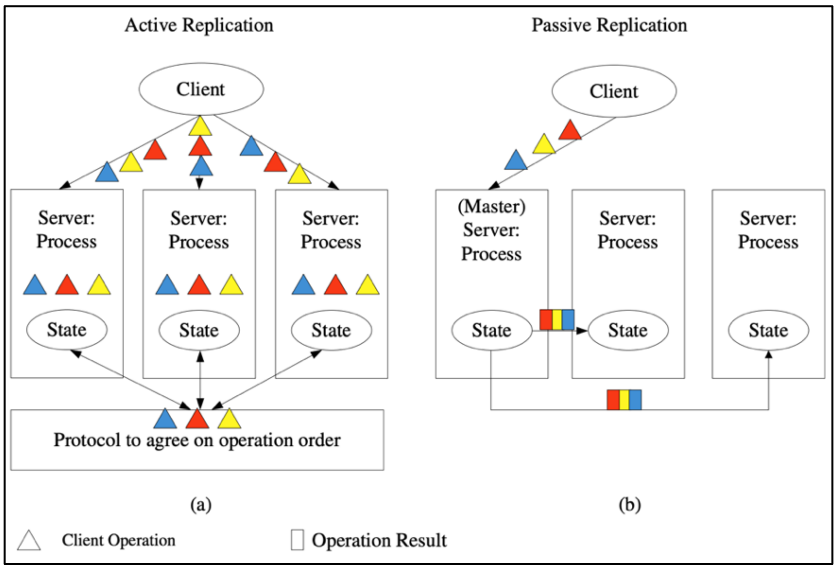


Figure 14. Active and passive replication techniques [39]

Along with these two procedures, the information replication can be apportioned into two basic systems: dynamic replication and static replication systems.

1.3.1.1. Passive and Active Replication Techniques

In the active replication technique, the operation requests are sent from the host to all replicas in an arranged order. The operations execution of all replicas agrees to be the same order by using some ordering protocol, such as Atomic Broadcast protocol. Then, every single replica runs the operations separately and finally, the result of these operations reaches a resulting state.

In the passive replication technique, the master replica is selected from the replication group. The host sends the operation execution to the master replica. The resulting state of the operation execution of the master replica will be sent to all other replicas that exist in the replication group. The bandwidth utilization is predicted to be much higher for this technique when the capacity of the resulting state is considerable.

1.3.1.2. Static and Dynamic Replication Systems

In the static replication mechanism, the host number and the copies (i.e., replicas) number are prearranged and well defined. This Static replication mechanism is easy to execute, however it is not frequently used because it is not flexible [41]. Google File System (GFS), MinCopysets, Multi objective Optimized Replication Management (MORM) and Multi-objective Evolutionary (MOE) are examples of static replication mechanism.

In the dynamic replication mechanism, the data replication will automatically generate and eliminate the replicas. The replica generation and elimination can be done according to the storage capacity, bandwidth and changes in the user entry pattern [41]. This mechanism makes its choices intelligently by taking advantage of the current environment information. There are some disadvantages, such as collecting the information of nodes in a complex infrastructure. Dynamic replication mechanism has some phases: creating and analyzing the connection between the system accessibility and the replicas number

1.3.2. Coded Distributed Storage System

In the distributed storage, the replication method is used for replicating the data to prevent data losses. Nevertheless, this replication is an unsophisticated method with low performance. The coded distributed storage system is introduced to enhance the system's execution and to boost the throughput of entire system with less storage space [37]. In the coded distributed storage system, instead of storing a data M on each node, the data M is encoded into m blocks, size of each one M/m, such that the data M be able to regenerated from all set of k blocks [42]. One of the well-known technologies of the coded distributed storage is the erasure codes such as maximum distance separable (MDS) [43], [44].

The erasure codes need less storage space if it is compared with simple replication in terms of storage. On the other hand, both erasure codes and the simple replication method requires the same repair bandwidth to repair the failed nodes [37].

In order to minimize the repair bandwidth used in the erasure code to regenerate the lost data, regenerating codes are proposed.

1.3.2.1. Regenerating Codes

In the MDS erasure codes, when a node fails in the system, the newcomer network device needs to collect entire data information files from all other surviving nodes to recreate the missing data. The repair bandwidth will be equal to the total data file size that collected by the newcomer node.

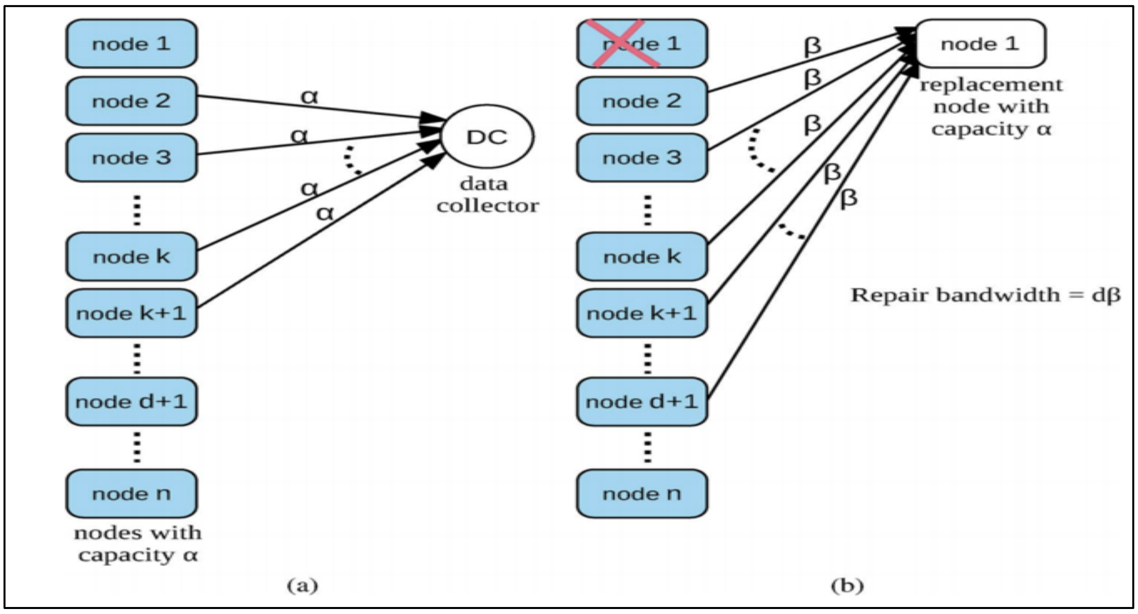


Figure 15. Structure of regenerating codes. a) Data reconstruction. b) Repair of a collapsed node [45]

Regenerating codes are proposed by Dimkis et al. to reduce this wasteful repair bandwidth [41]. The central concept of regenerating codes is the information flow graph. Figure 15 illustrates the setup of the regenerating code [45].

In the regenerating codes, all n nodes that exist in the network transmits their information to the data collector (DC). When any of these nodes fails, the newcomer node will collect the failed node's data β form survived node d to regenerate the lost data.

The information flow graph in the regenerating codes is utilized to evaluate the information from the S source data to the DC data collector. Single-source, data collectors and storage nodes are three categories of vertices that exists in the information flow graph. The source S stands for the node where the data initially generated.

1.3.2.2. Types of repair

In the distributed storage, the failed node's data information should get through the repair process to regenerate and store it in the newcomer node that will take the place of the failed one. There are many types of repair process used in the distributed storage system. These various types are exact repair of the systematic part, functional repair and exact repair [44]. In exact repair, the newcomer network device would produce the failed node's information without any change. In functional repair, the replacement node can generate the lost data in a functional way, in which the generated data may have different data as compared to the lost data. Exact repair of the systematic part is a combination type from functional and exact repair type.

