

**ENHANCING APPLICATION LAYER  
PERFORMANCE THROUGH CONGESTION  
CONTROL IN CoAP BASED INTERNET OF  
THINGS**

*Dissertation submitted in fulfilment of the requirements for the Degree of*

**MASTER OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

By

**ANANYA PRAMANIK**

**11502916**

Supervisor

**ISHA**



**School of Computer Science and Engineering**

Lovely Professional University

Phagwara, Punjab (India)

Month May, Year 2017

@ Copyright LOVELY PROFESSIONAL UNIVERSITY, Punjab (INDIA)

Month May, Year 2017

ALL RIGHTS RESERVED

**TOPIC APPROVAL PERFORMA**

School of Computer Science and Engineering

Program : P172::M.Tech. (Computer Science and Engineering) [Full Time]

COURSE CODE : CSE546

REGULAR/BACKLOG : Regular

GROUP NUMBER : CSERGD0228

Supervisor Name : Isha

UID : 17451

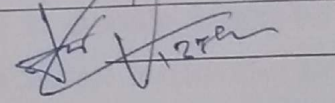
Designation : Assistant Professor

Qualification : M.Tech

Research Experience : 4 year

SR.NO.	NAME OF STUDENT	REGISTRATION NO	BATCH	SECTION	CONTACT NUMBER
1	Ananya Pramanik	11502916	2015	K1518	7589422669

SPECIALIZATION AREA : Networking and Security

Supervisor Signature: 

PROPOSED TOPIC : Enhancing Application Layer Performance through Congestion Control in CoAP Based Internet of Things

Qualitative Assessment of Proposed Topic by PAC		
Sr.No.	Parameter	Rating (out of 10)
1	Project Novelty: Potential of the project to create new knowledge	7.80
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	6.80
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	7.20
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	7.40
5	Social Applicability: Project work intends to solve a practical problem.	7.60
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	7.20

PAC Committee Members		
PAC Member 1 Name: Prateek Agrawal	UID: 13714	Recommended (Y/N): Yes
PAC Member 2 Name: Pushpendra Kumar Pateriya	UID: 14623	Recommended (Y/N): Yes
PAC Member 3 Name: Deepak Prashar	UID: 13897	Recommended (Y/N): Yes
PAC Member 4 Name: Kewal Krishan	UID: 11179	Recommended (Y/N): Yes
PAC Member 5 Name: Anupinder Singh	UID: 19385	Recommended (Y/N): NA
DAA Nominee Name: Kanwar Preet Singh	UID: 15367	Recommended (Y/N): Yes

Final Topic Approved by PAC: Enhancing Application Layer Performance through Congestion Control in CoAP Based Internet of Things

Overall Remarks: Approved

Approval Date: 26 Oct 2016

PAC CHAIRPERSON Name: 11011::Dr. Rajeev Sobti

## ABSTRACT

---

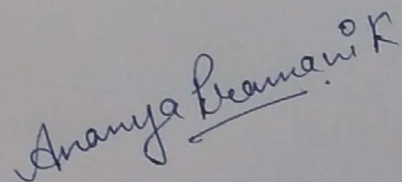
Internet of Things (IoT) is a network of interconnected sensors designed for constrained lossy and low powered network (LLN). Congestion in LLN occurs due to limited radio channel capacity, resource constrained, low bandwidth and high scale of communication. The Constrained Application Protocol (CoAP) is a lightweight RESTful application layer protocol working on maintaining reliability and congestion control over the constrained devices over UDP layer. Default CoAP has been standardised by Internet Engineering Task Force (IETF) to provide congestion Control for CoAP. Due to limitation in default congestion control mechanism, an alternative congestion control mechanism known as CoAP Congestion Control Advance (CoCoA) is introduced by new Internet- Draft. This paper evaluates congestion control performance for Simple UDP IPv6 network, Default CoAP and CoCoA using Cooja simulator in Contiki OS. For better result, modified CoCoA was evaluated by adding Re-Establishment mechanism that will terminate the transmission after n number of retransmissions and save the packet unless CoAP server is free. The results compare loss percentage of packet loss, goodput, Average RTT (Round Trip Time) value and RTO calculation in variety of network topologies with varying number of client and server. The result shows that CoCoA outperforms than Simple UDP IPv6 network and default CoAP in most of the scenarios.

## DECLARATION STATEMENT

---

I hereby declare that the research work reported in the dissertation entitled "ENHANCING APPLICATION LAYER PERFORMANCE THROUGH CONGESTION CONTROL IN CoAP BASED INTERNET OF THINGS" in partial fulfilment of the requirement for the award of Degree for Master of Technology in Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab is an authentic work carried out under supervision of my research supervisor Mrs. Isha. I have not submitted this work elsewhere for any degree or diploma.

I understand that the work presented herewith is in direct compliance with Lovely Professional University's Policy on plagiarism, intellectual property rights, and highest standards of moral and ethical conduct. Therefore, to the best of my knowledge, the content of this dissertation represents authentic and honest research effort conducted, in its entirety, by me. I am fully responsible for the contents of my dissertation work.



*Signature of Candidate*

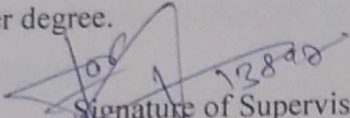
**ANANYA PRAMANIK**

**Registration No 11502916**

## SUPERVISOR'S CERTIFICATE

---

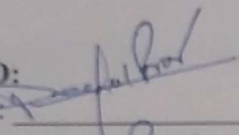
This is to certify that the work reported in the M.Tech Dissertation entitled "ENHANCING APPLICATION LAYER PERFORMANCE THROUGH CONGESTION CONTROL IN CoAP BASED INTERNET OF THINGS", submitted by Ananya Pramanik at Lovely Professional University, Phagwara, India is a bonafide record of his / her original work carried out under my supervision. This work has not been submitted elsewhere for any other degree.

  
Signature of Supervisor

ISHA

Date:

Counter Signed by:

1) Concerned HOD:   
HoD's Signature: \_\_\_\_\_  
HoD Name: Deepali Bhatnagar  
Date: 27/04/2012

2) Neutral Examiners:

**External Examiner**

Signature: \_\_\_\_\_

Name: \_\_\_\_\_

Affiliation: \_\_\_\_\_

Date: \_\_\_\_\_

**Internal Examiner**

Signature: \_\_\_\_\_

Name: \_\_\_\_\_

Date: \_\_\_\_\_

## **ACKNOWLEDGEMENTS**

---

This thesis is the culmination of my journey of Masters which was just like climbing a high peak step by step accompanied with encouragement, hardship, trust, and frustration. When I found myself at top experiencing the feeling of fulfilment, I realized though only my name appears on the cover of this dissertation, a great many people including my family members, well-wishers, my friends, colleagues and various institutions have contributed to accomplish this huge task.

First and foremost, I offer my sincerest gratitude to my supervisor, Isha, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. I attribute the level of my Master's degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. One simply could not wish for a better or friendlier supervisor.

I acknowledge the people who mean a lot to me, my parents, Siddhartha Shankar Pramanik and Uma Pramanik, for showing faith in me and giving me liberty to choose what I desired. I salute you all for the selfless love, care, pain and sacrifice you did to shape my life. Although you hardly understood what I researched on, you were willing to support any decision I made. I would never be able to pay back the love and affection showered upon by my parents. Also, I express my thanks to my brother Ankush support.

I thank the Almighty for giving me the strength and patience to work through all these years so that today I can stand proudly with my head held high.

**ANANYA PRAMANIK**

# TABLE OF CONTENTS

CONTENTS	PAGE NO
Inner first page – Same as Cover	i
PAC form	ii
Abstract	iii
Declaration by the Scholar	iv
Supervisor’s Certificate	v
Acknowledgement	vi
Table of Contents	vii
List of Abbreviations	ix
List of Tables	x
List of Figures	xi
<b>CHAPTER 1: INTRODUCTION</b>	<b>1-6</b>
1.1 INTERNET OF THINGS (IoT)	1
1.2 IoT SYSTEM ARCHITECTURE	2
1.3 PROTOCOLS SUITED FOR IoT	3
1.3.1 CHALLENGES IN WSN	3
1.3.2 STANDARD PROTOCOL FOR IoT	4
<b>CHAPTER 2: REVIEW OF LITERATURE</b>	<b>7-19</b>
2.1 CONSTRAINED APPLICATION PROTOCOL FOR INTERNET OF THINGS	7
2.1.2 CoAP VS HTTP	7
2.2 INTRODUCTION TO ENHANCING APPLICATION LAYER PERFORMANCE THROUGH CONGESTION CONTROL IN CoAP BASED INTERNET OF THINGS	8
2.3 HOW CONGESTION CONTROL IS DIFFERENT IN IoT THAN WSN?	9
2.4 MECHANISM OF VARIOUS CONGESTION CONTROL IN CoAP	10
2.4.1 TRADITIONAL CoAP CONGESTION	10



	CONTROL WITH EXP BACK-OFF	
	2.4.2 ADAPTIVE BACKOFF ALGORITHM	11
	2.4.3 EFFECTIVE STEAMING OVER CoAP	13
	2.4.4 ALTERNATE CONGESTION	13
	CONTROL ALGORITHMS	
	2.4.5 COCOA	15
	2.5.6 CONGESTION AVOIDANCE	17
	ALGORITHM IN SNS ENVIRONMENT	
<b>CHAPTER 3:</b>	<b>SCOPE OF STUDY</b>	20-21
<b>CHAPTER 4:</b>	<b>PRESENT WORK</b>	22-33
	4.1 PROBLEM FORMULATION	22
	4.2 OBJECTIVE OF THE STUDY	24
	4.3 RESEARCH METHODOLOGY	24
	4.3.1 PROPOSED ALGORITHM	25
	4.3.2 FLOW CHART	27
	4.3.3 TOOLS USED	27
	4.3.4 SIMULATION SETUP	28
<b>CHAPTER 5:</b>	<b>RESULT AND DISCUSSION</b>	34-44
	5.1 EXPERIMENTAL ANALYSIS OF PACKET LOSS PERCENTAGE	34
	5.2 EXPERIMENTAL ANALYSIS OF GOODPUT	36
	5.3 EXPERIMENTAL ANALYSIS OF RTO CALCULATION FOR ALGORITHMS	38
	5.3.1 SIMPLE CoAP ALGORITHM	38
	5.3.2 DEFAULT CoAP	39
	5.3.3 CoCoA ALGORITHM	39
	5.4 EXPERIMENTAL ANALYSIS OF RTT CALCULATION FOR ALGORITHMS	42
<b>CHAPTER 6:</b>	<b>CONCLUSION AND FUTURE SCOPE</b>	45
	6.1 CONCLUSION	45
	6.2 FUTURE SCOPE	45
	<b>REFERENCES</b>	47

## LIST OF ABBREVIATIONS

---

<b>ABBREVIATION</b>	<b>FULL FORM</b>
CoAP	Constrained Application Protocol
CoCoA	CoAP Simple Congestion Control Algorithm
DODAG	Destination Oriented Directed Acyclic Graph
IETF	Internet Engineering Task Force
IoT	Internet of Things
REST	Representational State Transfer
RPL	Routing Protocol for Low-power and Lossy Network
RTO	Retransmission Time Out
RTT	Round Trip Time
URI	Universal Resource Identifier
WSN	Wireless Sensor Network
6LoWPAN	Low-power Wireless Personal Area Network

## LIST OF TABLES

<b>TABLE NO.</b>	<b>TABLE DESCRIPTION</b>	<b>PAGE NO</b>
<b>Table 2.1</b>	DIFFERENCE BETWEEN CONGESTION DETECTION BETWEEN IoT AND WSN	10
<b>Table 2.2</b>	OUTPUT EVALUATION OF DIFFERENT CONGESTION ALGORITHMS	18
<b>Table 2.3</b>	COMPARING DIFFERENT CONGESTION CONTROL PARAMETERS	19
<b>Table 5.1</b>	TABLE FOR ANALYSING GOODPUT	36

# LIST OF FIGURES

FIGURE NO.	FIGURES	PAGE NO
<b>Figure 1.1</b>	HYPE CYCLE	2
<b>Figure 1.2</b>	IoT ARCHITECTURAL LAYER	2
<b>Figure 1.3</b>	VIEW OF 6LOWPAN IN IPV6 STACK PROTOCOL	5
<b>Figure 1.4</b>	RPL ROUTING TREE: DODAG	6
<b>Figure 1.5</b>	CoAP PROTOCOL STACK	6
<b>Figure 2.1</b>	HTTP AND CoAP STACK	8
<b>Figure 2.2</b>	FLOW CHART FOR EXPONENTIAL BACKOFF ALGORITHM	12
<b>Figure 2.3</b>	FLOW CHART FOR FIBONACCI BACKOFF ALGORITHM	12
<b>Figure 2.4</b>	CONNECTION BETWEEN THE CONTROL PLANE AND DATA PLANE	18
<b>Figure 3.1</b>	DETECTION OF CONGESTION IN APPLICATION LAYER OF CoAP IoT ENVIRONMENT	21
<b>Figure 4.1</b>	AN OVERVIEW OF THE DIFFERENT RTO VARIABLES USED IN CONGESTION CONTROL IN CoAP	22
<b>Figure 4.2</b>	CONGESTION CONTROL METHODOLOGY FLOW CHART	27
<b>Figure 4.3</b>	THE OVERVIEW OF IoT ARCHITECTURE LAYER AND DIFFERENCE BETWEEN IEFT PROTOCOL STACK (LEFT) AND CONTIKI OS STACK (RIGHT)	29
<b>Figure 4.4</b>	HARDWARE SPECIFICATION OF TMOTE SKY AND Z1 MOTE	30
<b>Figure 4.5</b>	THE THREE NETWORK TOPOLOGIES (CHAIN, GRID, DUMBBELL)	31
<b>Figure 5.1</b>	SAMPLE RESULT TERMINAL	34
<b>Figure 5.2</b>	ANALYSIS OF VARIOUS ALGORITHMS FOR	35

