Delft University of Technology
Master of Science Thesis in Embedded Systems

Improving Zigbee Performance of a Wireless Lighting System in a Smart Home Environment

Kaustubh Agarwal

Improving Zigbee Performance of a Wireless Lighting System in a Smart Home Environment

Master of Science Thesis in Embedded Systems

Embedded and Networked Systems Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Kaustubh Agarwal
k.agarwal-1@student.tudelft.nl
kaustubhagarwal18@gmail.com

31 August 2020

Author
Kaustubh Agarwal (k.agarwal-1@student.tudelft.nl)
(kaustubhagarwal18@gmail.com)

Title
Improving Zigbee Performance of a Wireless Lighting System in a Smart Home Environment

MSc Presentation Date
Friday September 4, 2020

Graduation Committee

dr. ir. Fernando A. Kuipers (chairman)	Delft University of Technology
ir. Leo Rozendaal	Signify
dr. ir. Arjan van Genderen	Delft University of Technology

Abstract

The accelerating growth of the Internet of Things (IoT) has led to the development of many technologies for homes like a wireless lighting system. Unlike traditional lighting systems, these can be operated remotely using mobile devices and even with the help of a smart assistant, such as Amazon Alexa or Google Assistant. Zigbee is used as the standard of communication between the wireless lighting system whereas other Wi-Fi applications are also extensively used in homes.

A common problem for wireless radios such as Wi-Fi and Zigbee is that they have to share the ISM band which could lead to coexistence issues. Zigbee has lower power and longer channel sense times than Wi-Fi which makes it more vulnerable to interference. Bandwidth hungry Wi-Fi applications such as multimedia streaming cause adverse effects on Zigbee's performance and the system can become temporarily out of reach as the nodes cannot gain access to the channel. This results in inconvenience to the user as the system might not respond in such situations.

As these standards are fundamentally different and it is almost impossible to operate on a non-overlapping channel especially with the wide deployment of Wi-Fi networks, a new centralized approach is proposed in this work to coordinate between the two heterogeneous networks. The Wi-Fi router which serves as a gateway for both networks in a home is utilized as a coordinator for these networks. By performing various experiments, we examine the lowest data rate for Wi-Fi at which Zigbee can transmit its packet in a reliable manner. We propose a system design that effectively converts a high Wi-Fi interference environment to a transient low interference environment during Zigbee transmission.

By combining the packet detection capabilities of the Linux router firewall and a custom queueing setup, we show we can provide reliability to the wireless lighting system with zero to a minimum loss for Wi-Fi transmission. We detect the Zigbee packet well in advance which enables us to adopt a preventive approach rather than a reactive one. Our system design results in a decrease of Zigbee packet loss from 67% to 7% while maintaining an average RTT of 35 ms at every load. We keep the complexity of the system design low by only making software changes to the router and not introducing a new node for synchronization between the two networks. The performance analysis of the system design is done using a test bed consisting of multiple Zigbee and Wi-Fi nodes, with the Wi-Fi router acting as a central controller for both of these networks.

Preface

This thesis marks the end of my studies at the Delft University of Technology for the Masters of Science degree in Embedded Systems in the track of Software and Networking. This thesis was carried out at the New IoT System Architectures department of Signify Research located in the High Tech Campus of Eindhoven.

Firstly, I am grateful to my supervisor Leo Rozendaal and my manager Bas Driessen from Signify (formerly known as Philips Lighting) for providing me with such a challenging yet interesting and fun graduation project. Leo, I would like to thank you for the time and effort that you have invested in this work. Thank you for always being available and for all the brainstorming sessions that we had. I would also like to express my gratitude towards all the members of the New IoT System Architectures department of Signify. Thank you for your support and for making it a great experience for me. A special thanks to all the members of the Philips Hue development team for helping me get accustomed to the Hue ecosystem and being always present to answer all of my questions (no matter how silly they were).

I would like to thank Dr. Fernando Kuipers for his continued patience and his supportive supervision during the different stages of this project. Thank you for repeatedly pointing me in the right direction and answer all my questions. This thesis wouldn't have been possible without your support and your invaluable inputs during the project.

A special thanks to Dr. Arjan van Genderen for agreeing to be a part of the thesis committee.

Finally, this journey would not have been possible without the unconditional support and love of my family in India and Japan. Last but not the least, a big shoutout to all my friends for their continuous support so that I could complete my thesis in a timely manner and celebrate with them afterward!

Kaustubh Agarwal

Eindhoven, The Netherlands
31st August 2020

Contents

Acronyms

AP Access Point.

ACK Acknowledgement.

API Application Programming Interface.

APS Application Sublayer.

CCA Clear Channel Assesment.

CSMA/CA Carrier Sense Multiple Access with Collision Avoidance.

CW Contention Window.

DIFS DCF Interframe Spacing.

HD High Definition.

HTB Hierarchical Token Bucket.

HTTPS Hypertext Transfer Protocol Secure.

IDE Integrated Development Environment.