1.3.2.3. Maximum Distance Separable Code

The code of Maximum Distance Separable (MDS) is one of the critical classes in error correcting codes that satisfy the Singleton bound. The MDS codes are established over a Finite field $\mathbb{F}$or Galois field GF [46]. The MDS codes are the conspicuous erasure codes used in network coding, and the famous code for the MDS is the Reed-Solomon codes [47]. The main job of the MDS code is to recover the lost information. So, to recover the lost information, the MDS codes will add some redundancy to the network. The MDS array codes are used to achieve this recovery. This array code is a two-dimensional array, and each column of this array represents the nodes inside the network. The array code will be responsible for deciding how the redundancy should be kept in the network. There are two types of array codes that can be used in MDS code. These types are horizontal array code and vertical array code [48]. In the vertical array code, both data and parity (i.e., redundancy) are kept in the same node n. While in the horizontal code, both data and parity kept in different nodes n. Figure 16 shows the type of array codes, where the nodes that located inside the network is referred as n, m and p are data of the node and parity (i.e., redundancy) respectively [47].

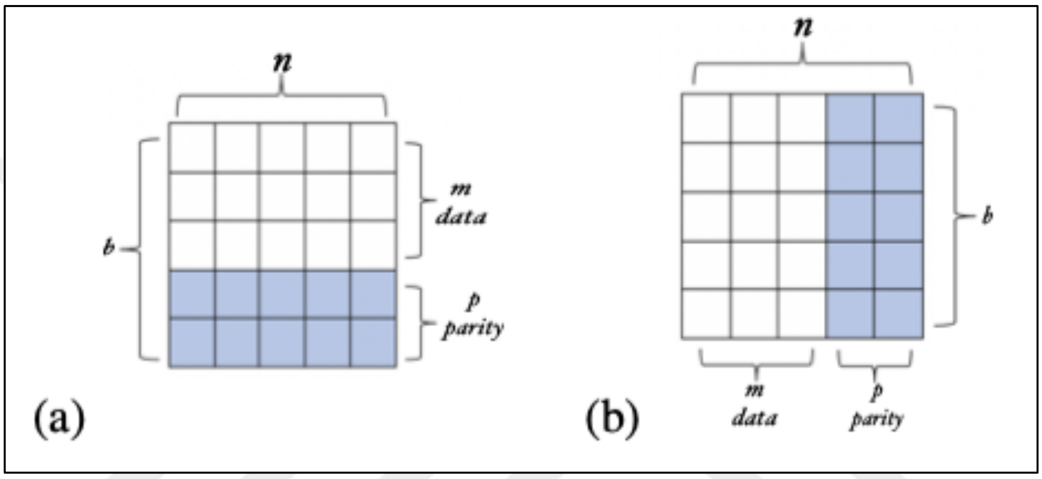


Figure 16. Type of the array codes: (a) Vertical codes and (b) Horizontal codes.

1.4. Network Partitioning

Day by day, the IoT application areas are increasing. This increase is responsible for defining a new node deployment scenario. In these new scenarios, there is a universal key challenge. The energy consumption of the nodes is the fundamental challenge. For instance, in these new scenarios, because of the node distribution that is in unreachable to individual humans, it is difficult to recharge the batteries or replace the node [49], [50].

The network partitioning is one of the assorted ways to reduce energy consumption on IoT devices [51]. The IoT network partitioning is done by partitioning the network into a group of IoT devices called clique/cluster. The node that has high energy inside this clique will act as a cluster head (CH) [52], [53].

CH are in charge of coordinating nodes within their groups, partitioning node's data within their groups, and communicating with each other or/and with external cliques on behalf of its nodes.

There are two main approaches to form the cluster. These approaches are leader-first and cluster-first. In the leader-first approach, the network devices that performs as cluster heads are selected. While in the cluster-first approach, clusters are sorted then the cluster header is selected [50]. Figure 17 shows the cluster architecture.

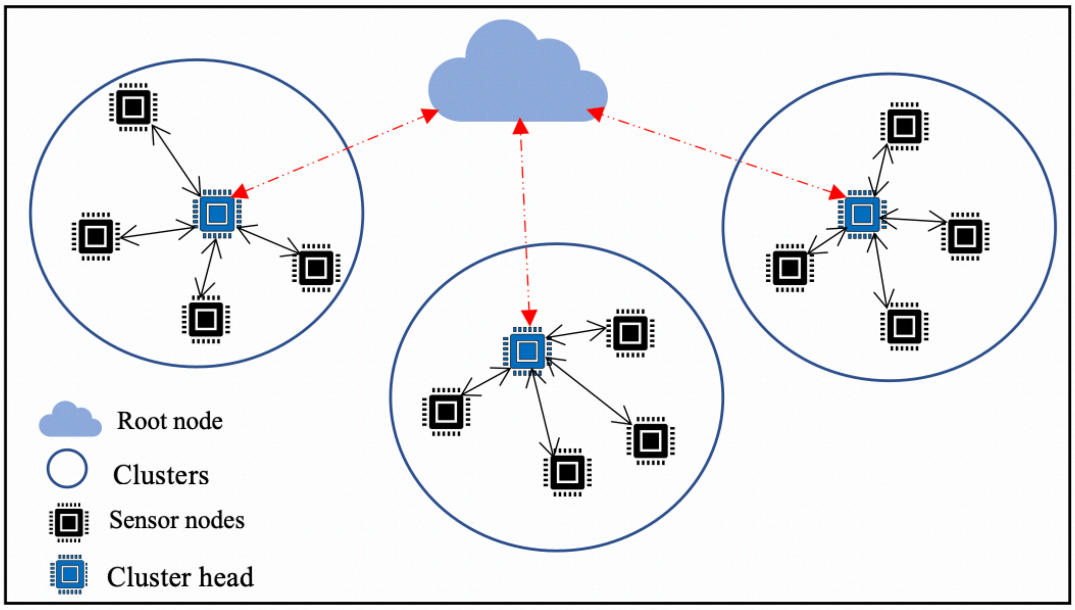


Figure 17. Cluster architecture

1.5. Thesis Motivation and Goal

With the launch of new network applications, the IoT notion has become a trending research topic.

It is imagined that billions of IoT low power devices will be deployed in the future, and these low power devices will be responsible for a large portion of future internet traffic. This considerable increase in the small data packet traffic is likely to put tremendous pressure on the Internet backbone. A solution of distributed storage is suggested in this work to reduce this effectiveness of the IoT network’s data traffic on the global Internet’s infrastructure.

One solution to reduce the effectiveness of IoT network’s data traffic on the global Internet’s infrastructure can be achieved by storing the information generated by the sensor devices within the wireless IoT network utilizing erasure codes that have Maximum Distance Separable (MDS) code features.

The wireless IoT networks are partitioned into sub-networks of disjoint cliques with a minimum number of overlapped nodes to achieve this distributed storage solution. The distributed storage solution improves the reliability of the data storage by selecting the clique configuration in a way that the outage of the links is minimized.

2. LITERATURE ANALYSIS

Nowadays, the IoT network has earned massive popularity as a research topic with the introduction of new Internet-enabled applications. There is enormous literature about IoT. Many studies have been done in a network partitioning, such as Seema et al. [54] proposed an algorithm to partition the network into sub-networks randomly to diminish the total energy consumption.

In another study, a weight-based network partitioning algorithm is proposed [55]. Storing the data and processing it in IoT have been mainly studied also recently by a large number of research studies [56], [57], [58] and [59]. Rodrigues et al. [60] suggested a hybrid design that can accurately renew the failed information block with fewer requirements such as bandwidth. They also compared, in the bandwidth-reliability space, the erasure codes with the normal data replication.