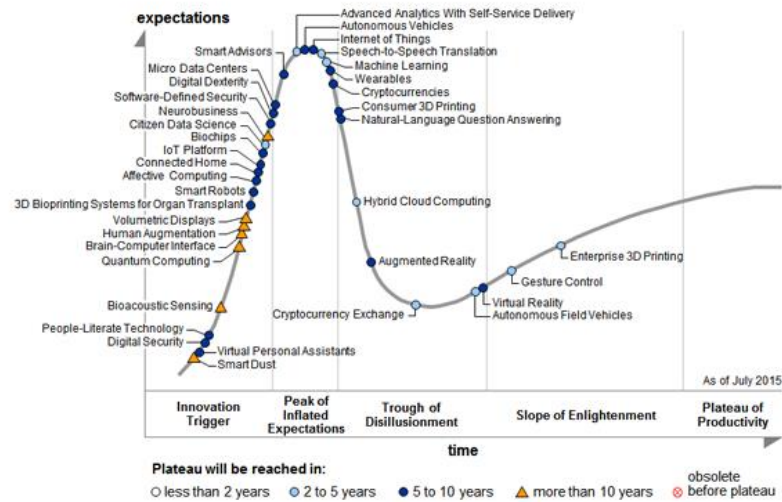
	PERCENTAGE LOSS WRT TO TIMELINE WITH VARIOUS TOPOLOGIES CHAIN (TOP), GRID (MIDDLE) AND DUMBELL (UNDER)	
<b>Figure 5.3</b>	SAMPLE PACKETS RECEIVED/ TOTAL NUMBER OF PACKETS SEND A COOJA SIMULATION	37
<b>Figure 5.4</b>	COMPARISON OF GOODPUT (USEFUL PACKET RECEIVED/TOTAL NO OF PACKETS SEND) OF VARIOUS CONGESTION CONTROL MECHANISM WITH DIFFERENT TOPOLOGIES.	38
<b>Figure 5.5</b>	CALCULATED RTO VALUES OF SIMPLE CoAP, DEFAULT CoAP AND COCOA PLOTTED AGAINST TIME FOR TRANSMITTED PACKETS WHEN RTT IS MEASURED	41
<b>Figure 5.6</b>	RTT COMPARISONS OF VARIOUS TOPOLOGIES WRT THEIR ALGORITHMS	43

### 1.1 Internet of Things (IoT)

The term “Internet of Things” (IoT) is used to connect various physical objects and that are embedded with the sensor and can to connect through internet for remote access, interactive integrated services, and management. It has been estimated at the end of 2020 more than 26 billion devices will be connected in with wide range of application for communication [1] These devices will be connected by “Unique Address” which is a unique way to identify the objects by sensors, actuators, Radio Frequency Identification (RFID), barcode, LAN etc. Thus, it creates a new application area for Internet – Connected automation, which is expected to generate vast amount of data from various location that are needed to be quickly aggregated to have better throughput, predictability, and reliability of the network.

In Internet of Things (IoT) constrained devices plays a crucial role for the connectivity in IPv6 network. Due to the limited hardware and communication capabilities there is a design protocols and standards in the IOT environment. Thus, Internet Engineering Task Force (IETF) is working on the standardization of various layers of the IoT protocol stack that consists of the constrained IPv6 capable devices. As a result, it designs IPv6 Routing Protocol for Low – Power and Lossy Network (RPL), IPv6 over Low – Power wireless Personal Area Network (6LoWPAN) and for application layer the Constrained Application Protocol (CoAP) was developed.

It is predicted that IoT has the potential that to increase global corporate profit up to 21% by 2022. And it has been assumed that there might be 20 billion things that will be connected to the Internet by 2020. according to latest Gartney Hype cycle for emerging technologies IoT has been place at the “Peak” of the cycle as shown below.

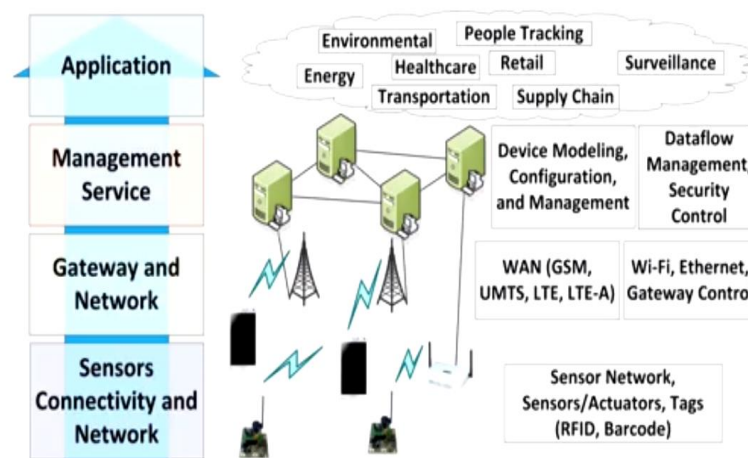


**Figure 1.1: Hype Cycle**

The fundamental thought of IoT is that it will effect on an almost all parts of regular life and the behaviour of potential clients. IoT has influenced industry, research, security, utilities, transportation, healthcare, manufacturing, supply and provisioning and facility management.

## 1.2 IoT System Architecture

Establishing a common ground for IoT standards for the development of the IoT reference architecture is very difficult. As proposed by [2] a Reference Architecture Model (RAM) for IoT provides a model for the communication between many heterogeneous IoT devices and the Internet as a whole.



**Figure 1.2: IoT Architectural Layers**

Each device in the network is consists of the IP address which unambiguously identifies it on the internet. The physical layer consists of the sensor devices ex RFID, barcode, Actuators, which collects real time information and use low power and low data rate connection using IPv6 header.

For the private, public or hybrid network model it requires robust and reliable performance. Thus, network models are designed to communicate QoS requirements for latency error, probability, scalability and bandwidth, security while achieving high level of energy efficiency.

Management service layer consists for the information analysis, security control, and process modelling and device management.

In application layer, various small and industry sector use IoT for service enhancement. This can be classified by courage, size, availability, heterogeneity and business model. It includes the areas like personal and homes, utility, enterprise, mobile etc.

## **1.3 Protocols Suited for IoT**

### **1.3.1 Challenges In WSN**

For the working of the IoT devices Wireless Sensor Networks (WSNs) plays an important role. The reconciliation of both WSN and other IoT components give the remote access to the heterogeneous devices that can have the capacity to give the basic and common services. WSN gives the platform for the IoT devices to work but there are some challenges for the WSN in Internet of things like security, QoS, configuration, scalability in[3]. As the IoT devices should accommodate large no of devices it uses IPv6 address space, the application needs larger scale, low power and low bandwidth and the data generated by the IoT devices need a large scale to handle this vast data. Following the few challenges in WSN in the IoT. [4]

### **Security**

Generally, in WSN without internet is however confidentiality, availability and authentication depending upon the application sensitivity. But opening WSNs to



the internet the attackers would threaten the WSN from anywhere. So, so measurers should be adopting to protect WSN from novel attacks.

### **Quality of Service**

Quality of Service (QoS) management is taken by optimizing the resources utilization by all the heterogeneous devices. Thus, to show the availability of resources to the nodes, some novel approaches should be ensured to avoid delay and loss guaranteed.

### **Configuration**

Apart from the above two the sensor nodes also required the WSN configuration so that the administrator can ensure its scalability and also the capabilities of detecting the faulty nodes in the network. Thus, to maintain the configuration and management the user is expected to install application and recover it from crashes.

#### **1.3.2 Standard Protocols for IoT**

So, due to the extensive protocols overhead against memory and computational limitation the Internet Engineering Task Force (IETF) has taken some communication standardization protocols suited for the IoT devices includes.

##### **i. For Lower Layer**

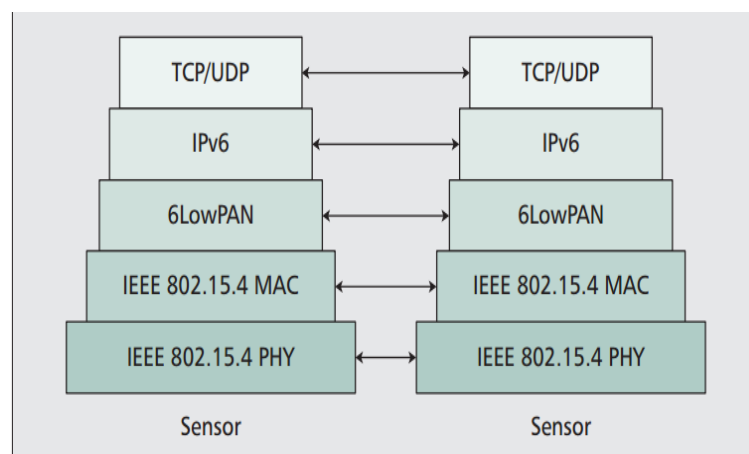
#### **IEEE 802.15.4**

IEEE 802.15.4 applies on both physical and MAC layer. It is a radio based technology standards for applicable to low power and low data rate application with the radio wave rate application with the radio wave coverage within few meters. Due to the above features the IETF develop low power stacks based on IEEE 802.15.4 such as WirelessHART and ZigBee. But they have some limitations in it WirelessHART and ZigBee as it is unlike 6LoWPAN it is not power saver the nodes deployed in the networks, it does not remain in sleep mode, it cannot communicate with other protocol, it has limited range problem and also have fixed data rate.

## 6LoWPAN

Using IEEE 802.15.4 a resource constrained sensor network is used to optimized IPv6 network was deployed to the MAC and Physical layer which uses IP connectivity namelt IPv6 over Low-Power WPAN (6LoWPAN). Its key feature is that it provides wireless communication, universality, stability and extensibility.

The main reason to adopt this is that IEEE 802.15.4 has frame size of 127 byte but IPv6 has minimum transmission value is 1280 byte. Thus it give possible solution to the limited channel capacity, energy scalability, and traffic diversity.

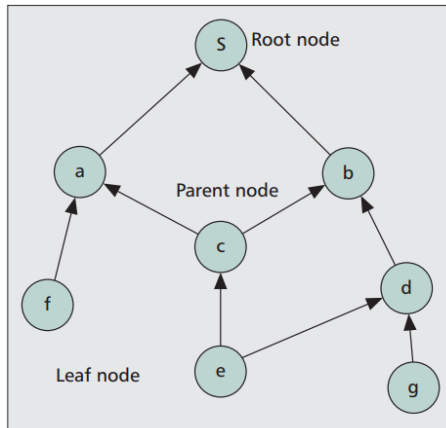


**Figure 1.3:** View Of 6LoWPAN In IPv6 Stack Protocol

### ii. Network Layer Protocols

In February 2008, the IETF routing our Lossy and Low Power Network (ROLL) has up bring to standardization of the IPv6 routing protocols for Lossy and Low-power Network (LLS).

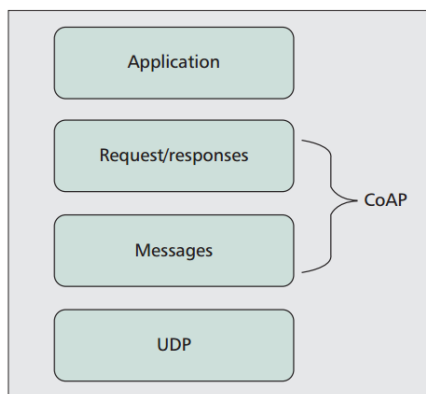
RPL-IPv6 routing protocols for Low Power and Lossy networks. This protocol comprising for constraint devices in power, computation capability and memory. Thus, this protocol is applicable to the networks that are unreliable and have low data rate but high loss rate. RPL is a distance vector routing protocol, in which nodes is constructed by a destination oriented acyclic graph (DODAG). But current RPL has some issues like End-to-end throughput, packet recording, and input of duty cyclin, multi-topology routing.



**Figure 1.4:** RPL Routing Tree: DODAG

### iii. Application Layer Protocols

Constrained Application Protocol (CoAP) this type of routing protocols is applicable on the application layer and it has the ability of translation the HTTP manage so as to integrate with web services. This protocol applied for the multicast with little overhead.



**Figure 1.5:** CoAP Protocol Stack

It thinks about of the Representational State Transfer (REST) Style. It records every one of the resources in the system and relating to every resources a Unique Universal Resource Identifier (URI) from which resources can be picked operation is stateless like GET, PUT, POST, DELETE or so on.

#### 2.1 Constrained Application Protocol for Internet of Things

“The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the **Internet of Things**. The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation.” [5]

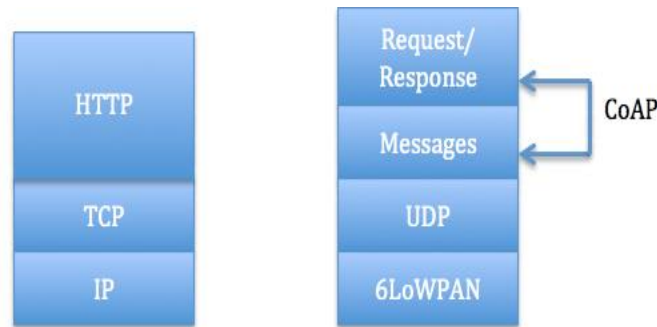
Our new generation technology is rely on the Internet so it Internet of Thing (IoT) is the new generation technology that evert object is connected to the internet no matter wat. There are numerous remote conventions (like IEEE 802.11 Series, 802.15 Series, Zigbee, and so forth) for correspondence between gadgets. Nonetheless, considering a ton of small devices can't communicate properly with constrained devices, so, Internet Engineering Task Force (IETF) has built up a lightweight convention protocol: Constrained Application Protocol (CoAP). [6] [7]

##### 2.1.1 CoAP VS HTTP

CoAP uses the similar feature like HTTP but it is a network-oriented protocol, as though it uses the HTTP based protocol but along with it also allows for overhead, multicast etc. HTTP is the one of the easily handled and managed protocol for the connection between the devices using its various tags and long term standard protocols. As we know that HTTP helps in the better communication between the all of the IoT devices commence over the application layer. But there is one disadvantage of the HTTP protocol that it does not suitable for the notification push services, as it is a point to point (P2P) communication model. Also though HTTP is too complex for the constrained devices[8].

As we know that HTTP works over the TCP layer and it have a complex congestion control. But CoAP works over the UDP layer. To access the internet

resources the CoAP architecture is based on the REST design. To overcome the lack in the constrained devices, the length of the datagram should be optimised and have a reliable communication among them. The working of the CoAP include the URI method, RESTful style which includes GET, POST, PUT and DELETE methods. Thus, it have to be managed over the constrained devices it uses lightweight UDP protocol. For the commence of the group communication CoAP allows IP multicast for the IoT devices. Underneath shows the HTTP and CoAP protocol stacks.