IoT Internet of Things.

IP Internet Protocol.

ISM Industrial, Scientific and Medical.

LAN Local Area Network.

LED Light Emitting Diode.

LOS Line Of Sight.

MAC Medium Access Control.

NWK Network.

PER Packet Error Rate.

PHY Physical Layer.

qdisc queueing discipline.

QoE Quality of Experience.

QoS Quality of Service.

RF Radio Frequency.

RTS Request To Send.

RTT Round Trip Time.

SoC System on Chip.

SIFS Shortest Interframe Spacing.

TBF Token Bucket Filter.

tc traffic control.

TCP Transmission Control Protocol.

ToS.Type of Service.

UDP User Datagram Protocol.

VoIP Voice Over Internet Protocol.

WAN Wide Area Network.

WSTK Web Services Tool Kit.

Wi-Fi Wireless Fidelity.

ZDO Zigbee Device Object.

ZC/ZR Zigbee Coordinator/Router.

ZED Zigbee End Device.

Chapter 1

Introduction

Advances in wireless networking technologies have led to the development of several home automation technologies. These technologies not only make our lives convenient but also conserve energy for households. IoT has been a part of home automation and has become synonymous with making homes smart. It carries out an extremely critical role in enabling technologies like smart lighting. According to [23], 50 % of newly connected devices between 2018 and 2030 will be used in smart building automation, smart industrial automation, and efficient lighting solutions.

In the traditional home lighting, the conventional light bulbs work by physically flipping the switch on and off. However, smart lighting, on the other hand, gives the consumer far more control over the lights. Philips Hue is a smart home lighting system developed by Signify (previous known as Philips Lighting) which not only controls switching lights on and off but can also be instructed to produce colors that can be controlled via a website or a smartphone [50]. Each light is still connected to the home's power, but each light can be controlled wirelessly even with the use of a smart assistant, such as Amazon Alexa or Google Assistant. It can be employed to create scenes that can completely change the atmosphere in the room according to the user's mood. There are multiple advantages to wireless networks for lighting like the ease of installation, maintenance, remote monitoring, and the ability to use the existing infrastructure. However, these applications require high reliability and low latency which are not always guaranteed.

The most widely used wireless technology for smart lighting is Zigbee as it supports low data rates and low power consumption which are important for controlling and monitoring applications. Zigbee also supports mesh topology which comes in handy especially in the case of broken or faulty links due to its self-healing properties. Philips Hue is based on the Zigbee communication protocol. It supports the Zigbee Light Link standard which includes multiple features like home ambiance and offers the possibility of controlling the lights via the Internet using an IP router or gateway as one of the network nodes. It also ensures third-party applications and light bulbs from other manufacturers can also work seamlessly alongside the Hue ecosystem [57, 37].

1.1 Problem Description

Both Zigbee and Wi-Fi are regarded as key technologies for IoT applications and are widely used in a home environment. They operate on the same unlicensed 2.4 GHz ISM (industrial, scientific and medical) band. Due to the ubiquitous deployment of these devices, cross-technology interference is inevitable more so in a home environment. These technologies share limited frequency resources and are present close to each other.

Zigbee and Wi-Fi both employ a contention-based listen before talk mechanism called CSMA/CA to gain access to the channel. However, many studies have shown Wi-Fi can severely degrade the performance of Zigbee networks, especially under high traffic conditions. On the other hand, Zigbee has little to no impact on Wi-Fi performance[52]. This behavior is due to unfair PHY and MAC parameters for Zigbee as it has low transmit power (close to 0 dBm) and high channel sense time as compared to Wi-Fi[22]. This asymmetry gives Wi-Fi an inherent advantage over Zigbee as they compete for the same limited resources. Zigbee can suffer from high latency and even packet loss when it's colocated with Wi-Fi. Over the years, improvement in the Wi-Fi standard to increase throughput has led to the use of Multimedia applications becoming more common. Bandwidth hungry streaming applications like Netflix, YouTube produces an excessive amount of traffic in a single burst. Such applications demand high throughput services which are time-bounded, especially when used for HD streaming. This can lead to high channel occupancy for Wi-Fi that can completely cut off Zigbee transmission and result in an unpleasant situation for a smart lighting system. The system can become temporarily unavailable as the nodes cannot gain access to the channel. This situation can be very irritating for a consumer as he/she temporarily might not be able to control the lighting system.

Over the years, this coexistence problem has been extensively examined in literature and many models exist which helps Zigbee in mitigating Wi-Fi interference. Some of these models suggest operating Zigbee networks on a non-overlapping channel with Wi-Fi[43]. This approach being helpful is not always practical due to the number of different Wi-Fi networks present nearby. As they try to operate on different non-overlapping channels, it makes it very difficult for a Zigbee network to find a free channel. To combat the issue of low transmit power and unfair MAC parameters, some solutions introduce a Wi-Fi node which they use to compete for the channel[62]. This node is used as a signaller which increases the visibility of the Zigbee network for Wi-Fi. However, this approach causes unfairness to Wi-Fi networks as the Wi-Fi transmission is cut off whenever there is ongoing Zigbee transmission. Some approaches implement changes to the Zigbee protocol. In one approach, two headers are sent instead of one as the authors claim that the collision between the Zigbee and Wi-Fi packet only affects the start of the header and the data packet can still be recovered[12]. The limitations of this approach include making changes at both the transmitter and receiver as they need to be synchronized to remove the first header which might have been corrupted.