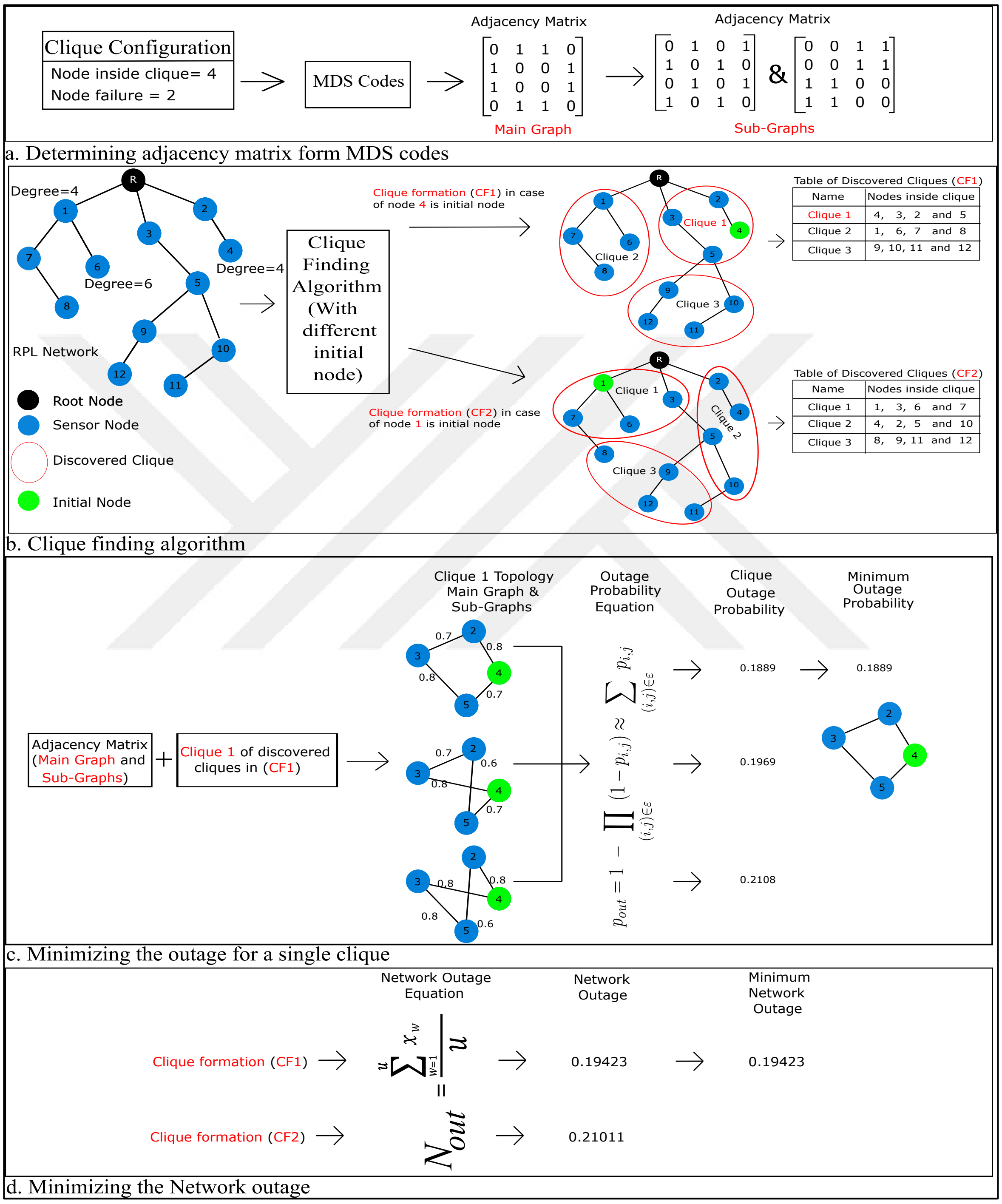
Another study is proposed to validate the efficient replacement of the storage for distributed information storage. Their proposal is in the partial network coding approach [61]. Jianjiang et al. [62] proposed a solution for data storage reliability. Their approach is based on the checksum of the XOR, and their model is about partitioning the primary data block into two sub blocks (i.e., parts) then run the XOR process on these sub blocks to produce different block (third part). Each of these parts would be reserved in different network devices to remake the primary information when the data is needed. A distributed storage system based on Decentralized Erasure Codes (DEC) is proposed in [56]. Their model is designed to increase the level of tolerated failure without losing the data in the network. They also proposed two more approaches to drop off the needed energy to design the information storage system to keep the data locally.

In another study, MDS erasure codes are utilized to improve reliability [63]. Magnus and his friend employed MDS codes in the Smart Meter networks to improve reliability. Their goal is to generate the MDS codes for any number of wireless network devices ($n \geq 4$) with any number of redundancy (i.e., erasure) ($r \geq 2$).

Gormus et al. [64] suggested an approach to lower the traffic of the network by distributing the data between nodes. Their design aims to partition the network into sub-network, each sub-network contains four nodes (i.e., four-node clique), with minimum overlapped node. The nodes of these sub-networks store redundant (i.e., parity) information for two of its neighboring nodes. Their network partitioning is done without considering the link characteristics.

3. PROPOSED MODEL AND RESULT OVERVIEW

A. Proposed Model



Clique Configuration
Node inside clique= 4
Node failure = 2
MDS Codes
Adjacency Matrix
Main Graph
Adjacency Matrix
Sub-Graphs
a. Determining adjacency matrix form MDS codes
Degree=4
Degree=6
Degree=4
RPL Network
Root Node
Sensor Node
Discovered Clique
Initial Node
Clique Finding Algorithm (With different initial node)
Clique formation (CF1) in case of node 4 is initial node
Clique formation (CF2) in case of node 1 is initial node
Clique 1
Clique 2
Clique 3
b. Clique finding algorithm
Clique 1 Topology Main Graph & Sub-Graphs
Outage Probability Equation
Clique Outage Probability
Minimum Outage Probability
Adjacency Matrix (Main Graph and Sub-Graphs)
Clique 1 of discovered cliques in (CF1)
0.1889
0.1969
0.2108
0.1889
c. Minimizing the outage for a single clique
Network Outage Equation
Network Outage
Minimum Network Outage
Clique formation (CF1)
Clique formation (CF2)
0.19423
0.21011
0.19423
d. Minimizing the Network outage

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \& \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Table of Discovered Cliques (CF1)

Name	Nodes inside clique
Clique 1	4, 3, 2 and 5
Clique 2	1, 6, 7 and 8
Clique 3	9, 10, 11 and 12

Table of Discovered Cliques (CF2)

Name	Nodes inside clique
Clique 1	1, 3, 6 and 7
Clique 2	4, 2, 5 and 10
Clique 3	8, 9, 11 and 12

$$p_{out} = 1 - \prod_{(i,j)\in\epsilon} (1 - p_{i,j}) \approx \sum_{(i,j)\in\epsilon} p_{i,j}$$

$$N_{out} = \frac{\sum_{w=1}^{n} x_w}{n}$$

Figure 18. The implementation of the system model [65]

The system model of this research is thought to be a query-based model where the sink node (i.e., root node) of the RPL network topology recovers the stored information from the distributed data on demand. This study provides a reliable high-level medium in which the data of the failed node can be regenerated from the accessible clique members.

- **Implementation Details**

The implementation of the proposition of this study can be visualized in figure 18 [65], and it can be classified into four separate parts. In the first step (figure 18 (a)), a suitable MDS code configuration is generated for the desired clique configuration of the distributed storage solution as given in [63]. In the second step (figure 18 (b)), the RPL network is partitioned to non-overlapping cliques utilizing the proposed clique finding algorithm. Here, the clique finding algorithm starts with different initial nodes to find a configuration that minimizes the network outage. In the third step (figure 18 (c)), outage probability of a single clique is minimized so that information sharing phase of the proposed solution consumes the least amount of energy. Finally (figure 18 (d)), the outage probability of the network is minimized by taking into account different clique configurations. Here, the clique configuration that minimizes the overall network outage is selected.