**Figure 2.1:** HTTP and CoAP Stack

## **2.2 Introduction to Enhancing Application Layer Performance Through Congestion Control in CoAP Based Internet of Things**

Internet of Things is a novel way to connect people, things, data and application through internet to enable them for remote access, interactive integrated services and management. It is a unique way to identify the objects in the coming future by “Unique Address” in the networking field of computer science, such as by sensors, actuators, Radio Frequency Identification (RFID), barcode, LAN etc.

For the standardization of the application layer protocol in IoT Internet Engineering Task Force (IETF) has introduced Constrained Application Protocol (CoAP) which function on the architecture of REST (Representational State Transfer). In CoAP has a basic congestion control protocol for the reliability that based on the retransmission timeout (RTO) together with exponential RTO backoff to deal with the congestion in the network.

CoAP is known as Representational State Transfer style (RESTful) protocol over the application layer which helps in the deployment of the connection between

the clients and servers over the Internet. This mechanism is done by the help of sensor deployed in the devices or actuator states. This protocol is implied on UDP layer by default which is although did not provide end-to-end reliability which is needed for the applications. For this CoAP introduces end-to-end acknowledgements (ACKs). It is done by setting a confirmable (CON) flag in an outgoing CoAP message from the destination node for end-to-end ACK. In CON message, it is designed for the reliable communication. They reply with ACK and retransmits ACK without 2-3sec if the sender does not receive ACK before expiration of the timer the time is doubled for every consecutive failure. The second one is if it does not care about the message failure and sender do not wait for ACK the Non-Confirmable (NON) message is send. This scenario can be seen in implementation of real time and multicasting data transfer.

### **2.3 How Congestion Control Is Different in IoT Than WSN?**

When we talk about standalone WSNs (not connected to Internet), the traffic is less and mainly contributed by up-link (nodes to sink) traffic only. However, down-link (sink to nodes) traffic also occurs in query based applications but it does not contribute much to the traffic and congestion, since in standalone WSNs, sensor nodes' main task is to sense and route the data towards sink.

On the other hand, when WSN is connected to Internet (through Gateway), many real-time Internet applications open their way to the sensor network through Gateway node. In this scenario, both up-link and down-link traffic can be observed in equal quantity since many application users sitting on the other side of WSN may directly log in to any sensor node and execute the required task. This whole process increases the network traffic significantly which leads to congestion. That's why many in-network processing techniques are being used to reduce the packet transmissions inside WSN. Saving resources such as energy is the prime concern of the researches in industry as well as academics.

Further mostly transmission control protocol users transport layer for reliable communication. But due to short communication in IoT, TCP is not sufficient. As

TCP needs more time for setting up connection control but the volume of data transfers is very low in IoT, thus TCP is impractical for IoT works on two principles.

- i. The various large amount of data is transmitted at the same time.
- ii. Due to heterogeneous network IoT nodes moves onto other networks, thus increase in the network node.

**Table 2.1:** Difference Between Congestion Detection Between IoT and WSN

<b>Parameters</b>	<b>IoT</b>	<b>WSN</b>
<b>Property</b>	WSN + Internet + Cloud Storage + Mob/Web Application	Things Connected to Wireless Network And Gather Data Or Monitor The Environment By Sensor Nodes
<b>Interaction Layer</b>	Application Layer	Transport Layer
<b>Application Layer Protocol</b>	UDP	TCP/UDP
<b>Mechanism</b>	Various Back-Off Algorithm at Coap	Routing Schemes with Centralized Strategy, Dedicated Scheme with Distributed Strategy
<b>Gateway</b>	6lowpan	IEEE 802.15.4

## 2.4 Mechanism of Various Congestion Control in CoAP

Work has been done in WSN related to the congestion control mechanism. But existing congestion control protocols cannot be directly applied to the CoAP in IoT as they are lack in reliability, not able to flow control in UDP (User Datagram Protocol and devices are restricted due to its Low-Power and Lossy networks. So, in this section congestion control protocols are presented which are reliable to the application layer to the CoAP.

### 2.4.1 Traditional CoAP congestion control with exponential Back-off

Traditional protocol is not applicable on the low data volume application protocols such as CoAP. Here RTT (Round Trip Time) is used for the calculation of RTO (Retransmission Time Out). In it upto conservative fixed value of 3sec when no RTT estimated can be obtained, controlling the transmission behaviour by not sending more than one UDP datagram per RTT on average to a destination, detecting packet loss and exponential back-off the transmission timer when a lost event is occurred[9].

In the back-off algorithm CoAP must track two endpoints first for each confirmable message it sends, the track should be maintained for its acknowledgement (or reset) known as timeout and a counter is maintained if there is any retransmission of message. Here an initial ACK\_TIMEOUT is marked for a random duration between [ACK\_TIMEOUT] and [ACK\_TIMEOUT \* ACK\_RANDOM\_FACTOR]. In this case when time out is occurred and retransmission counter is less than MAX\_RETRANSMIT the retransmission timeout is incremented and time out is doubled. If MAX\_RETRANSMIT is reached before the acknowledgement, then the message is cancelled and gives the information of the failure otherwise the transmission was considered to be successful [10].

The main problems that can arise in the tradition congestion control may be case of the accidental collision or the cross-layer detection between the requestor when sender may be cancel the confirmable message ACK even before the MAX\_RETRANSMIT counter value has reached. In this case receiver, may elicit some different ACK for the message which has not been detected. Here the ACK was lost and there is no need for the retransmission of the request[11].

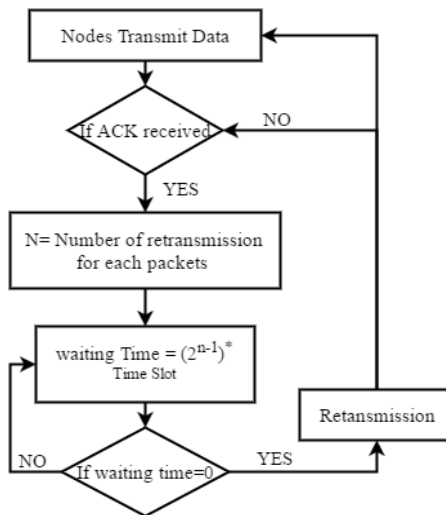
#### **2.4.2 Adaptive Backoff Algorithm**

Traditional internet protocol is not directly applicable to the IoT as due to its resource limitation of the devices. For this Binary Exponential Backoff or Truncated Binary Exponential Backoff algorithms are used for the retransmission of the missed congestion in the networks [12].

#### **Exponential Backoff Algorithm**

In this algorithm whether the collision over between two nodes it takes sometimes to retransmit. Then a random value(k) is taken from (0 .....  $2^n-1$ ) when waiting time calculate by  $k \cdot T_{slot}$ . When slot is the packet size. As in the algorithm whenever the retransmission attempts increase exponentially. But there is a drawback in this algorithm as after each collision window size increases which also increases the waiting time of nodes to access the channel and there is also loss of channel bandwidth.

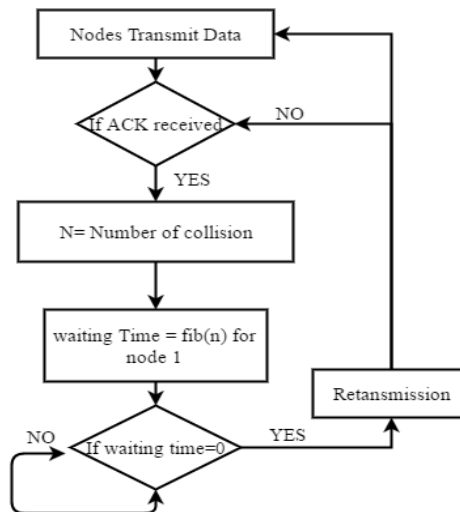




**Figure 2.2:** Flow Chart For Exponential Backoff Algorithm [12]

### Fibonacci Backoff Algorithm

The Fibonacci equation  $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$  and  $\text{fib}(0) = 0$ ,  $\text{fib}(1) = 1$  where  $n \geq 0$ . Here in it the waiting time increase slowly but as when there is more collision between the IoT nodes the increment factor is decreased. Here drawback of this algorithm is that as the timeout value is small which led to unnecessary retransmission within the nodes.



**Figure 2.3:** Flow Chart for Fibonacci Backoff Algorithm [12]

The conclusion and outcome of these algorithm are in Exponential Backoff Algorithm rapidly increases for the packet delay if there is low traffic in the networks. But for Fibonacci Backoff Algorithm it slowly increases for packet delay if there is

low traffic. Thus, according both the algorithms the change according to the traffic load and this phenomenon is termed as Adoptive Backoff Algorithm.

### **2.4.3 Effective Steaming Over CoAP**

The main purpose of this scheme is that it handles the packet losses due to accidental collision or channel error and reduces the effect unnecessary back-off delay. Here the efficiency of the packets sends in the form of block over the CoAP layer (Web Based Layer). In the streaming scheme the data packets which are send over lossless and lossy networks.

In the lossless network, each packet is send over the network one by one (up to 8 packets). If some packets are lost, then the particular packet is retransmitted for the recovery of the loss packet. In Lossy networks the multiple data blocks are retrieved by a single request by a single request message which overcomes the inefficiency of the stop and wait content. These two paradigms can be implemented in the RESTful architecture by the confirmable ACK and the Non-Confirmable ACK scheme[13].

The proposed Block Request and Recovery (BRR) mechanism works with the multiple blocks. The multiple block is request with the one block request (BREQ). If the network is not congested then multiple block are retrieved by a single BREQ request. This will enhance the throughput of the network. But in case of congested network retrieving multiple block with a single BREQ request will make the channel busy for longer time without increasing throughput. So, BREQ should be determined carefully.

### **2.4.4 Alternate Congestion Control Algorithms**

Default CoAP congestion control protocol reliability based upon retransmission timeout (RTO) mechanism and Exponential Backoff Algorithm is used but for more advance algorithm is used but for more advance congestion control a new alternative congestion control a new alternative congestion control algorithm is proposed known as CoAP simple Congestion Control/Advance (CoCoA) Its mechanism is simple but although it strives for higher performance. It uses TCP RTO

estimation algorithm that uses Round Trip Time(RTT) measurement for RTO estimation[14].

In this paper evaluates the performance of the default CoAP for congestion network with the proposed alternate CoCoA RTO was updated by measured RTTs value. The following are the RTT based algorithms[15].

### **TCP based RTO algorithm**

In this algorithm, RTO value is calculated using smoothed average of the RTT(SRTT) and RTT variation (RTTVAR). Initially the value of RTO is set to at least 1. After the 1<sup>st</sup> RTT measurement SRTT value is set as R and value of RTOVAR is set as R/2. Thus, RTO is calculated as

$$RTO = SRTT + \max(G * k * RTTVAR) \quad (1)$$

Where G= clock granularity

$$K=4$$

On subsequent updating of RTT we update value of RTO and get R', SRTT and RTTVAR value from the following equations

$$SRTT = (1-\alpha) * SRTT + \alpha * R' \quad (2)$$

$$RTTVAR = (1-\beta) * RTTVAR + \beta * |SRTT-R'| \quad (3)$$

When RTO timer expires the value of RTO is doubles on the subsequent retransmissions.

### **CoCoA Algorithm**

CoCoA runs two RTO estimators for estimation which are: a strong estimator which uses ACKs of the original transmissions and a weak estimator that uses ACKs of transmissions. Following are the equations to find overall RTO estimators for strong and weak estimators [11], [16], [17].

$$RTO = 0.25 * E_{weak} + 0.75 * RTO \quad (4)$$

$$RTO = 0.5 * E_{strong} + 0.5 * RTO \quad (5)$$

Where initially value of RTO is 2sec

CoCoA uses variable backoff factor depending upon estimated RTO but for TCP it uses fixed backoff factor which is 2. If value of RTO is between 1sec and 3sec the backoff factor is 2 i.e. the whole equation of RTO is multiplied by 2. Below 1sec backoff factor is 3 but when backoff factor is above 3sec backoff factor is 1.5[18].

For the conclusion of the CoCoA and Two TCP-based Congestion algorithm known as alternative congestion control mechanism proposed by IETF. The result show that Default CoAP algorithm is better in higher time completion time that Alternative Congestion Algorithm but it is more scalable and efficient that Default CoAP in high congestion network.

#### **2.4.5 CoCoA (Advance Congestion Control Protocol CoAP)**

There is a restriction in the congestion control mechanism in UDP based application such as in CoAP due to its Low Power and Lossy Networks CoCoA for CoAP deals more efficiently to handle the congestion in the network. For the confirmable messages and ACKs it uses RTTs (Round Trip Times) for the packet exchange in CoAP between two motes i.e. sensors motes to calculate parameterized Retransmission Time-out (RTO) that is calculated over time which depends on the RTT value calculated by the motes. Some experiments in paper [14], [15] for the evaluation of parameters to distinguish between different congestion control algorithms in IoT. The dynamically used and approached congestion control mechanism is standardized by IETF is CoAP and CoCoA algorithms which is further discussed below [17].

Here in this paper [19] a comparative study of CoAP and CoCoA for congestion control Protocol is performed. For the experiment Contiki operating system was used a firmware for the sensor motes and implemented in the Cooja simulation framework. For the evaluation, a basic grid topology of 18 motes or clients sending data to the data collector or server and a router which helps to connect

the sensor network within the internet. RPL was setup before the experiment was performed and for the motes connectivity its waits for 180sec. Various traffic was generated from 0.5 sec to 60sec to study the behaviour of the algorithm in lossy scenarios. In this scenario of no wireless lose with different packet frequency there is a same throughput seen in CoAP and CoCoA. But with packet with frequency 2packets/sec CoCoA has higher throughput. Further in case of lossy environment throughput of CoAP is better than CoCoA it is due to losses are proportion to distance. But for retransmission CoCoA performs better than CoAP.

In [20] Betzler et al. presents its work by evaluating the CoCoA performance over congestion control on CoAP under various topologies and compare the default CoAP congestion mechanism with the CoCoA parameters. As a result, CoCoA performs better than default CoAP in terms of throughput in constrained network. Extended its result further in [21] Betzler et al. over come the problems in CoCoA and named it as CoCoA+[22]. Which is more stable version of CoCoA algorithm.

Further in [19], [23] study on CoCoA for congestion control mechanism was performed. For experiment Contiki operating system was used with Cooja Framework. It calculates strong and weak RTO value. AS in CoCoA for estimation RTO of a Strong estimator is used which ACKs of the original transmission and a Weak estimator which uses ACKs of the transmissions.