Some approaches attempt to solve this problem in a centralized manner. They introduce an extra node that has support for both Wi-Fi and Zigbee standards. This gateway node works as a central controller to provide synchronization between the two heterogeneous networks. The main drawback of such an approach is that they increase the complexity of the system and latency as both technologies need to talk to the gateway node for coordination. Besides, using extra hardware is not always practical for cost reasons. In most cases, all previous solutions either make hardware/MAC level changes to the Zigbee network or increase the complexity/cost of the solution by the introduction of an extra node. The unfairness caused to the Wi-Fi network in these solutions is also a matter of concern. There is a lack of a solution that can not only help Zigbee mitigate Wi-Fi interference but can also provide fairness to the Wi-Fi network.

1.2 Research Goals

This work concerns the research and development of a system model that can support the Zigbee network of the smart lighting system to mitigate Wi-Fi interference without increasing the complexity of the system. Apart from providing Zigbee priority over Wi-Fi, it equally concerns causing little to zero Wi-Fi performance degradation. We analyze how we can adopt a centralized approach to gain control over the transmission of both networks. To provide fairness to the Wi-Fi network, instead of completely stopping its transmission, we propose lowering its bandwidth for a limited time whenever a Zigbee transmission is about to happen. This would result in a transient low interference environment for Zigbee which it could take advantage of. This time will ensure reliable transmission for Zigbee while ensuring zero or minimum loss for the Wi-Fi network. This leads to the following research questions:

1. How can we employ the existing infrastructure to provide reliable Zigbee transmission to a smart lighting system?
2. What is the optimal bandwidth and the time for Wi-Fi to allow reliable Zigbee transmission?
3. What is the effect of the proposed system design on the Zigbee network under different Wi-Fi traffic loads?
4. What is the effect of the proposed system design on the Wi-Fi network under different Wi-Fi traffic loads?

1.3 Contribution

This work aims to provide a novel centralized approach for a smart lighting system to mitigate Wi-Fi interference without the need for a secondary node. The proposed system design provides superior reliability and low latency to Zigbee transmissions meanwhile keeping Wi-Fi performance degradation to a minimum. As the Wi-Fi router is used as a gateway for both of these networks, the system design proposes employing it as a central controller. The router with the help of iptables, can detect a packet destined for the Zigbee network even before the Zigbee node starts transmitting [44]. This allows our system to utilize a preventive approach rather than a reactive approach, unlike previous approaches. As Zigbee networks are robust against low Wi-Fi interference, the router can then be employed to convert a high interference environment (high Wi-Fi bitrate) to a low interference environment (low Wi-Fi bitrate) by the help of custom queueing disciplines for a limited time [6]. This time of low Wi-Fi traffic ensures reliable transmission for Zigbee meanwhile keeping Wi-Fi transmissions ongoing albeit at a reduced rate. Packets that arrive during this duration are queued in the buffer of the router to minimize data loss for Wi-Fi. This method ensures that the Wi-Fi network doesn't suffer too much from Zigbee transmission and the Zigbee network can transmit its data with high reliability and low latency.

1.4 Outline

This thesis is structured as follows. Chapter 2 provides an introduction to the Zigbee network stack. In this chapter, we discuss the MAC, PHY specification for Zigbee, and also about different topologies and Zigbee device roles. Chapter 3, starts by providing an overview of the coexistence problem between Wi-Fi and Zigbee networks. We conclude this chapter by discussing the previous research on this topic. In chapter 4, we look at the Philips Hue smart lighting system and how it interacts with the Wi-Fi router. We present the results of a survey which was organized to confirm the performance drops of the lighting system was indeed due to Wi-Fi interference. We discuss how we can use a dynamic frequency allocation method to mitigate interference in a low-density environment. In chapter 5, we introduce our proposed system design. We look at how packets move in an IP network and the various steps involved in the firewall application of the router. We also look at different queueing disciplines that can be applied at the interface of a router to alter its traffic. We conclude this chapter by presenting our final system design. Chapter 6 lists the hardware and software tools used to implement our system design. In Chapter 7, we perform the performance analysis of our system design and present our results. We conclude the thesis in Chapter 8 with the conclusions and future work.

Chapter 2

Zigbee stack overview

Zigbee Alliance represents an association of companies that work together on the standardization of the Zigbee protocol suite based on an open global standard. Zigbee is intended as a cost-effective and low power solution for several markets including home automation, personal health care monitoring, smart energy, building automation, and wireless sensor networks. It is designed as a low data rate radio that operates in the industrial, scientific, and medical (ISM) radio bands of 2.4 GHz in which other radios like Bluetooth, microwave, and Wi-Fi also operates [27]. The software architecture of any Zigbee network comprises four basic stack layers: Application layer, Network layer, MAC Layer, and Physical layer as shown in Figure 2.1.