The RPL network protocol is utilized to design a tree-based topology where each network device connects to its neighboring network devices according to the Free Space Path Loss (FSPL) formula given in equation 1 where D and λ are the distance between two network devices and the wavelength respectively.

$$FSPL = \log_{10} \frac{4\pi D}{\lambda} \tag{1}$$

every single device in the IoT network creates a neighborhood table where each device stores an indicator about its distance to its neighbors and the value of RPL rank for its neighbors. The distance between network devices is inferred from the received signal of the Signal to Noise Ratio (SNR) and it is determined using calculated according to the equation 2 and 3

$$Prx_{dBm} = Ptx_{dBm} + Gain - FSPL \tag{2}$$

$$SNR = Prx_{dBm} - (System_{dBm}) \tag{3}$$

In equations 2, Ptx_{dBm}, Prx_{dBm} and $Gain$ are the power of the transmitted signal, the received signal power and the system gain respectively. In equation 3 the $System_{dBm}$ is the sensitivity of the receiver for the receiving node (the value of the $System_{dBm}$ used in the simulations is $-85\ dBm$). The SNR is utilized to configure the link's success between the neighboring nodes. This link metric is used for determining the outage probabilities of the isomorphic sub-graphs of all clique to make the performance of the network better. The parameter of $Gain$ is set to one since the antenna is assumed to by the isotropic antenna.

After finding the link quality and the path loss, the network is partitioned into n-clique sub-networks. The adjacency matrix that found from the MDS code utilized to configure the topology of the sub-networks (i.e., cliques). The outage probability of the configured cliques can be determined by

- **Generalized MDS Code**

In the second step of the system model, the MDS codes were generated utilizing the approach proposed in [63]. Their MDS code is working for all network node number n and all erasures number r. The generated MDS codes for appropriate network configuration determine the way that the data and the parity symbols are stored. From this perspective, the nodes located inside the clique are needed to store additional p parity symbols in their local data memory for m information symbols belonging to their neighbors. The overlaps between neighboring cliques should be kept low to reduce the parity symbols stored number in all nodes.

The generated MDS code can retrieve all mn information symbols up to r network device loss, by linking to $k = n - r$ network device (k is the node number that utilized to recreate the failed node's information). The array code used in the MDS code is a vertical code type. It can store both the information and the parity symbols in each node, and the size array code is $(m + p) \times n$. Equation 4 is the vertical array code, and equation 5 is the generator matrix for such array code.

$$\begin{pmatrix} d_{0,0} & \cdots & d_{0,n-1} \\ \vdots & \ddots & \vdots \\ d_{m-1,0} & \cdots & d_{m-1,n-1} \\ f_{0,0} & \cdots & f_{0,n-1} \\ \vdots & \ddots & \vdots \\ f_{p-1,0} & \cdots & f_{p-1,n-1} \end{pmatrix}_{(m+p)\times n} \tag{4}$$

$$G = (I_{nm}|A), \tag{5}$$

where matrix A in equation 5 is the non-systematic part, and its size is $nm \times np$. The information is encoded as a vector matric multiplication, where the array of data d symbols of size nm and the codeword c of size $n(m + p)$

$$G = \left(\begin{array}{cccc|ccc} I_m & 0 & \cdots & 0 & A_{0,0} & \cdots & A_{0,n-1} \\ 0 & I_m & \cdots & 0 & A_{1,0} & \cdots & A_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & I_m & A_{n-1,0} & \cdots & A_{n-1,n-1} \end{array}\right) \tag{6}$$

The matrix of A in equation 6, can be separated into $n \times n$ blocks with size $m \times p$ where $A = [A_{i,j}]_{i,j=0}^{n-1}$. The value of m information symbols and p parity symbols can be found by using the equations given in

$$m = \frac{k}{gcd\ (k,r)} \quad \& \quad p = \frac{r}{gcd\ (k,r)}\ , \tag{7}$$

where the gcd in equation 7 is the greatest common divisor. The MDS array code can be enounce as a Kronecker product

$$A = \acute{A} \otimes D \epsilon \mathbb{F}_q^{nm \times np}, \epsilon \mathbb{F}_q^{k \times r} \tag{8}$$

$$D \epsilon \mathbb{F}_q^{(m+p)\times(m+p)}\ , \tag{9}$$

where D is the antidiagonal matrix.

For example, let assume we have a network with $n = 4$ wireless device that capable of retrieving $r = 2$ erased wireless device. We can calculate the information number and the parity symbols number for each node after calculating the $k = n - r$. The value of m and p are: $m = \frac{2}{\text{gcd}\ (2,2)} = 1 \quad , \quad p = \frac{2}{\text{gcd}\ (2,2)} = 1$. The array code for this network would be:

$$\begin{pmatrix} d_{0,0} & f_{0,1} & d_{0,2} & f_{0,3} \\ f_{0,0} & d_{0,1} & f_{0,2} & d_{0,3} \end{pmatrix} \tag{10}$$

From the network's array code, it can be seen that in each row, there are one parity ($p = 1$) symbols, which is determined from (4,2) MDS code. The array code of this network can be generated utilizing a generalized Reed-Solomon code. A Cauchy (Singleton) matric is used in the array code as its generator matrix [66]. In equation 5, the generator matrix that given as a Kronecker product could be expressed in the following forms

$$G = (I_2 | A) \quad \in \ \mathbb{F}_3^{2\times 2} \tag{11}$$

$$\acute{A} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \quad \in \ \mathbb{F}_3^{2\times 2} \tag{12}$$

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \ \otimes \ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 \\ 1 & 0 & 2 & 0 \end{pmatrix} \tag{13}$$

$$f = \left(d_{0,0} d_{0,1} d_{0,2} d_{0,3}\right) \times \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 \\ 1 & 0 & 2 & 0 \end{pmatrix} \tag{14}$$

Equation 14 shows that node 0 and node 2 do not require to be connected, and it is also showing that node 1 and node 3 do not need to be connected too. Thus, the generated MDS code will work for complete and incomplete networks with minimal degree 2. From equation 10 the parity symbols for each network device can be determined by the following formulas which can be found in 2-dimensional Galois Field $GF(2)$

$$f_{0,0} = d_{0,1} + d_{0,3} \tag{15}$$

$$f_{0,1} = d_{0,0} + d_{0,2} \tag{16}$$

$$f_{0,2} = d_{0,1} + 2d_{0,3} \tag{17}$$

$$f_{0,3} = d_{0,0} + 2d_{0,2} \tag{18}$$

These equations are showing that the node one and node three would only be connected to node two and node four, while the node two and node four would only be connected to node one and node three. Fig 19. shows the MDS coding [64].

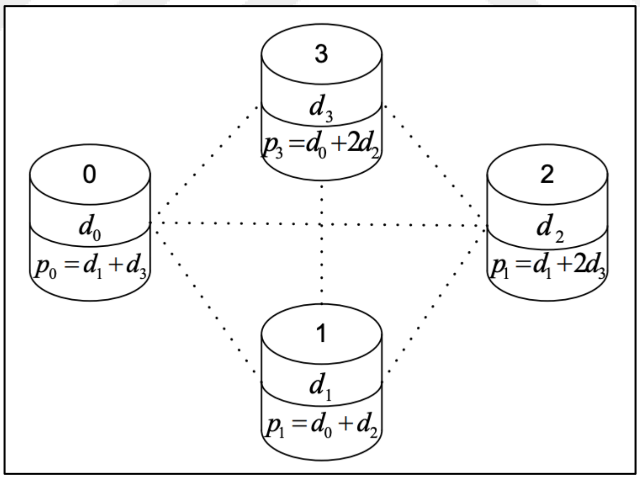


Figure 19. MDS coding [64].

- **Network Partitioning**

In the second part of the system model, the wireless IoT network is partitioned into sub-networks (Cliques). In this case, each wireless device creates a neighborhood table where each node keeps an indicator about its distance to its neighbors and its neighbor's

RPL rank values. The distance between the nodes is inferred from the received Signal to Noise Ratio.