Further by changing parameters of CoCoA like variable backoff factors, backoff thresholds, and initial and maximum RTOs, new scheme was proposed known as CoCoA-Fast (CoCoA-F). As a result, in lossy network it gives throughput is same, retransmission increases compared to CoCoA but throughput of CoAP is still more [24][25].

For more analysis to distinguish between wireless losses and congestion losses a 4-State Estimator scheme is proposed which is described underneath

#### **A Four- State Estimator Scheme**

In this scheme, CoCoA performs more conservative while wireless losses as it evaluates the congestion problem in the network and then apply the backoff algorithm

according to the variable backoff factor. Thus, it sends lower no of packets for the consequent retransmission and less packet losses were occurred.

Firstly, we have CoCoA Strong which ignores the first loss occur in the network. If there are more subsequent losses in the network, then it applies backoff algorithm. This scheme sometimes leads to more congestion in the network due to its aggressive network behaviour. Thus, when the failure was intermitted in the wireless network the lower percentage of obtained RTO value must be estimated. Thus, this will give the parameter for the decision. If more subsequent loss were occurred greater percentage of final RTO value is included. Which was calculated by the following equation. Here  $\omega$  is the weight of the obtained RTO to calculated the overall RTO value.

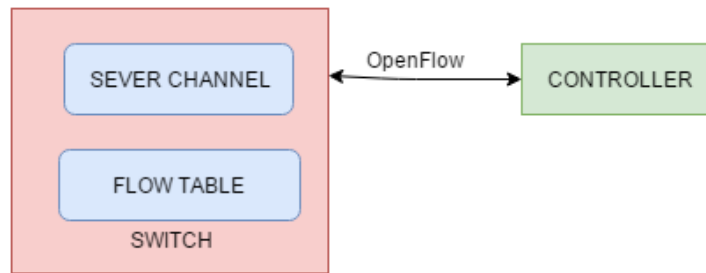
$$RTO_{\text{overall}} = \omega * RTO_{\text{obtained}} + (1-\omega) * RTO_{\text{obtained}} \quad (6)$$

Thus from this we can conclude that the RTO value measured in CoCoA can lead to dispropornate RTOs. To remove this disproporanate the actual RTT value was measured from the last retransmisstter or by matching the ACK to the retransmission. Thus, to distinguish between the wireless losses and congestion losses.

#### **2.4.6 Congestion Avoidance algorithm in SNS Environment**

As the no of devices increases in the IoT network there is a burst of traffic which cause a congestion in the network. Here in this paper [26] a new flow congestion algorithm has been proposed which is controlled by the network in SDN (Software Defined Networking) environment. The main purpose of congestion control by this protocol is link utilization which is calculated by the SDN controller and then OpenFlow configuration protocol is applied to switches by recalculated rerouting algorithm.

OpenFlow has the logically centralized controller that holds the single control plane for the network. Many a device containing only data planes will respond to the command given by centralized controller. All the network paths in the network and command are described by the OpenFlow controller. Thus, this paradigm was implied to SDN environment to simplify and optimize the network.



**Figure 2.4:** Connection Between the Control Plane And Data Plane

**Table 2.2:** Output Evaluation of Different Congestion Algorithms

Protocols	Output evaluation
Default CoAP	--
Effective Streaming	Latency is Under 1.2 sec and Throughput is 32% enhancement than Default CoAP
Exponential Backoff Algorithm	Algorithm rapidly increases for the packet delay if there is low traffic in the networks
Fibonacci Backoff Algorithm	Algorithm slowly increases for packet delay if there is low traffic
CoCoA	Higher throughput than default CoAP
CoCoA- F	Throughput is same, retransmission increases than CoCoA, throughput less than Default CoAP
4-State Estimator scheme	CoCoA 4-state-Strong achieves 35-60% higher throughput as compared to the default CoCoA scheme, with only 20% more
Flow control algorithm	Minimize the high congestion by reset the flow

**Table 2.3:** Comparison of Different Congestion Control Parameters

<b>Protocol</b>	<b>Congestion Notification</b>	<b>Congestion Control</b>	<b>Evaluation Type</b>	<b>Evaluation Parameters</b>	<b>Compare With</b>
Tradition CoAP	--	Back-Off Algorithm	Simulation	--	Default
Adaptive Backoff Algorithm	Serial Port and Hyperterminal Window	Exponential Backoff Algorithm	Arduino Board	Throughput and Mobility Speed	Adaptive Backoff Algorithm
Adaptive Backoff Algorithm	Serial Port and Hyperterminal Window	Fibonacci Backoff Algorithm	Arduino Board	Throughput and Mobility Speed	Adaptive Backoff Algorithm
Effective Streaming Algorithm	Block Wise Transfer	Multiple Data Blocks	Simulation Ns3	Latency, Throughput	Default CoAP
Alternate Congestion Control Algorithms	RTO Estimators	Cocoa and Two TCP-Based Congestion Algorithm	Netem Linux Emulator and Californium	Client Completion Time, Scalability, Efficiency	Default CoAP
Cocoa	RRT	Two RTO Estimator Algorithm (Strong, Weak)	Contiki Os Cooja Simulation	Throughput in Lossy And Lossless Environment	Default CoAP
CoCoA-F	RRT	Cocoa Algorithm	Contiki Os Cooja Simulation	Throughput in Lossy Environment	CoCoA and Default CoAP
4-State Estimator Scheme	Variable Backoff Factor	Cocoa-Strong Algorithm	Contiki OS Cooja Simulation	Distinguish Wireless Losses And Congestion Losses	--
SDN Environment	Using VLC Player Application Size 10k Byte Avg Bitrate 240kbps	Flow Control Algorithm	Mininet And Onos With Wireshark	Throughput, RTT Avg,	Legacy Network



## CHAPTER 3

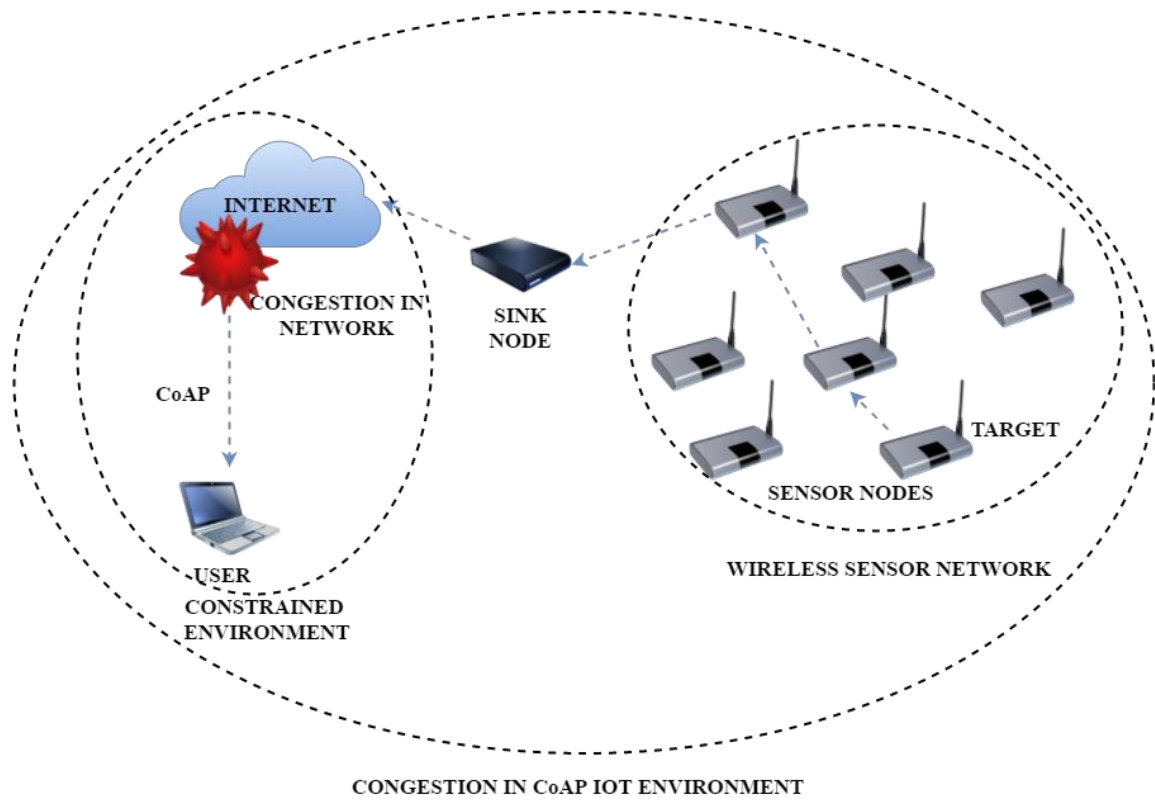
### SCOPE OF STUDY

---

CoAP based systems are designed for simple electronic devices that allow them to interact over internet. However, targeted low power device might not be capable enough to handle large no of requests and data loses. The main objective of network is to have reliable communication but in a congested network a reliable retransmission of ACK is necessary. So, there arise many congestion control problems like accidental collision or cross layer detection between sender and client or when the MAX\_RETRANSMIT counter value reached and the message ACK is not reached. So, taking all this into account backoff algorithm like exponential backoff algorithm and Fibonacci backoff algorithm developed. But these algorithms work according to the traffic load there is not accuracy in them. Further there is a problem in implementation of CoAP over streaming services for the multicast and asynchronous message exchange. Thus, retrieved multiple data blocks by a single message that are loss due to accidental collision and channel error tis somehow reduce backoff delay the congestion.

All these approaches minimize congestion in a network but for more reliability in a congested network new CoAP Congested Mechanism Approach (CoCoA) was proposed. It uses a variable RTO estimator for the estimation of the ACK message to reduce congestion in the network. It was compared with default CoAP control mechanism. As a result, it performs better than Default CoAP. By changing parameters of CoCoA like variable backoff factors, backoff thresholds, and initial and maximum RTOs, new scheme was proposed known as CoCoA-Fast (CoCoA-F). in this it gives throughput same but retransmission increases as compared to CoCoA but throughput of CoAP is still more. For more analysis to distinguish between wireless losses and congestion losses a 4-State Estimator scheme is introduced. Which reduces deprotonate RTT value. Further we will be enhancing CoAP for reducing latency, data losses by adding request handling mechanism which will terminate and

reestablish request connection in case if device is too busy and a re-transmission mechanism will be implemented to reduce lags and back-off delays in request processing. This may improve application performance in congested network.



**Figure 3.1:** Detection of Congestion in Application Layer of CoAP IoT Environment

4.1 Problem Formulation

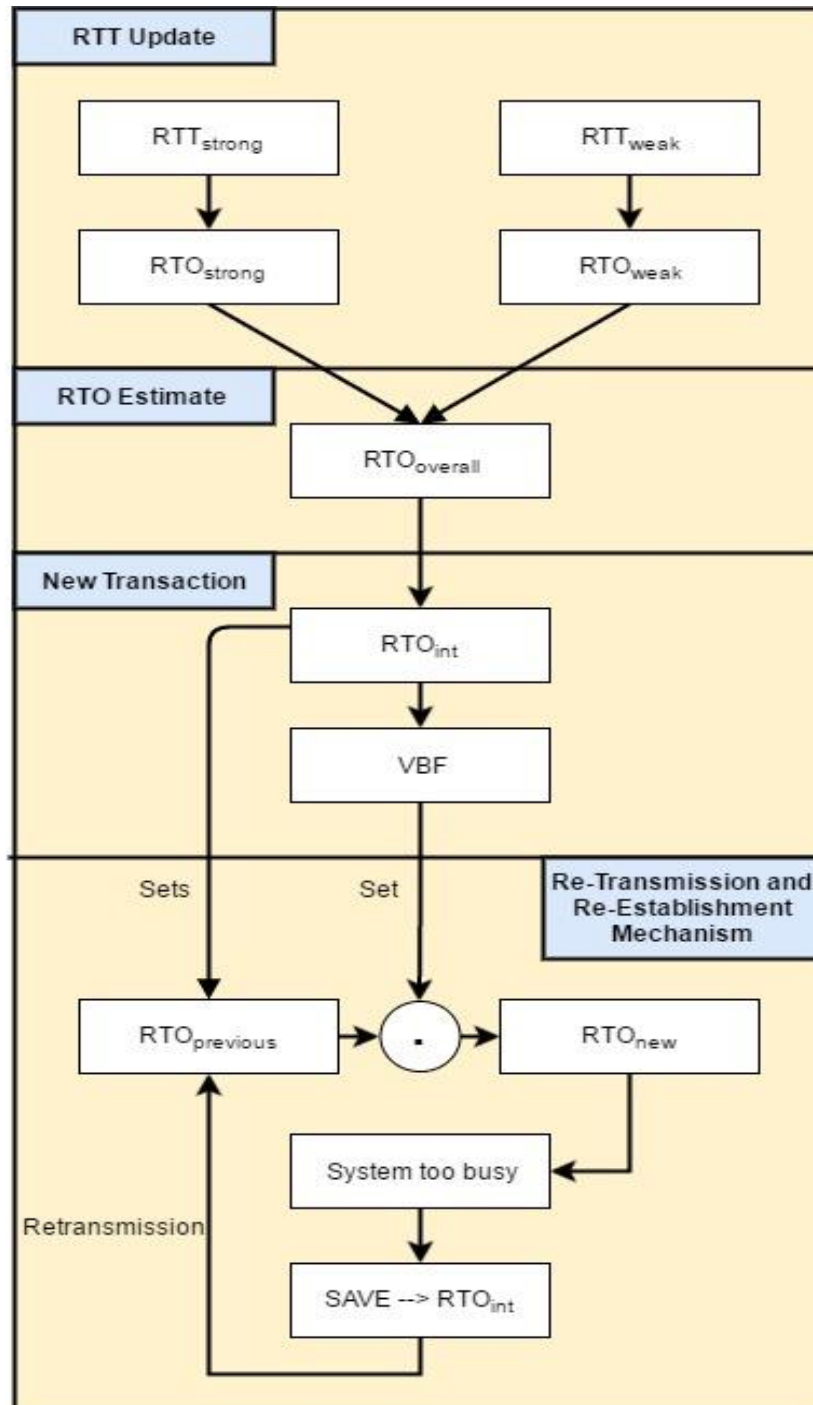


Figure 4.1: An overview of the different RTO variables used in Congestion Control in CoAP

A lot of work has been done on congestion control in wireless Ad-hoc and sensor network. But, due to Low- Power and Lossy Network in the constrained environment we must restrict our self to the congestion control over the UDP -based application layer protocol.