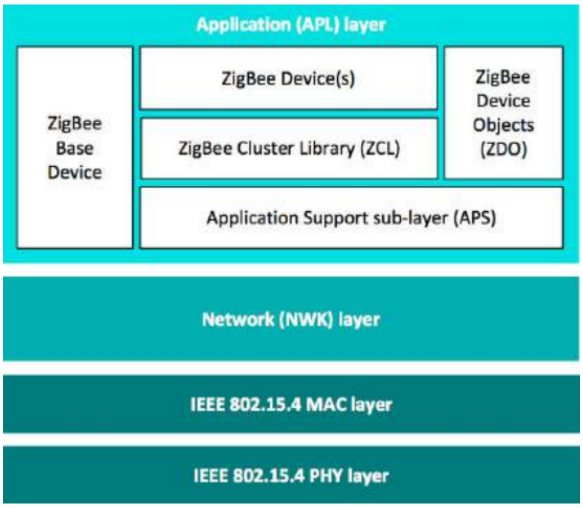


Figure 2.1: **Zigbee Stack Architecture**[**38**]

.

The PHY and MAC layer of Zigbee is built on top of the IEEE 802.15.4 standard. The network layer specification and the application layer framework to build up the Zigbee protocol is provided by the Zigbee Alliance. The application support sublayer (APS) and the Zigbee device objects (ZDO) together form the application layer framework. Unique application objects can be defined by end manufacturers, which use the application layer framework and share APS services with the ZDO. The Zigbee device type (Router, Coordinator, or End Device) is represented by the ZDO. It is responsible for maintaining track of device roles, device discovery, managing requests to join a network, and security [57].

2.1 IEEE 802.15.4 PHY Layer

The PHY layer is the first layer in the Zigbee stack and provides the data transmission service by managing the physical RF transceiver. It has multiple responsibilities such as channel selection, link quality indication (LQI), and energy measurement. Globally it can operate on the following unlicensed frequency bands-

1. 868.0-868.6 MHz: Europe, allows one communication channel
2. 902-928 MHz: North America, up to ten channels (2003), extended to thirty
3. 2400-2483.5 MHz: worldwide use, up to sixteen channels [1]

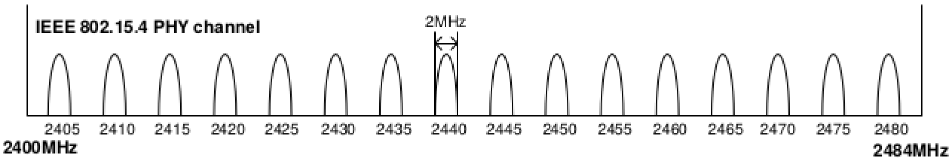
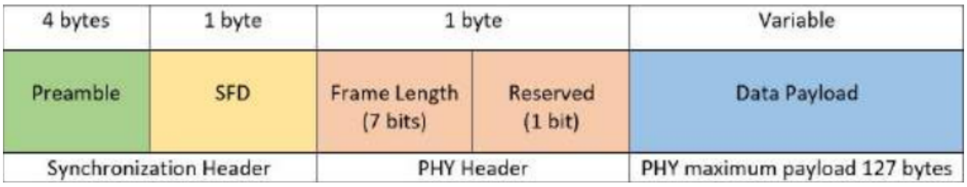


Figure 2.2: **Zigbee channels in the 2.4 GHz band**

As shown in Figure 2.2, the worldwide 2.4 GHz band is split in a total of 16 individual Zigbee channels. Each channel has a bandwidth of 2 MHz with 5 MHz inter-channel spacing and can transfer up to a maximum of 250 Kbps. The central frequencies of these channels are given by (in MHz)-

$$F_{\mathrm{c}} = 2405 + 5 \times (k - 11) \ for \ k = 11, 12, \ldots, 26 \qquad (2.1)$$

They use direct sequence spread spectrum (DSSS) with offset quadrature phase-shift keying for modulation of the signals[1].