The level of the SNR that received is mapped to link success probabilities for the purpose of calculating the outage probability within the clique. Here, the lowest outage probability configuration is chosen to minimize the outage probability of the entire wireless IoT network. Following the path loss and link quality modeling, the network is partitioned into n-clique sub-networks. Finally, the adjacency matrix from the MDS code is used to configure the topology of the cliques. The configured clique's outage probabilities can be determined by this formula,

$$p_{out} = 1 - \prod_{(i,j)\in\varepsilon} (1 - p_{i,j}) \approx \sum_{(i,j)\in\varepsilon} p_{i,j} \tag{19}$$

while the ε in equation 19 is the set of edges in the undirected connectedness graph [66]. Naturally, the isomorphic subgraph with the lowest outage probability, which is suitable for the MDS code configuration, is used for each clique. Here, the goal is to optimize the outage probability for the entire network by partitioning the network in such a way that the cumulative outage probabilities of the cliques are minimized.

- **Minimizing the outage of IoT Network**

In the last step of the proposed solution, the outage of the network is minimized using the equation 20

$$N_{out} = \frac{\sum_{w=1}^{u} x_w}{u} \tag{19}$$

where u and x_w are the number of clique formations and the outage of the clique formations respectively. Here, the clique formation that minimizes the average network outage is selected. The clique finding mechanism uses a heuristic approach to identify the cliques with minimum overlap. As a result, it starts with the minimum degree nodes which are placed at the corner of the network. This approach indeed minimizes the node overlaps as supported by the simulation results. Nevertheless, there generally are several network devices at the edge of the network with same minimum node degree. In this case, it is of interest to find the clique coverage with the minimum overlap providing the lowest average network outage. For example, in (figure 18 (b)), there are two different clique formations that have different initial nodes providing minimum node degree criteria. If we apply equation (20) to these clique formations, we get the average network outage probabilities of $N_{out}\ (CF1) = \frac{0.1899+0.1996+0.1931}{3} = 0.1942$ and $N_{out}\ (CF2) = \frac{0.2013+0.2301+0.1987}{3} = 0.2101$ respectively. According to these results, the formation that minimizes the network outage is the clique formation 1 in (figure 18 (b)).

B. Proposed Algorithm

A heuristic algorithm is proposed in this study that offers a generalized n- clique finding algorithm with minimal outage probability for a given configuration of the MDS code.

In the proposed heuristic algorithm, the network is partitioned by starting from the nodes that have minimum node degree. Generally, the nodes that placed at the boundary of the network are supposed to have the lowest degree (i.e., the lowest connections number). According to this, partitioning the network for the network's edge heuristically minimized the uncovered nodes number after the operation of the clique forming.

In this algorithm, the nodes are arranged according to their degrees, starting from the lowest degree. The nodes that match the connectivity criteria are involved in the clique. In the clique process, the process starts with the clique that has the least neighbor number.

Partitioning the network from a random degree node is the second approach proposed in this study. The second proposed approach partitions the network randomly instead of partitioning the network, starting from the lowest degree node. This random generalized clique finding algorithm begins to form a randomly selected node to start the process of the clique formation.

Obviously, starting the generalized clique finding algorithm from a random degree node may result in a more significant number of overlapping at the network's edge. Nevertheless, it may be possible to repeat this clique finding process to make better network outage probability. The outage performance results for the proposed generalized heuristic algorithms is compared with the Brute-Force (BF) algorithm to site their efficacy.

The Brute-Force algorithm' process starts by listing all the possible cliques in the network as an initial step. The created list by the BF algorithm utilized to create an adjacency matrix. After the initial step, all configurations of the possible clique are tried that covers the entire network with zero overlaps. Since the BF algorithm lists all the possible clique and tries all of it, it takes a longer time to finish, and it is not practical for real-time running implementation.

C. Performance Resilts

The system of the distributed storage that suggested in this study is implemented in MATLAB version 9.2 (R2017a) [67]. The tests are run on a processor type 2.9GHz Intel CORE i5 with RAM of 8 GB. The results of the suggested algorithms are set against each other, where Brute-Force algorithm is utilized as a benchmark.

In this study, two different network scenarios are used where 100 meters by 100 meters and 1000 meters by 1000 meters network configurations are simulated with increasing the node's number in the network.

The proposed generalized clique finding algorithms are tested for r erasures (i.e., redundancy) and n-cliques where the nodes inside these cliques are grouped in fully connected groups. The BF algorithm and the heuristic algorithm are analyzed for five different network topologies for both network scenarios (i.e., 1000m × 1000m and 100m × 100m network).

Table 1. Experimental results for 20 nodes, $n = 4$ and $r = 2$ for 100m × 100m network

Topolog-y	*Brute-Force*			*Min-Degree*				*Random-Degree*			
	Outage Probab-ility	*Clique Found*	*Uncove-red Nodes*	*Outage Probab-ility*	*Outage Probabi-lity [64]*	*Cliques Found*	*Uncove-red Nodes*	*Outage Probab-ility*	*Outage Probabi-lity [64]*	*Clique Found*	*Uncov-ered Nodes*
01	0.0423	*5*	*0*	0.0675	0.1025	*5*	*0*	0.0549	0.0978	*6*	*0*
02	0.0489	*5*	*0*	0.0569	0.1125	*5*	*0*	0.0574	0.1139	*6*	*0*
03	0.0481	*5*	*0*	0.0549	0.0961	*5*	*0*	0.0669	0.0907	*5*	*0*
04	0.0512	*5*	*0*	0.0625	0.1023	*5*	*0*	0.0522	0.1008	*6*	*0*
05	0.0473	*5*	*0*	0.0691	0.1087	*5*	*0*	0.0615	0.1071	*5*	*0*
Avg.	0.0476	*5*	*0*	0.0622	0.1044	*5*	*0*	0.0586	0.1021	*5.2*	*0*

The network at the first step is formed by 20 nodes, and the configuration of the clique is selected as $n = 4$ and $r = 2$. The configuration of the 20-node network is selected so that the BF clique finding algorithm finishes in a reasonable time. Besides these simulations, other simulations are done for different network sizes with different clique configurations ($n > 4$).

The result of simulations for the 20 nodes network showed that the BF algorithm outperforms in terms of clique coverage of all of the proposed heuristic algorithms. Table (1) is summarizing the result for different five networks with 20 nodes. From the results, it is possible to realize that the heuristic algorithm with the lowest node degree criteria outperforms the random node degree selection algorithm for the simulated topologies. On the other hand, all of the heuristic algorithms can cover the network with some overlaps.

In the table (1), the results are also showing that the average of the network outage performance of the suggested heuristic clique finding approach is better than the approach that proposed in [64]. In this case, it is concluded that the distributed storage system of the data-sharing part presented in this study will need fewer data transmissions number within the network, which improves both energy efficiency and reliability for the proposed solution.

In table (2), we can see that the proposed heuristic algorithms outperform Brute-Force algorithm in terms of simulation time. Table (2) is also proves that the heuristic algorithms can execute in real time.

Table 2. Simulation times

Topology	**Brute-Force Algorithm**	**Heuristic Algorithms**
01	4 **Hours**	1.1 **Seconds**
02	1 **Hours**	1.5 **Seconds**
03	6 **Hours**	1.0 **Seconds**
04	9 **Hours**	1.9 **Seconds**
05	3 **Hours**	1.5 **Seconds**

Table (3) is about the network outage results for various network sizes with different clique configurations for the proposed heuristic approaches. Tested simulations are done in a rectangular area 1000m by 1000m, where nodes inside the network are selected to be 100, 300 and 500.

As expected, the uncovered nodes in the network will increase with increasing the size of the clique for low-density networks (i.e., $N = 100$). In case the network density is higher (i.e., $N = 300$ and 500 node networks), it is possible to form cliques for all nodes that locates inside the topology. Of course, the clique size may negatively impact the coverage of the node in such a higher density scenario too.