In the CoAP based constrained network Eggert [9], highlight the need of congestion control mechanism over the UDP-based application layer protocol. But later in [10], Eggert and Fairhurst concluded that the traditional IEFT congestion control mechanism are not feasible with application protocol such as in CoAP due to its low data-volume. The concluded that when no RRT value is estimated (in unidirectional communication) within 3 sec or when RTT value is estimated for any destination there should be control over the transmission by not sending more than one UDP packet per RTT on average. So, that it would detect packet loss and will calculate exponentially backoff the retransmission when there is loss in the network.

In RFC 7252 [11], is the traditional CoAP congestion control with exponentially Back-off. Here in exponentially back-off algorithm acknowledgement (or reset) known as [ACK\_TIMEOUT] and a counter is maintained if there is any transmission of message. If time out occurs along with retransmission counter is incremented and [ACK\_TIMEOUT] is doubled or the message is cancelled then it gives the message of failure.

For more Advance Congestion Control a new alternative congestion control mechanism was proposed [14]. Which uses RTO estimation algorithm that uses Round Trip Time (RTT) measurement for RTO estimation. There are two algorithms that calculate RTO when known as TCP-based RTO algorithm [15] and as CoAP Simple Congestion Control/Advance (CoCoA) Algorithm [17] [20]. When at last overall RTO value was calculated and averaged over the previous RTO value and obtain the last RTO estimated which being over weak or strong on is calculated.

So, for further research the enhancement of the CoCoA has dome by adding Re-Establishment mechanism that will terminate the transmission after n number of retransmission and save the packet unless the CoAP server is free. It also uses VBFs for adaptive RTO calculation from Strong RTTs and Weak RTTs and uses aging mechanism to optimise performance.

## 4.2 Objective of the Study

To Enhance application layer performance through congestion control in CoAP based on Internet of Things, the main objective of network is to have reliable communication but in a congested network a reliable retransmission of ACK is necessary. So, there arise many congestion control problems like accidental collision or cross layer detection between sender and client or when the MAX\_RETRANSMIT counter value reached and the message ACK is not reached. There are many algorithms that reduce congestion in network up to some extend but to minimize congestion in a network for more reliability a new CoAP Congested Mechanism Approach (CoCoA) was proposed.

The main objective of this research work is to implement congestion control in application layer of CoAP based IoT are as following:

- i. To compare existing congestion control mechanism based on CoAP.
- ii. To propose a new CoAP Congested Mechanism Approach (CoCoA) to enhance reliability in a congested network.
- iii. To evaluate the performance of proposed approach in terms of loss percentage of packets, goodput, RTT and RTO calculation it in Cooja emulator.

## 4.3 Research Methodology

CoAP based systems are designed for simple electronic devices that allow them to interact over internet. However, targeted low power device might not be capable enough to handle large number of requests and thereby reducing delays and data losses. We will be enhancing CoAP for reducing latency, data losses by adding request handling mechanism which will terminate and re-establish request connections in case it device is too busy. Track of requests will be kept in a small structure only for the aborted requests.

We will implement request re-establishment mechanism that will help in reducing lags in request processing. On sensing lagging CoAP requests will be terminated and re-established to revamp resource and to improve performance of the device.

### 4.3.1 Proposed Algorithm

CoCoA runs two RTO estimators: a strong and weak RTO estimator for each destination end point which gets updated when strong RTTs and weak RTTs is calculated respectively. Strong RTTs ( $RTT_{strong}$ ) are calculated after the ACK received of the first transaction of a CON CoAP message. This  $RTT_{strong}$  value is used to update strong RTO estimator ( $RTO_{strong}$ ). Similarly if the ACK received after one transaction weak RTTs is calculated after the ACK received of the further transaction. This  $RTT_{weak}$  value is used to update weak RTO estimator ( $RTO_{weak}$ ).

For better result, we modify the CoCoA by adding Re-Establishment mechanism that will terminate the transmission after n number of retransmission and save the packet unless the CoAP server is free. It also uses VBFs for adaptive RTO calculation from Strong RTTs and Weak RTTs and uses aging mechanism to optimise performance. Following algorithm describe the detail working of the proposed algorithm.

Start

For each transmission of message between server and client in CoAP based IoT network

If first transmission ACK received

Then

Calculate  $RTTVAR_{strong} = (1-\beta) * RTTVAR_{strong} + \beta * (RTTVAR_{strong} + RTTVAR_{strong\_new})$

And

$RTT_{strong} = (1-\alpha) * RTTVAR_{strong} + \alpha * RTTVAR_{strong\_new}$

Where  $\alpha=1/4$  and  $\beta=1/8$

End if

Else if ACK received after at least one retransmission  
upto three transmission

Then

Calculate  $RTT_{VAR_{weak}} = (1-\beta) * RTT_{VAR_{strong}} + \beta * (RTT_{VAR_{weak}} + RTT_{VAR_{weak\_new}})$

And

$RTT_{weak} = (1-\alpha) * RTT_{VAR_{weak}} + \alpha * RTT_{VAR_{weak\_new}}$

Where  $\alpha=1/4$  and  $\beta=1/8$

Then calculate  $RTT_{weak} = RTT_{weak} + K_{weak} * RTT_{VAR_{weak}}$

Where  $K_{weak}=4$

End else if

Now calculate overall RTO value

$RTO_{overall} = 0.5 * RTO_{strong/weak} + 0.5 * RTO_{overall}$

Now next initial RTO value is calculated as

$RTO_{int} \leftarrow [RTO_{overall}, RTO_{overall} * 1.5]$

Else

Retransmission is greated than three

Then

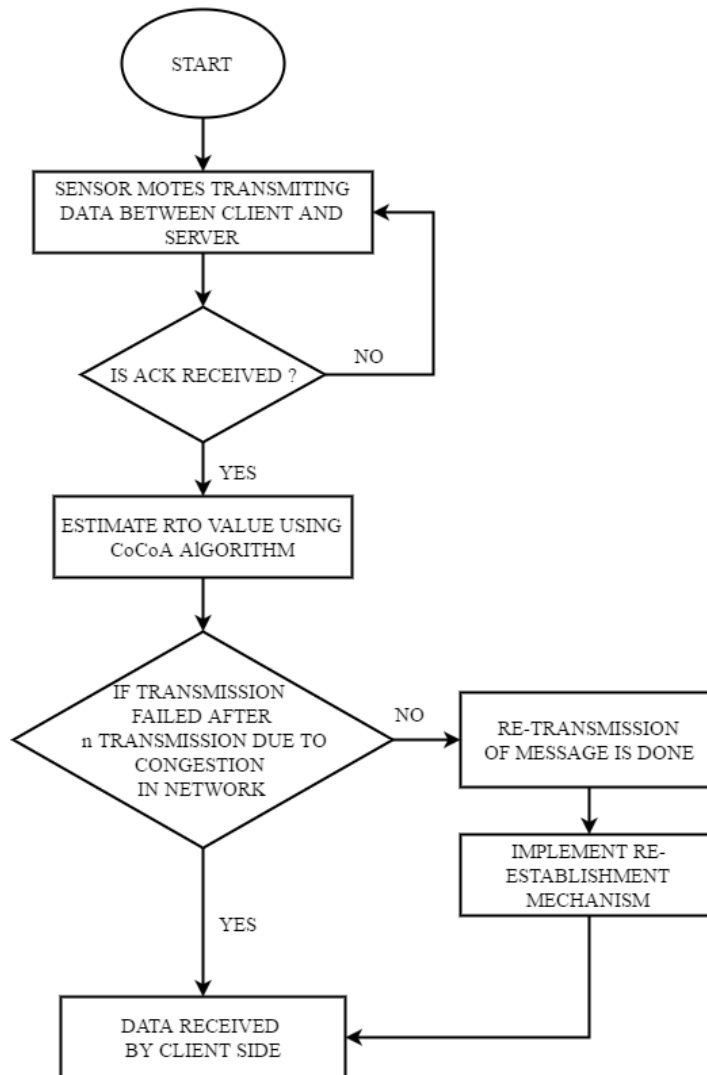
Save  $\leftarrow [RTO_{int}]$

For retransmission time greater than 60sec

Then

Retransmit  $\leftarrow [RTO_{int}]$

### 4.3.2 Flow Chart



**Figure 4.2:** Congestion Control Methodology Flow Chart

### 4.3.3 Tools Used

Contiki comes with a sophisticated Wireless Sensor Network Simulator, COOJA, which is based on real hardware emulation and time-accurate WSN simulation. It can be compiled on linux pc or on virtual environment like VMWare on windows for establishing sensor motes. It is an extensible java based simulator for emulating different nodes like Tmote, sky etc. The code is mostly written in standard C language then upload to target mode. Thus, it make fast and easy for rapid development. Contiki OS supports fully standardized IPv6 and IPv4 internet standards like 6Lowpan, RPL, CoAP etc. Foremost it as a open source software which is



available online easily. Following are the features included in Contiki are mentioned below.

- i. CoAP is a lightweight HTTP protocol that so interaction with the IoT sensors nodes for read and write purpose. Cooja helps to establish a network which is compatible with its protocol.
- ii. Cooja establish RESTful Protocol which uses actions like GET, POST, PUT, DELETE, OBSERVE, DISCOVER in CoAP.
- iii. Cooja helps to establish particular motes (sensors) with IPv6 network.
- iv. Cooja execute CoAP output on browser from where the parameters can be controlled. Example `coap://[aaaa::212:7402:2:202]` or any other sensor mote
- v. Program is written in C language so it is easy to deploy in targeted motes.
- vi. The codes that were executed by the nodes is the exact same firmware which can be later on may uploaded to the physical devices.
- vii. It can simulate motes from smaller to larger network.
- viii. Cooja gives a GUI on which we can observe how various motes interact with each other in the targeted environment.
- ix. On output window, we can observe network simulation, simulation control, notes, motes output, timeline of motes etc.

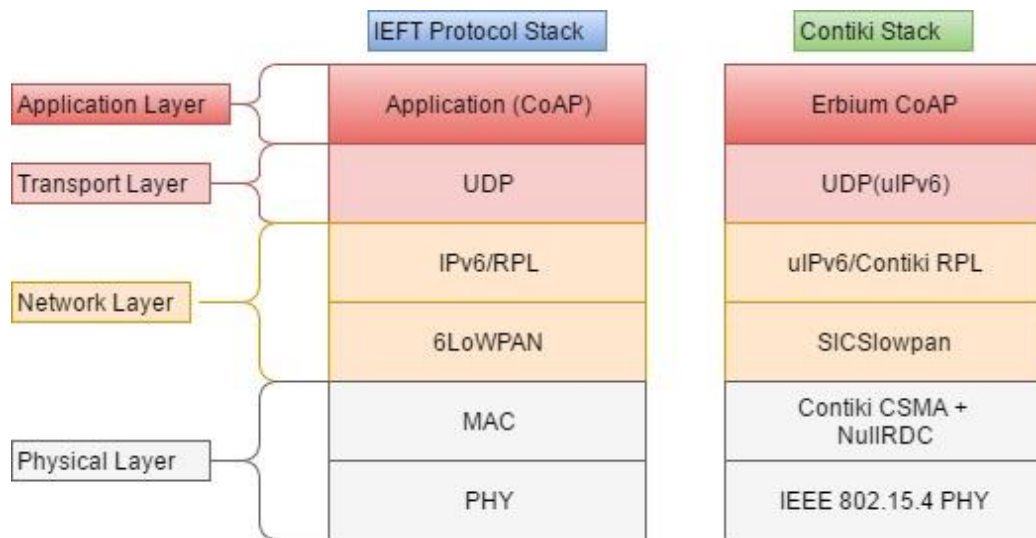
#### **4.3.4 Simulation Setup**

In this section we introduces the Contiki OS and the Cooja simulation[27] along with the over view of the protocol stack used in the simulation. This section also cover configuration of the nodes on which the different scenarios network topologies were tested.

#### **Contiki OS and the Cooja Simulation**

Contiki OS is a open source operation system which implements in constrained devices over the IPv6 network. In figure the difference between the protocol stack of IEFT over IEEE 802.15.4 and Contiki OS is shown in Fig 1. It shows in Contiki OS for application layer for CoAP implementation of Erbium CoAP is used and for the routing of IPv6 network it includes Contiki RPL[28] [29].

In Contiki 2.7 RPL is a part of IPv6 Stack, which causes network formation, packet routing and maintain the network topology. With one or more root nodes formed at the RPL network topology resulting in the Directed Acyclic Graph (DAGs). If it only contains one root then is called Destination Oriented DAG (DODAG). Thus, for communication with another nodes Serial Socket (SERVER) should be open for forming the RPL DAGs. In RPL Object Function (OF) determines which link should be part of RPL-DAG. In this paper we uses Expected Transmission Count (ETX) Object Function which is the default OF [30].



**Figure 4.3:** The overview of IoT architecture layer and difference between IEFT protocol stack (left) and Contiki OS stack (right)

In this paper for deploying nodes in the Cooja simulator several radio medium are there. Radio medium determines how radio signals will propagate in the network. The radio medium used in this paper is Unit Disk Graph Medium (UDGM) which defines the link delivery ration within the transmission range in meter (m) which is 100% and interference range which determine the range upto which distance radio signal can interface with the notes. Interference range is set to be double of transmission range hence 200%. Then to start the simulation initial mote startup delay(ms) is set and along with it the initial seed value (manual or automatic) is determined.

## Sensor Mote Configuration

Cooja provide us the software which is enough to emulate the real sensor node over the real hardware. Thus, it provide the real memory and processing capabilities in the simulation environment which is in real hardware nodes. For simulating CoAP over application layer Cooja provides IEEE 802.15.4 capable node for radio transceiver (CC 2420) for simulation are: Z1 from Zolertia and Tmote Sky from Moteiv which is similar to TesloB mote for real hardware scenario. Following table describe the RAM, ROM and MCU capabilities of the mode used in the simulation.

In this experiment three types of nodes for different algorithms are required: CoAP SERVER (sink node), CoAP Client (source node) and RPL-Border Router. In RPL Border Router node establish a routing table for each node in the network. If a node does not have routing table entry then the neighbouring nodes push itself towards the RPL root until all nodes in the network has destination entry. Thus, to hold the memory of all nodes it requires higher RAM. So, Tmote Sky provide higher RAM than the Z1 mote. Hence, for all the RPL border root node Tmote Sky is deployed.