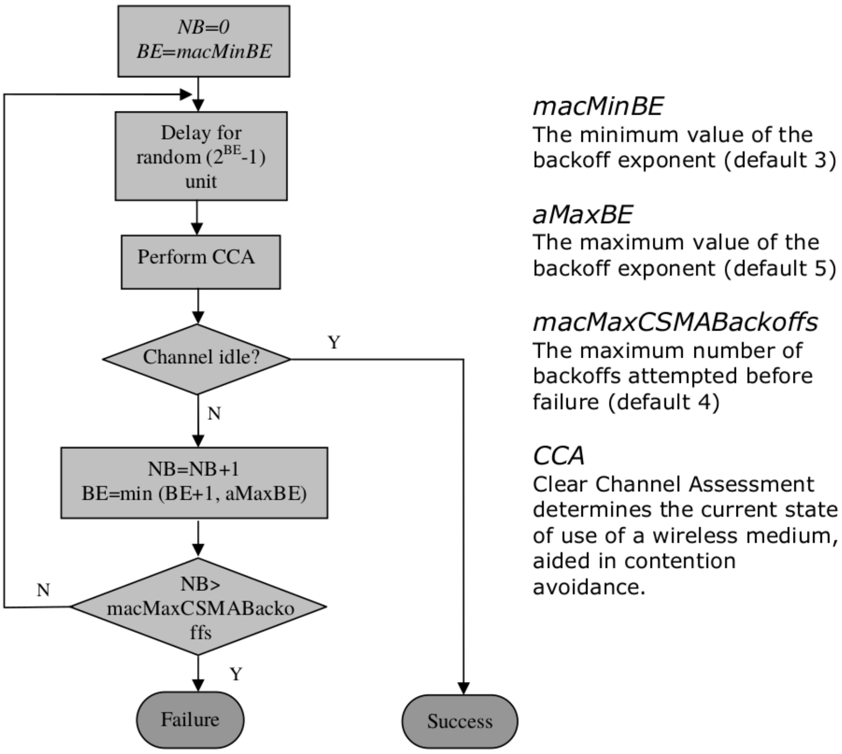
4 bytes	1 byte	1 byte		Variable
Preamble	SFD	Frame Length (7 bits)	Reserved (1 bit)	Data Payload
Synchronization Header		PHY Header		PHY maximum payload 127 bytes

Figure 2.3: **IEEE 802.15.4 frame format**

The PHY layer packet typically consists of a Physical Service Data Unit (PSDU) which can deliver a maximum payload of 127 bytes along with Synchronization and Physical headers. As shown in Figure 2.3, the synchronization header consists of a Preamble and an SFD (Start Frame Delimiter). The preamble allows the devices in the network to synchronize easily with their receiver clocks and the SFD is used to mark a new incoming frame.

2.2 IEEE 802.15.4 MAC Layer

The IEEE 802.15.4 MAC sublayer is not solely responsible for assembling and disassembling of data frames but also for congestion control. It also determines the addressing of data frames. It possesses mechanisms for joining and forming a network, listening for a clear channel, and a link layer to handle acknowledgments and retries to provide reliable end-to-end communication. The MAC layer defines two channel access mechanisms-



macMinBE
The minimum value of the backoff exponent (default 3)

aMaxBE
The maximum value of the backoff exponent (default 5)

macMaxCSMABackoffs
The maximum number of backoffs attempted before failure (default 4)

CCA
Clear Channel Assessment determines the current state of use of a wireless medium, aided in contention avoidance.

Figure 2.4: **Complete Unslotted CSMA/CA Mechanism [35]**

1. Unslotted CSMA/CA mechanism in the non-beacon enabled network

2. Slotted CSMA/CA mechanism in the beacon enabled network

In the popular non-beacon enabled mode, a node simply transmits its packet, whenever it has some data to send. It uses unslotted CSMA/CA as the mechanism for medium access. As shown in Figure 2.4, whenever a node has to transmit a packet, it first waits for a random amount of time and then senses the channel for a fixed amount of time (CCA). The symbol time and CCA time for Zigbee are 16 μs and 128 μs respectively[1].

Two variables called NB and BE are maintained by every node for each transmission attempt. NB represents the number of backoffs the CSMA/CA algorithm was required to perform while attempting the current transmission. It is initialized to zero before every new transmission attempt. If the channel is deemed to be free and the algorithm ends in success, the MAC is allowed to undertake transmission of the packet. Otherwise, the algorithm terminates with a CCA failure.

2.3 Zigbee Device Types

The Zigbee specification supports a single network with a maximum of one coordinator, multiple routers, and multiple end devices. The following section describes these node types -

1. Zigbee Coordinator (ZC): This device is responsible for establishing a centralized network. This node is a router with other responsibilities such as - choosing the appropriate frequency channel for the whole network and selecting a PAN ID (a unique and common identifier among all devices on the same Zigbee network). If multiple Zigbee networks operate within the range of each other, each should have a unique PAN ID. The Zigbee Coordinator also works as the trust center by providing authorization functionality to the whole network. It also solves network issues such as PAN ID conflicts or channel changes due to interference.

2. Zigbee Router (ZR): This device is utilized to provide routing services to network devices. Apart from merely receiving messages, they can relay data from other devices. This relay mechanism establishes a mesh of router nodes which increases the reachability and robustness of the network. They are not designed to sleep and remain on as long as the network remains established.

3. Zigbee End Device (ZED): This device includes just enough functionality to talk to its parent node (either the Zigbee Coordinator or the Zigbee Router) and cannot relay data from other devices. They are perceived as leaf nodes and could be sleepy or non-sleepy end devices[54].