Table 3. Results for 1000m × 1000m network

N	*n-Clique*	*r*	*Heuristic Algorithm Min-Degree*			*Heuristic Algorithm Random-Degree*		
			Network Outage	*Cliques Found*	*Uncovered Nodes*	*Network Outage*	*Cliques Found*	*Uncovered Nodes*
100	*4*	*2*	*0.1555*	*23*	*13*	*0.1546*	*27*	*14*
	6	*3*	*0.1081*	*11*	*50*	*0.1032*	*8*	*63*
	8	*4*	*0.0819*	*4*	*75*	*0.0799*	*4*	*75*
300	*4*	*2*	*0.1392*	*79*	*0*	*0.1309*	*87*	*0*
	6	*3*	*0.0998*	*54*	*7*	*0.0942*	*62*	*8*
	8	*4*	*0.0764*	*44*	*20*	*0.0751*	*44*	*37*
500	*4*	*2*	*0.1198*	*127*	*0*	*0.1151*	*134*	*0*
	6	*3*	*0.0909*	*88*	*0*	*0.0929*	*101*	*0*
	8	*4*	*0.0726*	*67*	*0*	*0.0713*	*80*	*2*

Another satisfying result in the table (3) is that the clique finding an algorithm with a random degree of node selection outperforms the heuristic algorithm with lowest degree node selection for higher density networks in terms of network's outage probability at the expense of more node overlaps. These results can be illustrated by the fact that having many nodes that participating in more than one clique may make the outage of the clique lower. Nevertheless, the storage need of such a solution may be unreasonable for such limited power Internet of Things networks.

Table 4. Average results for 1000m x 1000m three different networks

$N = 200 \quad n = 4, r = 2$			$n = 6, r = 3$			$n = 8, r = 4$			Try
Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	
Minimum degree									
0.1498	55	0	0.1047	44	22	0.0783	14	114	
Random degree									
0.1528	59	0	0.0984	35	38	0.0721	14	114	1
0.1513	59	0	0.0994	38	30	0.0716	13	120	2
0.1591	59	1	0.0990	38	24	0.0716	13	120	3
0.1535	58	0	0.0981	36	30	0.0730	14	114	4
0.1554	60	2	0.1020	36	31	0.0750	13	118	5
0.1504	62	0	0.0999	37	28	0.0643	12	125	6
0.1445	56	0	0.1014	37	21	0.0760	13	115	7

Table 4. Average results for 1000m x 1000m three different networks (Continue)

0.1559	61	0	0.1007	39	26	0.0716	13	113	8
0.1426	57	0	0.0978	38	26	0.0688	14	116	9
0.1495	59	1	0.1013	40	28	0.0706	12	128	10
$\mathrm{N} = 400$ $\mathrm{n} = 4, \mathrm{r} = 2$			$\mathrm{n} = 6, \mathrm{r} = 3$			$\mathrm{n} = 8, \mathrm{r} = 4$			Try
Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	
Minimum degree									
0.1309	102	0	0.0975	71	0	0.0759	57	2	
Random degree									
0.1558	112	0	0.0976	79	2	0.0729	70	5	1
0.1616	110	0	0.0992	78	0	0.0740	65	7	2
0.1536	111	0	0.0988	81	1	0.0722	64	9	3
0.1582	110	0	0.0990	81	0	0.0739	60	5	4
0.1498	111	0	0.0946	80	0	0.0739	65	6	5
0.1563	108	0	0.0975	82	0	0.0719	67	5	6
0.1593	112	0	0.0994	83	0	0.0709	63	10	7
0.1569	111	0	0.0977	81	0	0.0711	65	9	8
0.1555	109	0	0.1015	79	0	0.0740	65	10	9
0.1472	112	0	0.0993	77	1	0.0735	61	7	10
$\mathrm{N} = 600$ $\mathrm{n} = 4, \mathrm{r} = 2$			$\mathrm{n} = 6, \mathrm{r} = 3$			$\mathrm{n} = 8, \mathrm{r} = 4$			Try
Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	Outage Probability	Clique Found	Uncover-ed Node	
Minimum degree									
0.1141	153	0	0.0893	104	0	0.0697	78	0	
Random degree									
0.1221	165	0	0.0918	113	0	0.0721	91	0	1
0.1120	162	0	0.0926	117	0	0.0688	93	0	2
0.1188	163	0	0.0862	115	0	0.0709	92	1	3
0.1099	158	0	0.0892	118	0	0.0659	96	1	4
0.1191	161	0	0.0799	113	0	0.0689	94	4	5
0.1257	162	0	0.0998	114	0	0.0729	95	1	6
0.1219	162	0	0.0968	114	0	0.0747	91	2	7
0.1175	160	0	0.0813	113	0	0.0673	91	1	8
0.1102	161	0	0.0973	113	0	0.0661	94	2	9
0.1263	159	0	0.0884	116	0	0.0707	90	1	10

The third table is the result of the further investigations done on the performance of the heuristic algorithms. These simulations are run for three various network sizes (200, 400, and 600 nodes respectively) with different clique configurations (4, 6 and 8 respectively) and different erasures. The simulations are done for three different random network topologies, and table (4) summarizes the averaged results.

For the heuristic clique finding an algorithm with random node selection, 10 simulations are done starting from a different node degree randomly in the network for each network size. The purpose form these simulations is to determine if it is possible to outperform the clique finding algorithm with minimum node degree selection.

The results show that there is a chance to find better network settings utilizing a random node degree algorithm in terms of network's outage probabilities at the cost of a bit larger clique overlap for the networks with low density (i.e., 200 node network). Here, the results form table (4) shown that the random node degree selection algorithm can get closer to the algorithm with minimum node degree selection performance with only 10 iterations for low network density (i.e., 200 nodes).

For the higher densities, it has been noticed that the algorithm with minimum node degree selection outperforms the random node degree selection algorithm. This result shows that the networks with higher density may require a higher iterations number to optimize the performance of the random node degree approach. This approach can be parallelized easily, and the performance of this approach can be improved at the cost of a higher computation cost.

It is noticeable from these results that the outage for any network size reduces as the clique size increases. The outage of the network for each heuristic clique finding approach proposed in this study have similar value with increasing the clique sizes.

Figure 20 and Figure 21 are showing the percentage of the uncovered node for three different clique configurations and the percentage of the overlapped node for three different clique configurations, respectively.

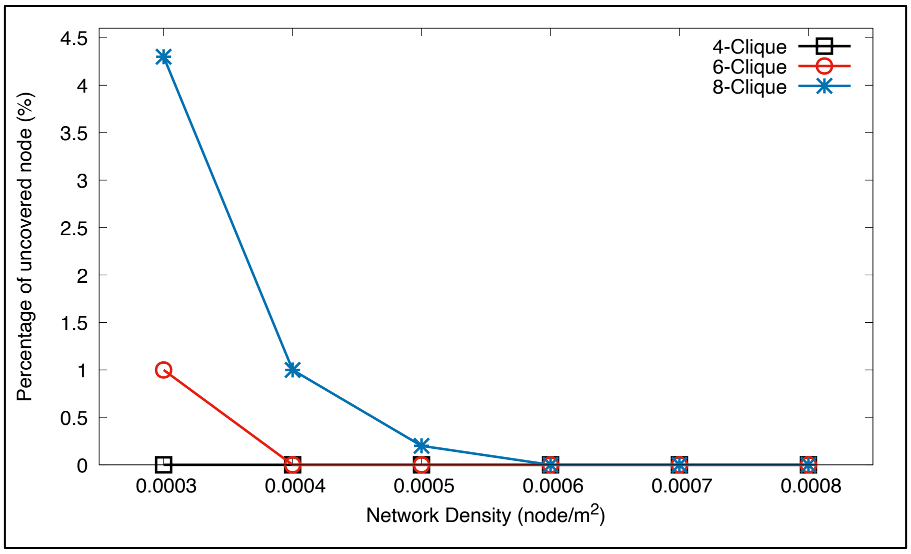


Figure 20. Percentage of the uncovered node for three different clique configurations