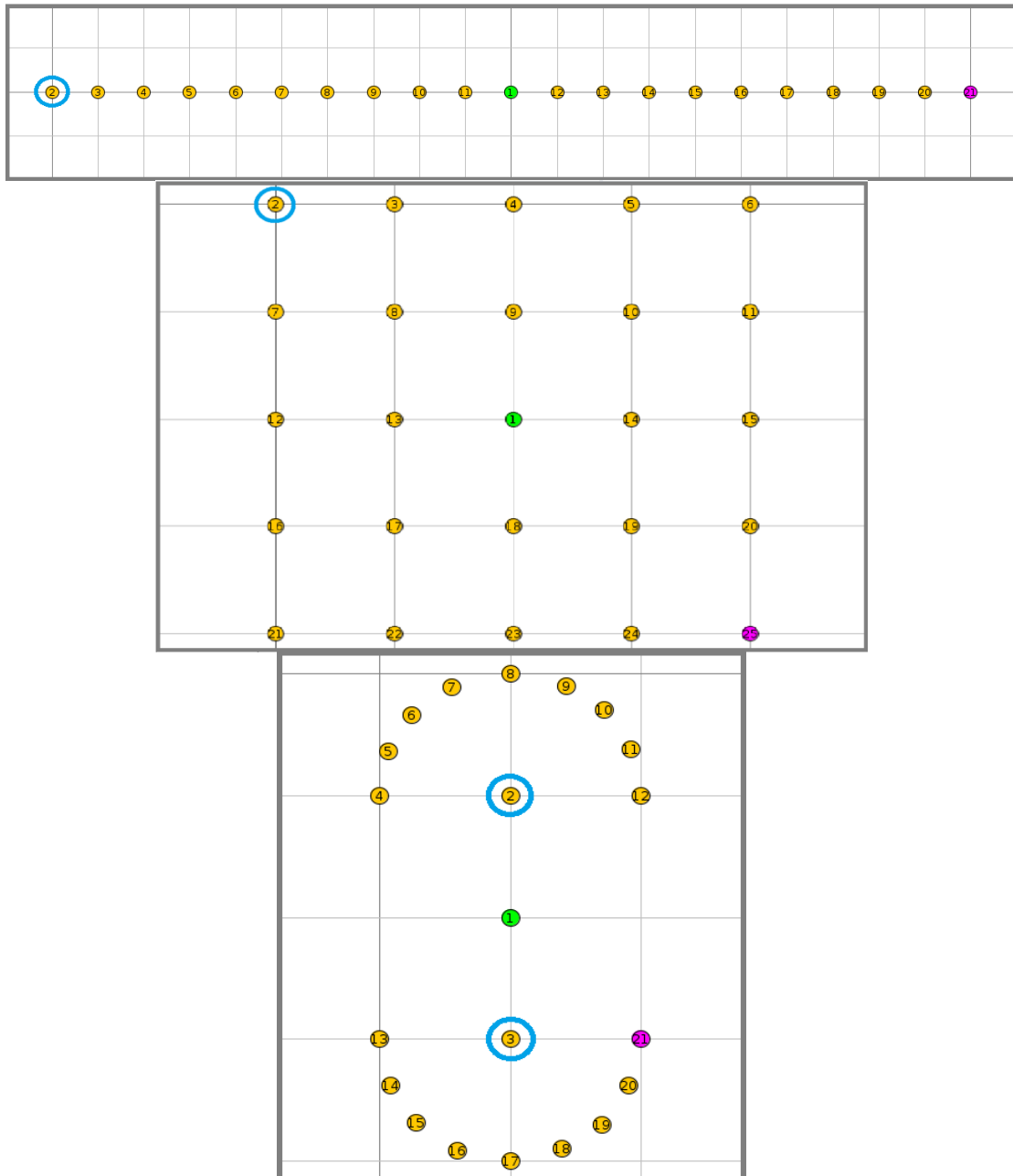
	Tmote Sky	Z1
RAM	10 KB	8 KB
ROM	48 KB	92 KB
MCU	MSP 430F1611	MSP 430F2617
Radio	CC 240	

**Figure 4.4:** Hardware specification of Tmote Sky and Z1 mote

Rest of the node in the network topology runs on Contiki OS stack and Erbrium CoAP (Er-CoAP) is implemented on CoAP version 13. For the Er-CoAP SERVER and Er-CoAP CLIENT Tmote Sky node is used for the implementation of Simple CoAP and Default CoAP, as Tmote Sky has the higher RAM. But for CoCoA algorithm it requires more ROM thus Z1 mote is more suitable for implementation.

## Network Topologies

For evaluate CoAP congestion control in Contiki OS three network topologies are defined in this simulation. The different topologies and the positioning of the nodes in the simulation helps us to determine the number of direct neighbouring of the nodes. The evaluated topologies are: i) a Chain topology of 21nodes ii) a Rectangular Grid topology of 25 nodes (5X5) and iii) a Dumbbell topology with 21 nodes.



**Figure 4.5:** The three network topologies (chain, grid, dumbbell) for performance analysis. Nodes apart at 10m distance. Node with blue circle are sink node, node pink in colour are source node and node green in colour are RPL border router

Fig 3 Gives us the GUI snapshot of nodes positioned in the Cooja Simulation for three different topologies. The transmission range is set to be 10m which is green in colour and determines the nodes within the coverage area. Whereas, the inference range is set to be 20m shows in dark colour depicts the nodes outside the range but still approachable as due to using UDGM it has distance loss. In chain topology, the neighbouring nodes are selected in such a way that direct neighbour nodes are within the transmission range. Thus interference range is twice the transmission range which is two hops away. Similarly, in grid topology there are four direct neighbour to the transmission range and in Dumbbell topology half circle are exactly 10m away from the node which is part of the axis.

RPL border router has the RPL root for creating a routing table for all the nodes and it should be positioned at the centre of the network for easily available. In this paper CoAP SERVER (sink node) having IPv6 address is shown in yellow colour. The message is send from the CoAP CLIENT (source node) which is placed as far as can from sink node for better result which is shown in pink colour. In centre of every topology we have a RPL root denoted by node number 1 having IPv6 address [aaaa::212:7401:1:101] shown in green colour. All the topology has one source node, one RPL root node expect in dumbbell which has two RPL root node and rest are sink node. But for evaluation we have taken CoAP SERVER (sink node) number 2 having IPv6 address [aaaa::212:7401:2:202] for evaluating the network performance in congested network.

### **Test Run Configuration**

To test the run configuration of the simulation in the Contiki OS initially when the simulation is start RPL-Root creates a RPL-DODAG by spreading DAG Information Objects (DIOs) throughout the network. Before the RPL-DoDAG the node configuration statistics as [fe80::212:740\_:\_:\_] but after completing the RPL-DODAG network its node address is [aaaa::212:740\_:\_:\_] in IPv6 network. After this a periodic traffic was generated from the CoAP CLINT (source node) by sending a message to CoAP SERVER (sink node) number 2 at a period of every 10sec interval. Which sets the colour of LED at sink node set to blink in red colour.

To analyse the result at sink node 2 we take the statistics of the packets send from source node at every 30sec interval at the terminal window. Simulation is run for 5min for every algorithm and network topology to gets the statistics value to evaluate the congestion control in network. Here to generate CoAP request we take NSTSRT=1. It means we have one possible destination and the packet will be dropped when a node is waiting for a confirmation of the previous send CoAP request.

This section explains the simulation result of the various congestion control mechanism in Cooja simulation for three previously introduced network topology. All the analyzed configuration of CoAP congestion control mechanism has been repeated three times with variable random seeds for 5min each to get meaningful result. We had analysis three different algorithms namely Simple CoAP for UDP IPv6 network in CoAP, Default CoAP which was standardized by IETF and CoCoA an Advance congestion control algorithm for CoAP which uses two RTO mechanism strong and weak for calculating RTTs value along with Re-Establishment mechanism. These algorithms were analyzed on three different topologies such as Chain, Grid and Dumbbell. Following in this chapter describe the detailed discussion on packet loss, goodput of network, RTOs calculation and RTTs with respect to timeline.

Following is the sample snapshot from where we get the result to analyses the result for congestion control in CoAP. These results were taken at terminal of Contiki OS by pinging the CoAP node no 2 [aaaa::212:7402:02:202] which is the sink node.

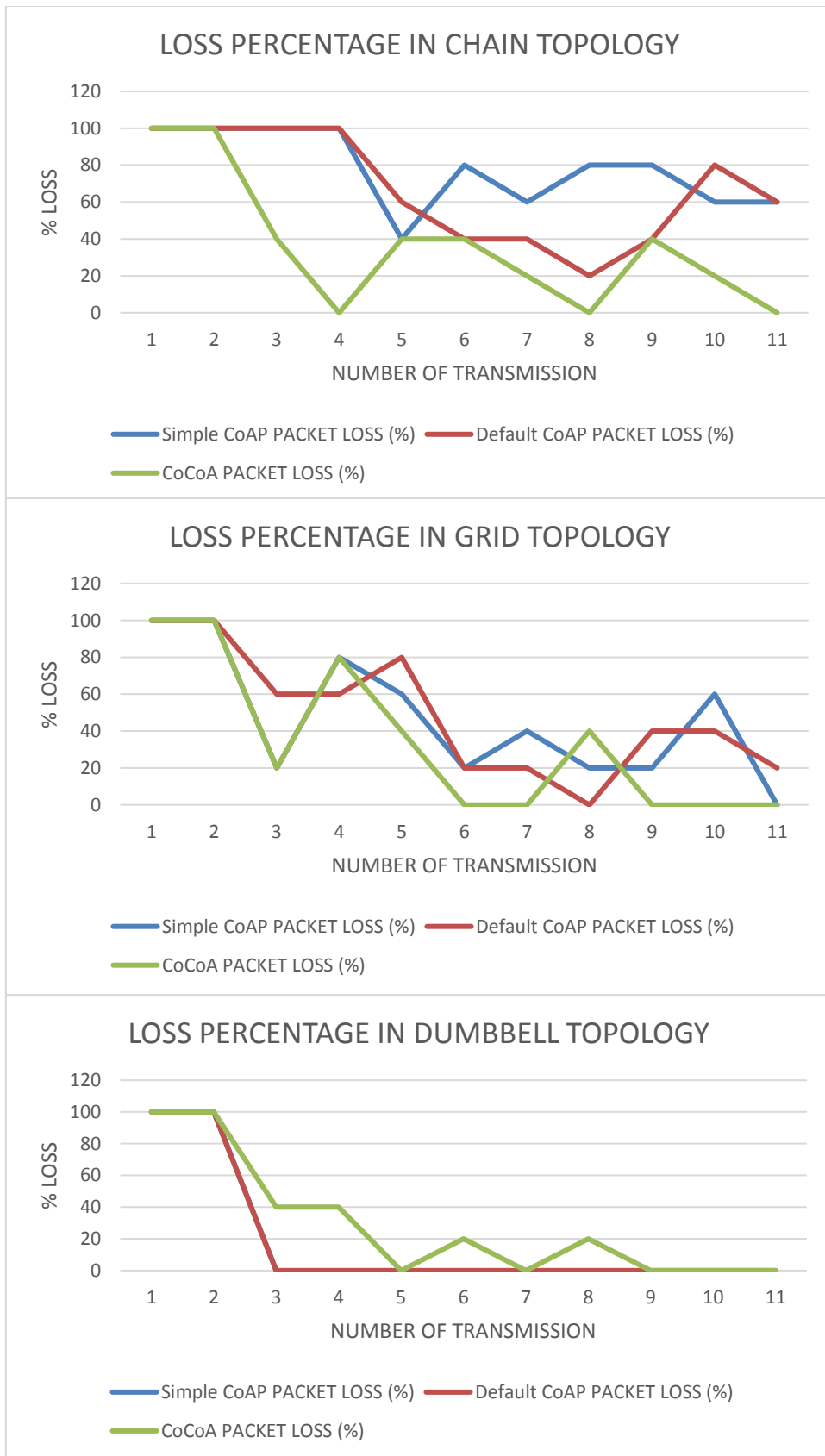
```
user@instant-contiki:~$ ping6 -c 5 -s 8 aaaa::212:7402:02:202
PING aaaa::212:7402:02:202(aaaa::212:7402:2:202) 8 data bytes
16 bytes from aaaa::212:7402:2:202: icmp_seq=1 ttl=54 time=1613 ms
16 bytes from aaaa::212:7402:2:202: icmp_seq=4 ttl=54 time=1488 ms
16 bytes from aaaa::212:7402:2:202: icmp_seq=5 ttl=54 time=1722 ms

--- aaaa::212:7402:02:202 ping statistics ---
5 packets transmitted, 3 received, 40% packet loss, time 4009ms
rtt min/avg/max/mdev = 1488.691/1608.132/1722.551/95.544 ms, pipe 2
```

Figure 5.1: Sample Result Terminal

#### 5.1 Experimental Analysis of Packet Loss Percentage

To examine the percentage loss in CoAP congestion control mechanism five packets were send from source node (Client CoAP) to sink node (Source CoAP) were send at the interval of 30ms. If no packet was delivered at the sink node then percentage loss was 100% if all packets were delivered then packet loss is 0%. Duplicate packets were also considered as loss packets. Following were the analysis of different congestion control algorithm.



**Figure 5.2:** Analysis of various algorithms for percentage loss wrt to timeline with various topologies Chain (Top), Grid (Middle) and Dumbbell (Under)



To examine the loss percentage of various algorithms in the above figure the loss percentage wrt no of transmissions has been shown. Each Algorithm for different topologies like chain(top), grid (middle) and dumbbell(bottom) had been analyzed by transmitting 5 packets each over several times. In Simple CoAP the loss percentage of packets throughout the timeline is more than the Default and CoCoA. Here CoCoA has the least packet loss. In Grid and Chain topology, the variation among the different algorithms can be analyzed very easily. Here Simple CoAP is showing high peaks thus the loss of packet transmitted is 100%. In CoCoA, the shows the various along with the axis means it has least loss of packet loss and Default CoAP has the average packet loss. But in Dumbbell scenario due to nodes are more reachable to each other thus here Default CoAP performs better than the Simple CoAP and CoCoA.