2.4 Zigbee Network topology

Zigbee networks supports three types of topologies-

1. Star: Only two types of devices are present in star topology - Coordinator and the End Device, with the Coordinator node being the hub of all communications. This constructs a centralized network in which every data packet traverses through the Zigbee Coordinator. Star topology is elementary and easy to deploy, however, the coordinator can become bottlenecked with network processing. The communication range of the network is also limited to the range of the coordinator and there is no alternative path between different end devices. It is extremely limited in

terms of robustness as the Zigbee coordinator can act as a singular point of failure.

2. Mesh: Mesh supports all three device types. A mesh network consists of one Coordinator device, one or multiple Router and End Devices. Coordinators and Routers can both act as parent nodes and can have child nodes whereas end devices cannot have child nodes. In this topology, every Router can communicate with every other Router in its radio range as shown in Figure 2.5. They maintain multiple paths to other nodes which makes it very robust against route failures. Generally, a Zigbee network is deployed in this topology due to its reliability and the ability of self-healing in case any link failure occurs. Also when some devices are powered off, new routes to remaining devices are automatically discovered. This topology also scales well as the range of the network can be increased by adding more router devices.

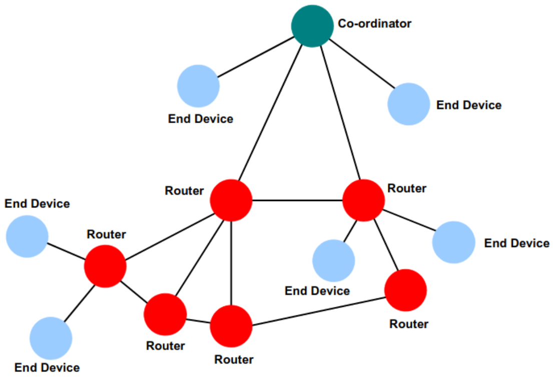


Figure 2.5: **Mesh Topology [38]**

3. Tree: It is the combination of Star and Mesh topology. In this topology, the network consists of a coordinator node, several routers, and end devices. The end devices connected to the coordinator or the routers are called child devices. Only routers and the coordinator can have children. Each end device is only just able to communicate with its parent (coordinator or router)[3].

Chapter 3

Coexistence Issues in the ISM band

The Industrial, Scientific, and Medical (ISM) unlicensed frequency spectrum is extremely popular with wireless radio technologies such as Zigbee, Wi-Fi, Bluetooth, microwave oven, cordless phone, and even LoRa [45].

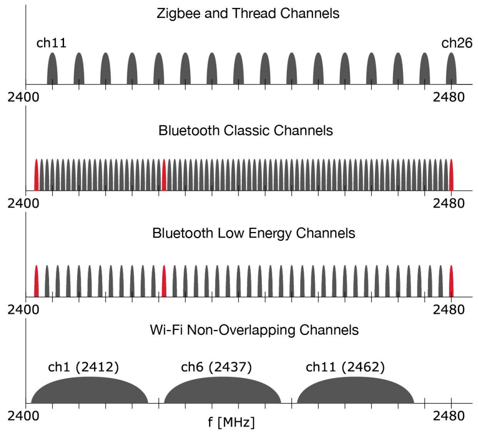

Zigbee and Thread Channels
ch11
ch26
2400
2480
Bluetooth Classic Channels
2400
2480
Bluetooth Low Energy Channels
2400
2480
Wi-Fi Non-Overlapping Channels
ch1 (2412)
ch6 (2437)
ch11 (2462)
2400
f [MHz]
2480

Figure 3.1: **Channels of radios operating on the 2.4GHz band** [48]

As multiple radios share the same frequency spectrum it can inevitably lead to unwanted interference and potential performance degradation for competing technologies. Among these, Bluetooth typically divides the whole spectrum into 79 channels with 1 MHz bandwidth and channel separation of 1 MHz. On the other hand, Bluetooth Low Energy uses 2 MHz spacing between its channels, which accommodates 40 channels. Both of these radio technologies operate three channels as advertising channels which are highlighted in red color in Figure 3.1, and the rest of the channels are used for data transmission. Bluetooth can support a data rate of 3 Mbps and boasts a range of 100m. BLE sacrifices range (50m) and data rate (0.27 Mbps) for significant savings in power consumption

(1% of Bluetooth). It targets low power devices that do not need high data rates or constant data transfer.

The Wi-Fi standard splits the spectrum into 14 channels with each channel providing a bandwidth of 22 MHz. The channel separation is 5 MHz which results in adjacent channel overlap leading to signal interference between these channels. As a result of this frequency overlap, a maximum of three non-overlapping channels could be used.

Out of the possible combinations, the specific set of channels 1, 6, and 11 is most widely used [15]. Wi-Fi typically has a high transmit power of 20 dBm and can support an effective range of 100m. Some Wi-Fi standards like IEEE 802.11n (2.4 GHz) can theoretically reach soaring data rates of 300 Mbps using high modulation coding schemes and MIMO techniques. However, in an indoor environment, it typically supports a third of the theoretical value due to interference from other sources which is still significantly higher than the max bitrate of Zigbee (250 Kbps) [46].