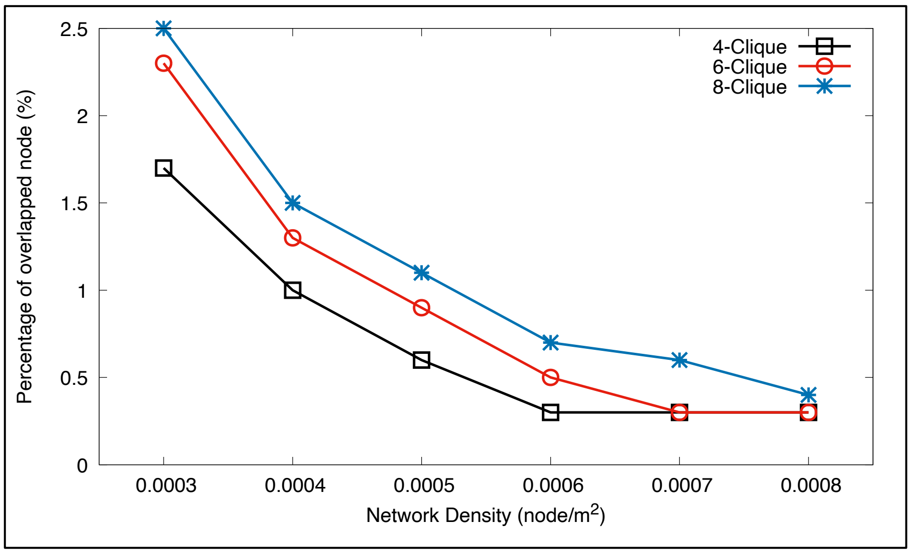


Figure 21. Percentage of the overlapped node for three different clique configurations

From these figures, it very well may be seen that the percentage of both of the overlapped nodes and the uncovered node decreases as the wireless IoT network densities increase.

In the simulated network deployment for this study, some of the nodes of the network may be located far from other nodes, preventing them from becoming inside the clique. In this case, the node that locates far from nodes can choose to act as an independent clique with a fewer number of clique members.

4. CONCLUSIONS AND FUTURE WORKS

Large number of IoT network devices are expected to be utilized everywhere shortly. This deployment will be responsible for a massive Internet's traffic. In this study, a solution of a distributed storage for wireless IoT network is proposed with the purpose of lowering the effect of IoT network's traffic on the Internet infrastructure.

The goal is to design reliable storage and retrieval of the IoT device information within the network that reduces this IoT network traffic. An MDS based erasure code model is utilized to store the information of the IoT devices inside the network locally.

Two heuristic algorithms are proposed in this study are about partitioning the network to sub-network to reduce the traffic of the network. Conclusion of this study show that it is probable to get a suitable coverage of n-cliques with a suitable network outage utilizing a heuristic approach. The MDS erasure code for a distributed storage has been shown to make the retrieval of the information generated in the network better.

As a future works, it may be possible to decrease the outage of the network further by improving the clique finding algorithm by creating a machine learning-based algorithm that starts the clique search from different locations with a suitable node in the IoT network.

5. REFERENCES

1. https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT Definition of Internet of Things. 1 December 2019.

2. Atzori, L., Iera, A. ve Morabito, G., The Internet of things: A survey, Computer Networks,15 (2010) 2787-2805.

3. https://www.ibm.com/blogs/industries/little-known-story-first-iot-device/ IBM The little-known story of the first IoT device. 1 December 2019.

4. https://www.autoidlabs.org Auto-ID Labs 2 December 2019.

5. Zameel, A., Najmuldeen, M. ve Gormus, S., Context-Aware Caching in Wireless IoT Networks, 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), (2019) 712-717.

6. https://data-flair.training/blogs/how-iot-works/ Data Flair How IoT works. 2 December 2019.

7. Soumyalatha, S.G.H., Study of IoT: Understanding IoT architecture, applications, issues and challenges, 1st International Conference on Innovations in Computing & Networking (ICICN16), (2016).

8. Kraijak, S. ve Tuwanut, P., A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends, (2015).

9. Bandyopadhyay, D. ve Sen, J., Internet of things: Applications and challenges in technology and standardization, Wireless personal communications, 58,1 (2011) 49-69.

10. Smith, I.G., Vermesan, O., Friess, P. ve Furness, A., The Internet of Things New Horizons, 2012.
11. Welbourne, E., Battle, L., Cole, G., Gould, K., Rector, K., Raymer, S., Balazinska, M. ve Borriello, G., Building the Internet of things using RFID: the RFID ecosystem experience, IEEE Internet computing, 13,3 (2009) 48-55.

12. https://www.elprocus.com/introduction-to-wireless-sensor-networks-types-and-applications/ Elprocus Wireless Sensor Networks and their Applications. 1 December 2019.

13. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y. ve Cayirci, E., Wireless sensor networks: a survey, Computer Networks, 38,4 (2002) 393-422.

14. Estrin, D., Govindan, R., Heidemann, J. ve Kumar, S., Next century challenges: Scalable coordination in sensor networks, Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, (1999) 263-270.

15. Matin, M.A. ve Islam, M., Overview of wireless sensor network, Wireless Sensor Networks-Technology and Protocols, (2012) 1-3.

16. Alkhatib, A.A.A. ve Baicher, G.S., Wireless sensor network architecture, International Conference on Computer Networks and Communication Systems, (2012).

17. Akyildiz, I.F., Melodia, T. ve Chowdhury, K.R., A survey on wireless multimedia sensor networks, Computer Networks, 51,4 (2007) 921-960.

18. Sethi, P. ve Sarangi, S.R., Internet of Things: Architectures, Protocols, and Applications, Journal of Electrical and Computer Engineering, (2017) 25.

19. https://www.cloudflare.com/learning/ddos/glossary/internet-protocol/ Cloud Flare What is the Internet Protocol. 2 December 2019.

20. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M. ve Ayyash, M., Internet of things: A survey on enabling technologies, protocols, and applications, IEEE communications surveys & tutorials, 17,4 (2015) 2347-2376.

21. Salman, T. ve Jain, R., Networking protocols and standards for Internet of things, Internet of Things and Data Analytics Handbook, 7 (2015).

22. https://iot6.eu/6tisch IoT6 6TiSCh. 1 December 2019.

23. Bormann, C., Castellani, A.P. ve Shelby, Z., Coap: An application protocol for billions of tiny internet nodes, IEEE Internet computing,2 (2012) 62-67.

24. B., V.J.a.R.M., Internet of Things (IoT): Security Analysis & Security Protocol CoAP, International Journal of Recent Trends in Engineering & Research (IJRTER), 3,3 (2017) 417-425.

25. https://www.avsystem.com/blog/iot-protocols-and-standards/ AVSystem IoT Standards and Protocols Guide—Protocols of the Internet of Things., 1 December (2019).

26. http://mqtt.org MQTT Internet of Things protocol. 4 December 2019.

27. Schmitt, A., Carlier, F. ve Renault, V., Dynamic bridge generation for IoT data exchange via the MQTT protocol, Procedia computer science, 130, (2018) 90-97.

28. Luzuriaga, J.E., Perez, M., Boronat, P., Cano, J.C., Calafate, C. ve Manzoni, P., Improving mqtt data delivery in mobile scenarios: Results from a realistic testbed, Mobile Information Systems, (2016).

29. Kushalnagar, N., Montenegro, G. ve Schumacher, C., IPv6 over low-power wireless personal area networks (6LoWPANs), 2007.

30. Hinden, R., Internet protocol version 6 (IPv6) specification, 1998.

31. Hui, J.W. ve Culler, D.E., Extending IP to low-power, wireless personal area networks, IEEE Internet computing, 4 (2008) 37-45.