## 5.2 Experimental Analysis of Goodput

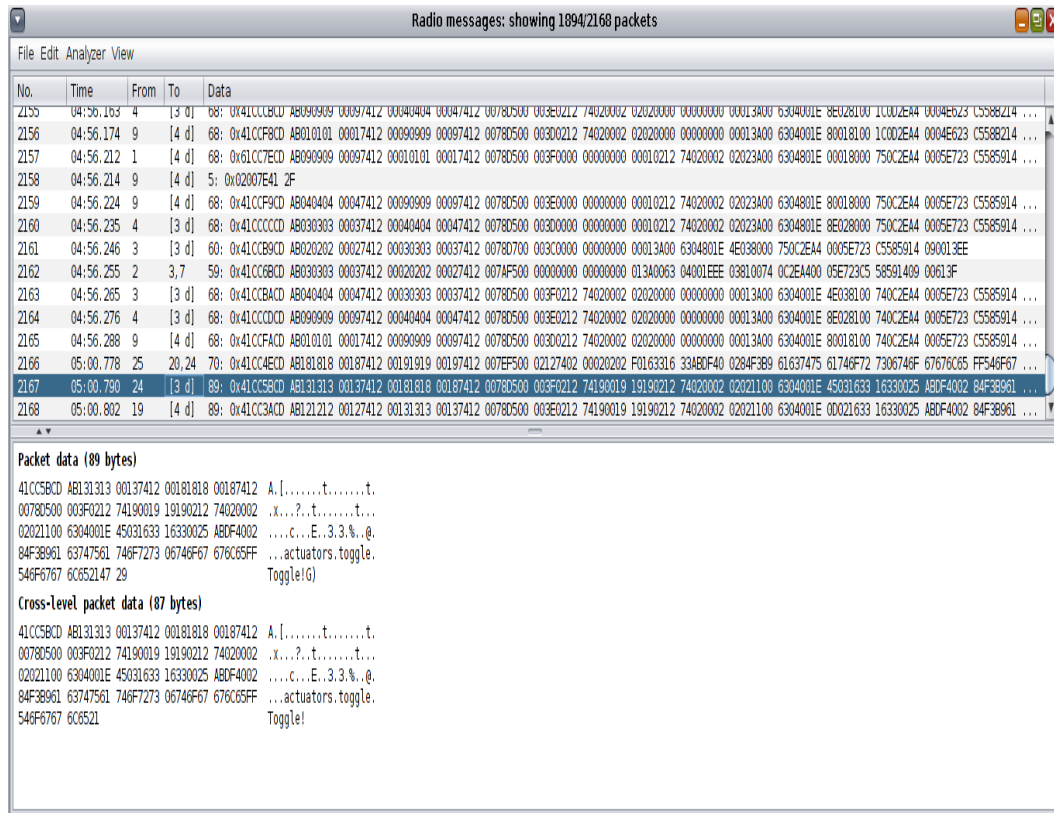
In this section goodput is calculated which is useful packets received vs total number of packets send. Following is the table from where the result is taken.

$$Goodput = \frac{\text{useful packet received or ACKs received}}{\text{Total number of packets send}} \quad (7)$$

**Table 5.1:** Table for analyzing Goodput

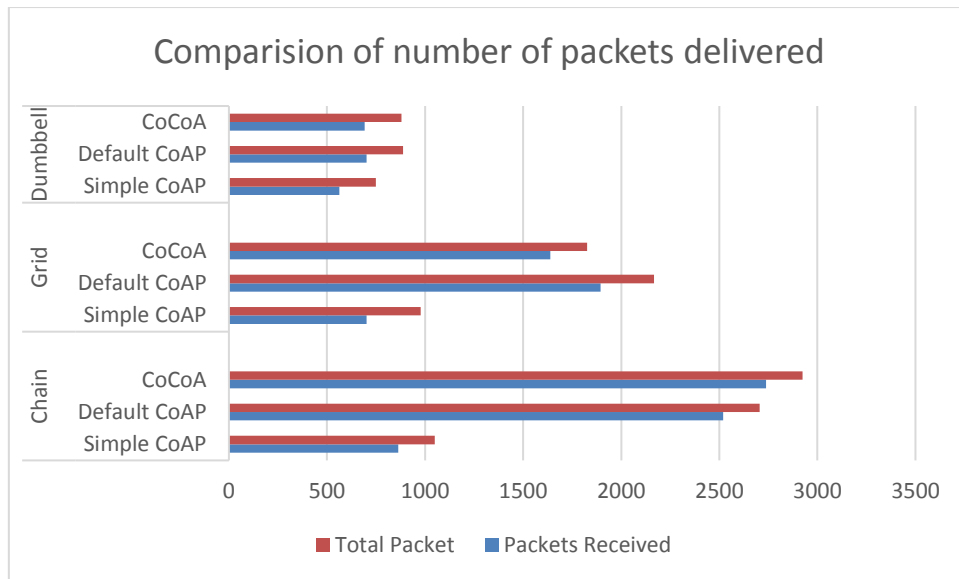
TOPOLOGIES	ALGORITHMS	PACKETS RECEIVED	TOTAL PACKETS	PERCENTAGE GOODPUT
Chain	Simple CoAP	864	1050	82.28571
	Default CoAP	2520	2706	93.12639
	CoCoA	2739	2925	93.641
Grid	Simple CoAP	701	978	71.67689
	Default CoAP	1894	2168	87.3616
	CoCoA	1638	1826	89.7043
Dunbell	Simple CoAP	563	749	75.16689
	Default CoAP	701	888	78.94144
	CoCoA	692	879	78.72558

From the above table, we can analyze that percentage goodput of CoCoA is always more than rest of the algorithms. Then for the Default CoAP and Simple CoAP has the least goodput.



**Figure 5.3:** Sample Packets received/ Total number of Packets send a Cooja simulation

The above figure was a sample snapshot taken for radio message send from CoAP Client to CoAP source. It shows the detail packets data at a particular time and by clicking on the row for specific time it describes the Packet data in bytes and Cross-Level Packet data in bytes.



**Figure 5.4:** Comparison of Goodput (useful packet received/Total no of packets send) of various congestion control mechanism with different topologies.

Fig Shows the effective goodput of different schemes we calculate the useful packet received over total number of packets send. More from the graph we can detect that the CoCoA performs better than the both Default CoAP and Simple CoAP algorithms. Thus, when the packets send between source and sink node the ratio of packet received at Sink node in CoCoA is more. On X-axis is the total no of packets send over time is shown it is calculated from the RPL border router.

### 5.3 Experimental Analysis of RTO Calculation for Algorithms

Retransmission Time Out (RTO) is the interval when no ACK was received for the send packet. Following are the different RTOs was used to evaluate our result with different algorithms.

#### 5.3.1 Simple CoAP Algorithm

Constrained RESTful environment (CoRE) standardized CoAP with RESTful architecture which is suitable for most of the constrained devices. [16] IETF has standardized in CoAP draft version 18. It uses simple stop-and-wait retransmission reliability with exponential backoff for confirmable messages and it also detect duplication messages for both confirmable and non-confirmable messages

In this algorithm, initial time out is set to a random duration between  $ACK\_TIMEOUT$  and  $(ACK\_TIMEOUT * ACK\_RANDOM\_FACTOR)$ , re-

transmission counter is set to 0. The message retransmitted counter is incremented and timeout is doubled when the timeout is triggered and retransmission counter is less than MAX\_RETRANSMIT. Thus, RTO value was calculated as

$$RTO = ACK\_TIMEOUT * ((2 * MAX\_RETRANSMIT - 1) * ACK\_RANDOM\_FACTOR) \quad (8)$$

Initially ACK\_RANDOM\_FACTOR= 1.5

### 5.3.2 Default CoAP Algorithm

In CoAP, when a CON message is send it requires an ACK from the receiver and retransmission is done 4 times unless it receives failed message. For the initial message re-transmission CoAP randomly select initial transmission timeout (RTO) value between interval of 2s and 3s. If CON\_ACK is not received then binary exponential backoff (BEB) is applied before the MAX\_RETRANSMIT reached by doubling the RTO value as [11]

$$RTO = ACK\_TIMEOUT * ((2 * MAX\_RETRANSMIT + 1) - 1) * ACK\_RANDOM\_FACTOR \quad (9)$$

Here parameter NSTART determines how many transaction in CoAP can be created to one destination end. In base CoAP document it has taken value 1.

### 5.3.3 CoCoA Algorithm

CoCoA is more flexible congestion control solution than default CoAP as it gives better result tha or atleast similar result as default CoAP. In CoAP, the initial RTO always have a fixed interval and BEB is applied on the retransmission of message. The three main mechanism of CoCoA: adaptive RTO calculation, a variable backoff factor (VBF) and RTO aging [23].

CoCoA runs two RTO estimators: a strong and weak RTO estimator for each destination end point which gets updated when strong RTTs and weak RTTs is calculated respectively. Strong RTTs ( $RTT_{strong}$ ) are calculated after the ACK

received of the first transaction of a CON CoAP message. This  $RTT_{strong}$  value is used to update strong RTO estimator ( $RTO_{strong}$ ). Similarly if the ACK received after one transaction weak RTTs is calculated after the ACK received of the further transaction. This  $RTT_{weak}$  value is used to update weak RTO estimator ( $RTO_{weak}$ ).

$$RTTVAR_{strong} = (1-\beta) * RTTVAR_{strong} + \beta * (RTTVAR_{strong} + RTTVAR_{strong\_new})$$

$$RTT_{strong} = (1-\alpha) * RTTVAR_{strong} + \alpha * RTTVAR_{strong\_new}$$

Where  $\alpha=1/4$  and  $\beta=1/8$

$$RTTVAR_{weak} = (1-\beta) * RTTVAR_{strong} + \beta * (RTTVAR_{weak} + RTTVAR_{weak\_new})$$

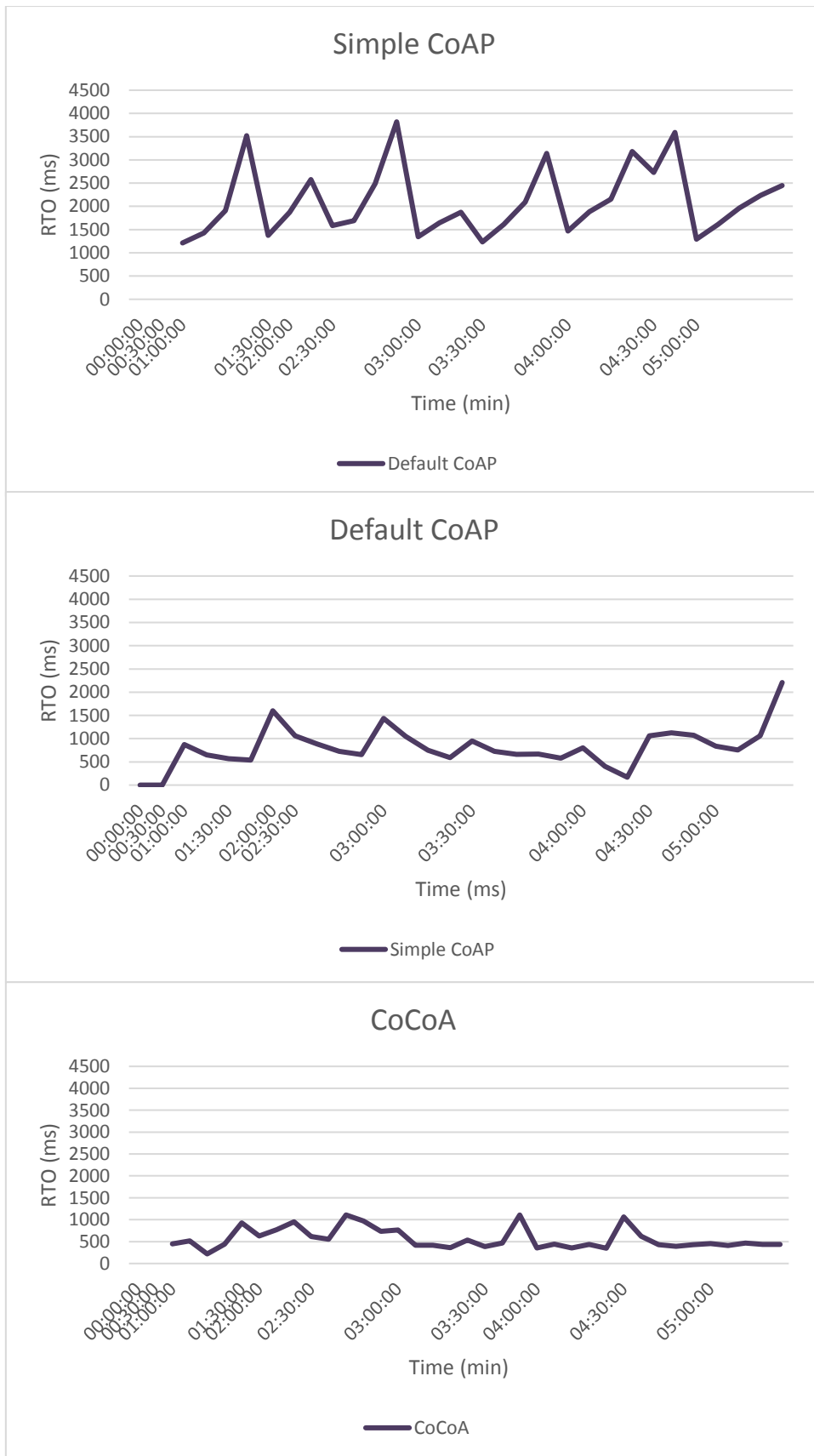
$$RTT_{weak} = (1-\alpha) * RTTVAR_{weak} + \alpha * RTTVAR_{weak\_new}$$

Where  $\alpha=1/4$  and  $\beta=1/8$

After calculating the  $RTO_{strong}$  and  $RTO_{weak}$  values we will calculate Overall RTO ( $RTO_{overall}$ ) as shown below.

$$RTO_{overall} = 0.5 * RTO_{strong/weak} + 0.5 * RTO_{overall} \quad (10)$$

This will be the next initial RTO ( $RTO_{initial}$ ) value for CON message, which will be randomly between chosen interval of  $[RTO_{overall}, RTO_{overall} * 1.5]$