Among the mentioned radios, Bluetooth has little impact on other wireless networks as it uses a frequency hopping mechanism that enables it to perform a maximum of 1600 hops per second based on a pseudorandom sequence and hence mitigate interference[20]. However, Wi-Fi with its high data rate and high transmit power causes the most interference for Zigbee networks [11].

Wi-Fi is very mainstream and is broadly utilized in homes because of the accessibility of numerous gadgets that can support high bitrates. The interest for multimedia content has expanded in light of the prevalence of streaming services like Netflix, YouTube, Amazon Prime Videos, and so forth. Users additionally demand videos in HD which further builds the requirement for a high bitrate. Time bounded applications like video conferencing, Voice over IP (VoIP), and web-based gaming, etc. require high throughput and low latency for fulfilling their required QoS[2]. Zigbee, on the other hand, targets low data rate applications like home automation and monitoring applications. They do not require too much power or bandwidth. However, since they are employed for control applications, they do require high reliability and low latency. These situations further complicate the coexistence of Zigbee networks present alongside Wi-Fi networks. Since Zigbee networks are most vulnerable to Wi-Fi interference, we will focus on their coexistence in this work. In the following sections, we will carefully look at how Wi-Fi networks can affect Zigbee networks when they are present in the same environment.

3.1 Zigbee and Wi-Fi Overview

IEEE 802.11 standard defines the MAC and the PHY layer for wireless LANs. Similar to the IEEE 802.15.4 standard, the IEEE 802.11 standard also employs CSMA/CA mechanism. Before starting its transmission, the IEEE 802.11 node tunes in to the channel to determine whether the channel is idle. In the event that the channel is detected inactive for a DIFS time stretch, the node will continue with its transmission. On the off chance that the channel isn't idle, the node concedes the transmission. When the channel is sensed idle for a DIFS time span, the node generates a random backoff delay which is uniformly chosen in an interval. This interval [0, W] is called Contention Window, where W represents the size of the window and is initially set to $\mathrm{CW}_{\mathrm{min}}$. The backoff delay is diminished by one unit as long as the channel is detected inactive for a backoff time slot. If the node senses the channel to be busy, the backoff timer is frozen and resumed when the channel is sensed idle again for a DIFS duration. This is performed to keep the sensing time to a minimum. The node transmits a data packet when the backoff timer arrives at zero. In the wake of receiving the data packet, the destination node sits tight for a SIFS span and afterward transmits the ACK back to the source node.

A considerable difference between the CSMA/CA mechanism for the IEEE 802.11 and IEEE 802.15.4 standards is that the IEEE 802.15.4 node does not sense the medium during the backoff period but only during the CCA period. The contention window in the IEEE 802.15.4 standard is doubled whenever the medium is determined busy during a CCA period. However, in the IEEE 802.11 standard, the contention window is the same when the channel is determined busy[61]. This difference has a major impact on their behavior when they share an overlapping channel. In the following sections, we will look at how these parameters affect the two standards when they are present in the same environment.

3.1.1 Frequency Overlap

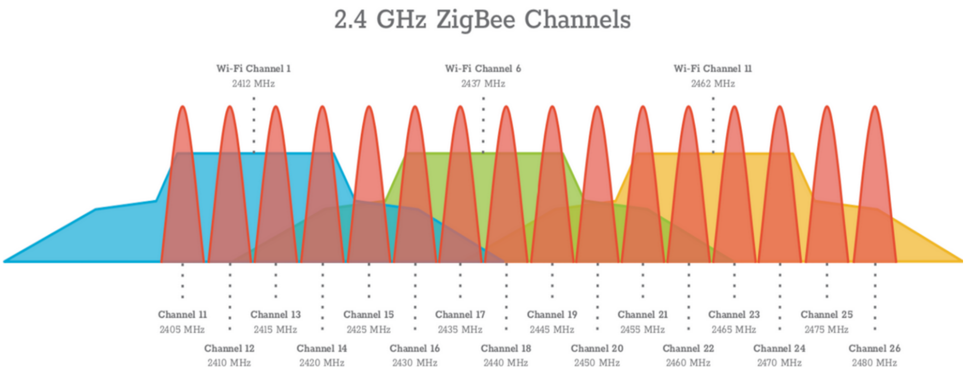


Figure 3.2: **Frequency Overlap Wi-Fi and Zigbee Channels** [33]

A Wi-Fi network can operate in any one of the 14 defined channels in the 2.4 GHz spectrum. Each channel is 22 MHz wide whereas in IEEE 802.15.4 every channel is 2 MHz wide. Because of large bandwidth, many channels in the IEEE 802.11 standard interfere with each other. Ideally, Wi-Fi has three non-overlapping channels namely Channel 1, 6, and 11. Three IEEE 802.11 networks working in close proximity typically operate on these channels to prevent interference among themselves. As evident from Figure 3.2, IEEE 802.15.4 networks can exclusively operate on Channels 15, 20, 25, 26 to prevent interference from Wi-Fi networks operating on popular channels. These channels are present in the Wi-Fi guard bands and receive little interference from Wi-Fi networks operating on three non-overlapping channels. This leaves IEEE 802.15.4 networks with very little choice and underutilization of the frequency spectrum.