32. Sharma, C. ve Gondhi, N.K., Communication Protocol Stack for Constrained IoT Systems, 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU), (2018) 1-6.

33. Mehmood, T., COOJA Network Simulator: Exploring the Infinite Possible Ways to Compute the Performance Metrics of IOT Based Smart Devices to Understand the Working of IOT Based Compression & Routing Protocols, arXiv preprint arXiv:1712.08303, (2017).

34. Gan, W., Shi, Z., Zhang, C., Sun, L. ve Ionescu, D., MERPL: A more memory-efficient storing mode in RPL, 19th IEEE International Conference on Networks (ICON), (2013) 1-5.

35. Association, I.S., 802.15. 4-2011 IEEE Standard for Local and Metropolitan Area Networks-Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), 2011.

36. Techel, J., Zhao, X., Talasila, P., Zhang, Q. ve Lucani, D.E., Demonstration of Reliable IoT Distributed Storage using Network Codes, 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), (2019) 1-2.

37. Zhang, Q., Sun, Y., Qu, S., Zhang, J. ve Wang, X., Performance analysis on distributed storage systems in ring networks, IEEE/CIC International Conference on Communications in China (ICCC), (2017) 1-6.

38. Milani, B.A. ve Navimipour, N.J., A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions, Journal of Network and Computer Applications, 64 (2016) 229-238.

39. Liu, Y. ve Vlassov, V., Replication in distributed storage systems: State of the art, possible directions, and open issues, International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, (2013) 225-232.
40. Gill, N.K. ve Singh, S., Dynamic cost-aware re-replication and rebalancing strategy in cloud system, Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA), (2015) 39-47.

41. Dimakis, A.G., Godfrey, P.B., Wu, Y., Wainwright, M.J. ve Ramchandran, K., Network coding for distributed storage systems, IEEE transactions on information theory, 56,9 (2010) 4539-4551.

42. Dutta, P., Guerraoui, R. ve Levy, R.R., Optimistic erasure-coded distributed storage, International Symposium on Distributed Computing, (2008) 182-196.

43. Li, J. ve Li, B., Erasure coding for cloud storage systems: a survey, Tsinghua Science and Technology, 18,3 (2013) 259-272.

44. Dimakis, A.G., Ramchandran, K., Wu, Y. ve Suh, C., A survey on network codes for distributed storage, Proceedings of the IEEE, 99,3 (2011) 476-489.

45. Shah, N.B., Rashmi, K., Kumar, P.V. ve Ramchandran, K., Interference alignment in regenerating codes for distributed storage: Necessity and code constructions, IEEE transactions on information theory, 58,4 (2011) 2134-2158.

46. https://wiki.cse.buffalo.edu/cse545/content/mds-codes MDS Codes. 5 December 2019.

47. https://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed_solomon_codes.html An introduction to Reed-Solomon codes: principles, architecture and implementation. 5 December 2019.

48. Sandell, M. ve Tosato, F., Lowest-density MDS array codes for reliable Smart Meter networks, Transactions on Emerging Telecommunications Technologies, 26,11 (2015) 1251-1264.

49. Yang, S., Gao, Z., Huang, R. ve Qiu, X., A data correlation-based virtual clustering algorithm for wireless sensor network, Seventh International Conference on Mobile Ad-hoc and Sensor Networks, (2011) 335-336.

50. Deshpande, V.V. ve Patil, A.R.B., Energy efficient clustering in wireless sensor network using cluster of cluster heads, tenth international conference on wireless and optical communications networks (WOCN), (2013) 1-5.

51. Ebadi, S., Ghasembaglou, M., Navin, A.H. ve Mirnia, M.K., Energy balancing in wireless sensor networks with selecting two cluster-heads in hierarchical clustering, International Conference on Computational Intelligence and Communication Networks, (2010) 230-233.

52. Soni, S. ve Dey, B., Dynamic selection of cluster head in cluster of cluster heads within the cluster in Heterogeneous Wireless Sensor Network, IEEE International Conference on Advanced Communications, Control and Computing Technologies, (2014) 877-881.

53. Sun, K., Peng, P., Ning, P. ve Wang, C., Secure distributed cluster formation in wireless sensor networks, 22nd Annual Computer Security Applications Conference (ACSAC'06) (2006) 131-140.

54. Bandyopadhyay, S. ve Coyle, E.J., An energy efficient hierarchical clustering algorithm for wireless sensor networks, IEEE INFOCOM. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies, (2003) 1713-1723.

55. Chatterjee, M., Das, S.K. ve Turgut, D., WCA: A weighted clustering algorithm for mobile ad hoc networks, Cluster computing, 5,2 (2002) 193-204.

56. Al-Awami, L. ve Hassanein, H.S., Robust decentralized data storage and retrieval for wireless networks, Computer Networks, 128 (2017) 41-50.

57. Xing, J., Dai, H. ve Yu, Z., A distributed multi-level model with dynamic replacement for the storage of smart edge computing, Journal of Systems Architecture, 83 (2018) 1-11.

58. Talari, A. ve Rahnavard, N., CStorage: Decentralized compressive data storage in wireless sensor networks, Ad Hoc Networks, 37 (2016) 475-485.

59. Itani, M., Sharafeddine, S. ve ElKabani, I., Dynamic multiple node failure recovery in distributed storage systems, Ad Hoc Networks, 72 (2018) 1-13.

60. Hendricks, J., Ganger, G.R. ve Reiter, M.K., Verifying distributed erasure-coded data, Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing, (2007) 139-146.

61. Chou, P.A. ve Wu, Y., Network coding for the Internet and wireless networks, IEEE Signal Processing Magazine, 24,5 (2007) 77.

62. Li, J., Zhang, P., Li, Y., Chen, W., Liu, Y. ve Wang, L., A data-check based distributed storage model for storing hot temporary data, Future Generation Computer Systems, 73 (2017) 13-21.

63. Sandell, M. ve Tosato, F., Lowest density MDS array codes on incomplete graphs, 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton), (2013) 645-652.

64. Gormus, S., Koc, A.G., Jin, Y. ve Sooriyabandara, M., A storage centric approach to scalable sensor networks, IEEE 81st Vehicular Technology Conference (VTC Spring), (2015) 1-7.

65. Najmuldeen, M. ve Gormus, S., A Generic Clique Based Distributed Caching Approach For Wireless IoT Networks, 4th International Conference on Computer Science and Engineering (UBMK), (2019) 1-5.

66. Roth, R.M. ve Seroussi, G., On generator matrices of MDS codes (corresp.), IEEE transactions on information theory, 31,6 (1985) 826-830.

67. https://www.mathworks.com/products/matlab.html MathWorks MATLAB. 10 December 2019.

CURRICULUM VITAE

Mustafa NAJMULDEEN was born on the 16th of December 1994 in Kirkuk, Iraq. After finishing his primary education at Ata TERZİBAŞI primary school and his secondary education at Parlak secondary school, NAJMULDEEN got his B.Sc. from the Computer Engineering department at Al Qalam University, Kirkuk, Iraq in 2016. In 2017, he started his master's studies in Computer Engineering at Karadeniz Technical University in Trabzon, Turkey. NAJMULDEEN speaks English and Arabic very well as well as his mother language, which is Turkish.

Publications

1. M. Najmuldeen and S. Gormus, "A Generic Clique Based Distributed Caching Approach For Wireless IoT Networks," *2019 4th International Conference on Computer Science and Engineering (UBMK)*, Samsun, Turkey, 2019, pp. 1-5, doi: 10.1109/UBMK.2019.8907052.
2. A. Zameel, M. Najmuldeen and S. Gormus, "Context-Aware Caching in Wireless IoT Networks," *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*, Bursa, Turkey, 2019, pp. 712-717, doi: 10.23919/ELECO47770.2019.8990647.