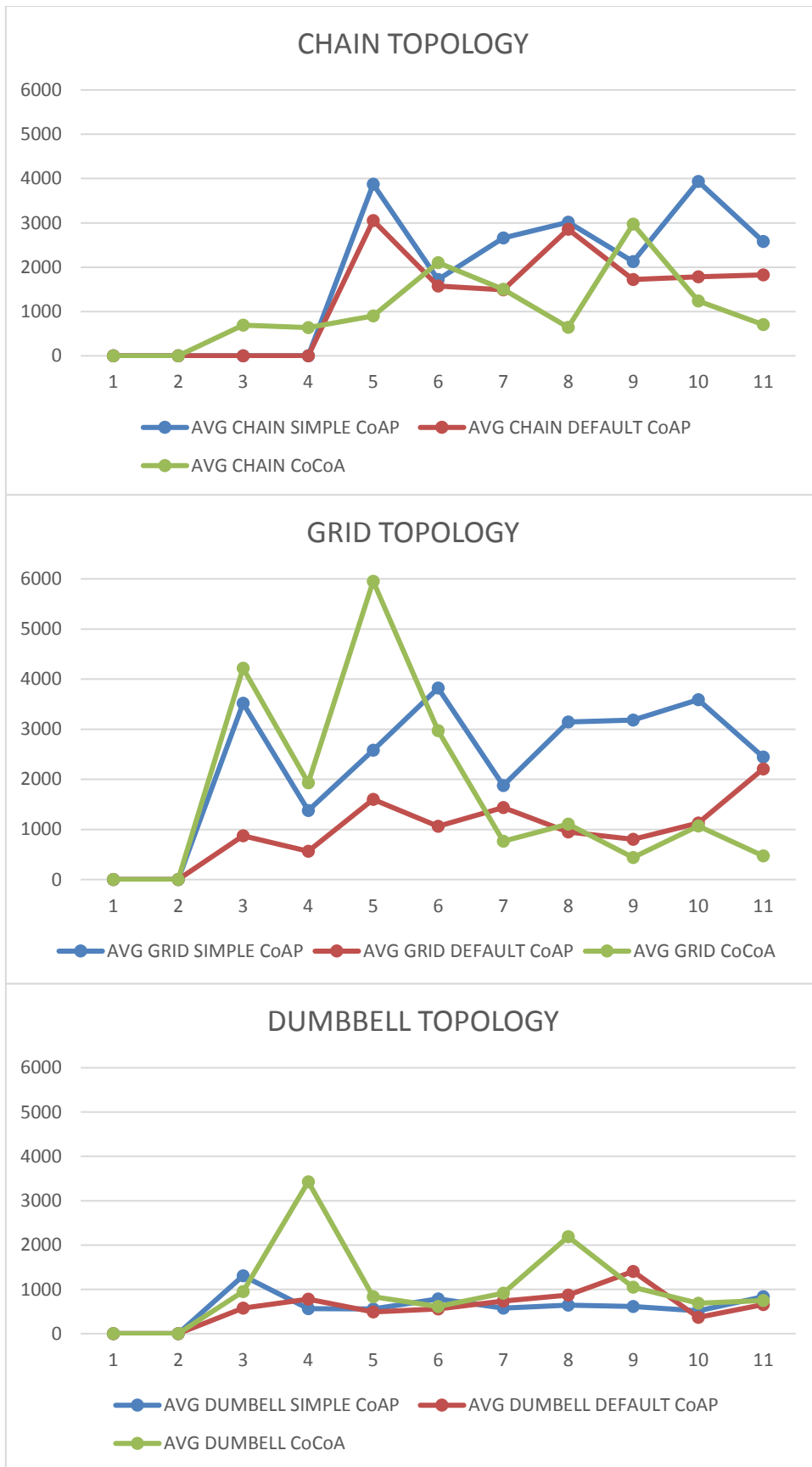
**Figure 5.5:** Calculated RTO values of Simple CoAP, Default CoAP and CoCoA plotted against time for transmitted packets when RTT is measured

An important observation from our experiments is that CoCoA is more conservative than Simple and Default when the packets are transmitted. It means packets transmitted by CoCoA has lower RTO value and consequently, there are less packet losses. It is because it uses variable backoff factor. While in first attempt if packet ACK is delivered CoCoA Strong, which simply ignores the first loss while behaving exactly like CoCoA for successive losses. If the first ACK is not received then it implies CoCoA Weak for RTT calculation.

From the graph, we can observe that the gap between two consecutive transmission is greater in CoCoA. It gives us the idea that packet transmitted by CoCoA has lower packet loss than the Simple CoAP or Default CoAP.

#### **5.4 Experimental Analysis of RTT Calculation for Algorithms**

In this section, we analyse the calculated RTTs for different algorithms and then compare with various topologies. Round Trip Time (RTT) of a CoAP packet exchange (confirmable message and its ACK) between two motes (i.e., sensor nodes) to calculate a parameterized Retransmission Time-Out (RTO) that is adapted over the time depending on RTT estimates made by the mote. For Default CoAP only one RTT value was calculated but for CoCoA runs two RTO estimators for estimation which are: a strong estimator which uses ACKs of the original transmissions and a weak estimator that uses ACKs of transmissions. In the following graph the calculated RTTs was shown over timeline.



**Figure 5.6:** RTT comparisons of various topologies wrt their algorithms



Here, from the above graph it can be observed that in Chain and Grid Topology CoCoA has the lower value of RTT. But in the dumbbell CoCoA outperforms than Default CoAP which has lower RTT value.

From the above section, we can conclude in comparison to Simple CoAP, default CoAP and CoCoA for analysing the network performance. CoCoA consistently delivers a performance which is better than, or at least similar to, that of default CoAP. CoCoA performs better in evaluating the percentage loss of packets, goodput of the network, analysis the RTO and calculating the RTTs value. The percentage of goodput is always approx. equal to or 2% more than Default CoAP. Thus using the Re-Establishment mechanism in CoCoA for congestion control in CoAP based IoT perform better.

# CONCLUSION AND FUTURE SCOPE

---

### 6.1 Conclusion

In this paper, we carry out the experiment to study three congestion control mechanism in CoAP based IoT for Client-Server communication. For implementing we use Cooja simulation in Contiki OS which includes protocol stack standardised by IETF for constrained network. Three different mechanism namely UDP based IPv6 network Congestion Control called Simple CoAP, Default CoAP which is IETF proposed alternative congestion control mechanism and an advance Congestion control mechanism for CoAP known as CoCoA.

For better result we modify the CoCoA by adding Re-Establishment mechanism that will terminate the transmission after n number of retransmission and save the packet unless the CoAP server is free. It also uses VBFs for adaptive RTO calculation from Strong RTTs and Weak RTTs and uses aging mechanism to optimise performance.

In comparison to Simple CoAP, default CoAP and CoCoA for analysing the network performance. CoCoA consistently delivers a performance which is better than, or at least similar to, that of default CoAP. CoCoA performs better in evaluating the percentage loss of packets, goodput of the network, analysis the RTO and calculating the RTTs value. The percentage of goodput is always approx. equal to or 2% more than Default CoAP.

### 6.2 Future Scope

This thesis has demonstrated various mechanism of congestion control in CoAP base IoT namely, Simple Coap, Default CoAP and CoCoA using Re-Establishment mechanism. But IoT has low power and lossy network so there is scope of improving the network performance through various means. As a future work, many opportunities for extending the scope of this thesis remains as discussed below.

## **Dynamic CoAP nodes**

In this research work we have taken static nodes with defined interference and transmission range. This makes a DODAG with the initialisation of the simulation and make a routing table with border router node. However, we can enhance its performance with adding the mobile nodes that will dynamically sense the neighbouring nodes and make an optimised path to the destination node. CoAP nodes will not have to follow a defined path to make a DODAG but it can access any path and any node to make the transmission of packets with more efficiency.

## **Security in CoAP nodes**

In this research work we had implemented nodes without any security measures. Nodes transmits the packets containing data from source node to destination node without any security measures. Further while transmitting the data packets various security algorithms applicable to IoT in CoAP application layer can be deployed to get more authentic packets at destination nodes.

## **Changing NSTART value**

NSTART defines the number of transaction can be created at one destination. Here in this paper we had used NSTART value as 1. So, in future we can apply the changes by taking higher NSTART value and then analysing our result.

## **Testing with Cooper (Cu)**

The Copper (Cu) CoAP user-agent for Firefox installs a handler for the 'coap' URI scheme and allows users to browse and interact with Internet of Things devices. Using the Copper (Cu) plugin we can send message/packets online through IPv6 network to the targeted node from anywhere.

## REFERENCES

- [1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Networks*, vol. 54, no. 15, pp. 2787–2805, Jun. 2010.
- [2] J. Macaulay, L. Buckalew, and G. Chung, "Internet of Things in Logistics," Sep. 2008.
- [3] C. Alcaraz, P. Najera, J. Lopez, and R. Roman, "Wireless Sensor Networks and the Internet of Things: Do We Need a Complete Integration?," *1st Int. Work. Secur. Internet Things*, no. IEEE, p. xxxx, Dec. 2010.
- [4] D. Christin, A. Reinhardt, P. S. Mogre, and R. Steinmetz, "Wireless Sensor Networks and the Internet of Things : Selected Challenges," pp. 31–33.
- [5] S. Cirani *et al.*, "A scalable and self-configuring architecture for service discovery in the internet of things," *IEEE Internet Things J.*, vol. 1, no. 5, pp. 508–521, Oct. 2014.
- [6] C. From, "A CCEPTED FROM O PEN C ALL A S URVEY ON THE IETF P ROTOCOL S UITE FOR THE I NTERNET OF T HINGS : S TANDARDS , C HALLENGES , AND O PPORTUNITIES S HUSEN Y ANG , I MPERIAL C OLLEGE L ONDON," *IEEE Wirel. Commun.*, vol. 13, pp. 91–98, Dec. 2013.
- [7] G. K. Teklemariam, J. Hoebeke, F. Van Den Abeele, I. Moerman, and P. Demeester, "Simple RESTful Sensor Application Development Model Using CoAP," *9th IEEE Work. Pract. Issues Build. Sens. Netw. Appl.*, vol. 9, pp. 552–556, 2014.
- [8] C. Hou, "SeaHttp : A Resource-Oriented Protocol to Extend REST Style for Web of Things," vol. 29, no. 2013, pp. 205–215, 2014.
- [9] C. Control and C. A. Protocol, "1/27/2017 draft-eggert-core-congestion-control-01 - Congestion Control for the Constrained Application Protocol (CoAP)," *Netw. Work. Gr. Internet- Draft*, no. c, pp. 1–10, Jan. 2017.

- [10] “1/27/2017 <https://www.ietf.org/rfc/rfc5405.txt>,” *Netw. Work. Gr. Internet-Draft rfc5405*, pp. 1–22, 2017.
- [11] D. R. Tobergte and S. Curtis, “RFC 7252 - The Constrained Application Protocol (CoAP),” *J. Chem. Inf. Model.*, vol. 53, no. 9, pp. 1689–1699, 2013.
- [12] M. Swarna, S. Ravi, and M. Anand, “Adaptive Backoff Algorithm for Congestion Control in IoT,” *Int. Sci. Press*, vol. 9, no. 4, pp. 205–214, 2016.
- [13] G. Choi and D. Kim, “Efficient Streaming over CoAP,” *Inf. Netw.*, pp. 476–478, Mar. 2016.
- [14] I. Järvinen, L. Daniel, and M. Kojo, “Experimental Evaluation of Alternative Congestion Control Algorithms for Constrained Application Protocol ( CoAP ),” *Internet of Things (WF-IoT)*, pp. 0–5, Jan. 2015.
- [15] M. Sargent, “RFC 6298 - Computing TCP’s Retransmission Timer,” *Internet Eng. Task Force RFC 6298*, pp. 1–11, 2017.
- [16] T. Constrained and A. Protocol, “3/19/2017 draft-ietf-core-coap-18 - The Constrained Application Protocol (CoAP),” *CoRE Work. Gr. Internet- Draft*, pp. 1–118, 2017.
- [17] I. Demirkol, “1/28/2017 draft-ietf-core-cocoa-00 - CoAP Simple Congestion Control/Advanced,” *CoRE Work. Gr. Internet- Draft*, no. c, pp. 1–14, 2017.
- [18] A. Levakov and N. Sokolov, “Internet of Things, Smart Spaces, and Next Generation Networking,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7469, no. January, pp. 424–428, 2012.
- [19] R. Bhalerao, S. S. Subramanian, and J. Pasquale, “An analysis and improvement of congestion control in the CoAP Internet-of-Things protocol,” *2016 13th IEEE Annu. Consum. Commun. Netw. Conf. CCNC 2016*, pp. 889–894, 2016.
- [20] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, “Congestion control in

- reliable CoAP communication,” *Proc. 16th ACM Int. Conf. Model. Anal. Simul. Wirel. Mob. Syst. - MSWiM '13*, pp. 365–372, 2013.
- [21] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, “CoCoA+: An advanced congestion control mechanism for CoAP,” *Ad Hoc Networks*, vol. 33, pp. 126–139, 2015.
- [22] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, “CoCoA plus: An advanced congestion control mechanism for CoAP,” *Ad Hoc Networks*, vol. 33, pp. 126–139, 2015.
- [23] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, “CoAP Congestion Control for the Internet of Things,” *IEEE Commun. Mag.*, pp. 154–160, Jul. 2016.
- [24] Y. Chen and T. Kunz, “Performance evaluation of IoT protocols under a constrained wireless access network,” *2016 Int. Conf. Sel. Top. Mob. Wirel. Networking, MoWNeT 2016*, 2016.
- [25] A. Betzler, C. Gomez, I. Demirkol, and M. Kovatsch, “Congestion control for CoAP cloud services,” *19th IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA 2014*, Jan. 2014.
- [26] S. Song, J. Lee, K. Son, H. Jung, and J. Lee, “A congestion avoidance algorithm in SDN environment,” *Int. Conf. Inf. Netw.*, vol. 2016-March, pp. 420–423, 2016.
- [27] A. Sehgal, “Using the Contiki Cooja Simulator,” *Comput. Sci. Jacobs Univ. Bremen Campus Ring*, vol. 1, p. 28759, 2013.
- [28] A. Dunkels, “Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors,” *Local Comput. Networks*, vol. 4, pp. 455–462, Nov. 2004.
- [29] F. Österlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, “Cross-level sensor network simulation with COOJA,” *Proc. - Conf. Local Comput. Networks, LCN*, pp. 641–648, 2006.
- [30] C. Thomson, “Cooja Simulator Manual,” no. C, pp. 2015–2016, 2016.