3.1.2 Power problem

There is a significant difference between the transmission powers of Wi-Fi and Zigbee nodes as shown in Table 3.1. Due to this large power difference, three regions can develop-

1. R1: Zigbee nodes and Wi-Fi nodes are placed in close proximity and both can sense each other
2. R2: Zigbee node can sense the Wi-Fi node due to its high power but not vice versa
3. R3: Neither Zigbee nor Wi-Fi node can sense each other, however, Zigbee node could still suffer interference from Wi-Fi

Parameters	Wi-Fi (IEEE 802.11n)	Zigbee (IEEE 802.15.4)
Transmit Power	20 dBm	0 dBm
Number of RF channels	14	16
Bandwidth	22 MHz	2 MHz
Transmit rate	$\leq$ 260 Mbps	250 Kbps
CW_{min}	15	7
CCA duration	N/A	128 μs
Backoff unit T_{bs}	9 μs	320 μs
SIFS	10 μs	192 μs
DIFS	28 μs	N/A

Table 3.1: **Wi-Fi and Zigbee parameters.**

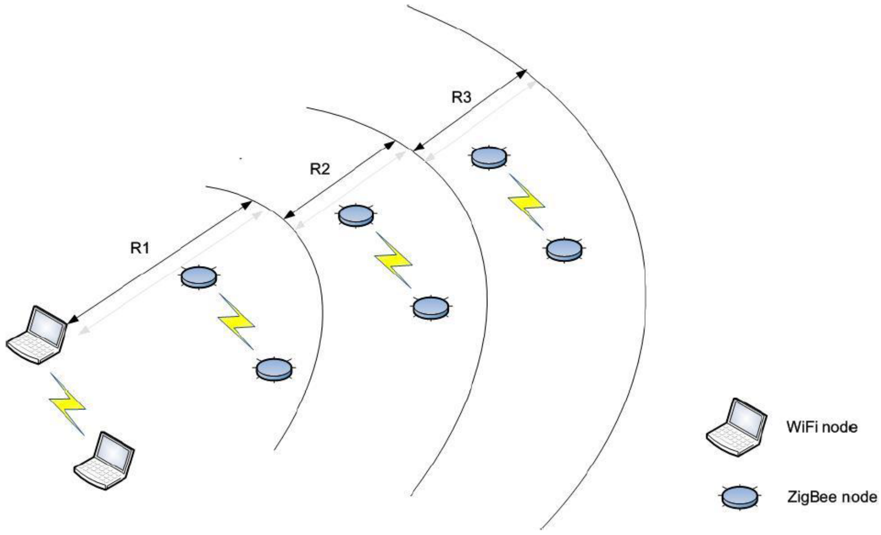


Figure 3.3: **Different regions due to power asymmetry.**

The three regions are depicted in Figure 3.3.

In Region R1 and R2: a Zigbee node would suffer from an increased number of backoffs and CCA fails during Wi-Fi interference. As the channel sensing time of Zigbee is considerably higher than Wi-Fi, this will lead to increased latency due to multiple backoffs. The Wi-Fi network would saturate the channel and give Zigbee network very little chance to successfully access the channel. In Region R2 and R3: a Zigbee node would also suffer from packet collisions as the Wi-Fi node is unable to sense the presence of the Zigbee node due to its weak signal strength. There would be scenarios in which the Zigbee node wins the channel but since the Wi-Fi node cannot sense the presence of the Zigbee network, it will still transmit and a collision will occur, reducing the throughput for both networks.

3.1.3 Timing problem

As discussed in Section 2.2, in order to reduce the chance of collision, both Zigbee and Wi-Fi use the CSMA/CA scheme to contend for the medium. It can also be seen from Table 3.1 that Wi-Fi has an advantage over Zigbee as it spends less time to deem the channel idle.

The shorter time interval gives Wi-Fi nodes priority over Zigbee nodes to get access to the channel and therefore cause unfairness to the Zigbee nodes as illustrated in Figure 3.4.

A sufficient coexistence condition for this scenario, is that the CCA of the Zigbee node happens during the idle time between two consecutive Wi-Fi packets [61]. To satisfy this condition, the idle time needs to be,

$$\mathrm{t_{idle}} \geq \mathrm{DIFS} + \mathrm{m} \times \mathrm{T_{bs}} \tag{3.1}$$

where m is an irregular number drawn from a uniform distribution over the stretch $[0, \mathrm{CW_{min}}]$ and $\mathrm{T_{bs}}$ is a backoff unit for Wi-Fi. According to Equation 3.1, the minimum value of m for the Wi-Fi node must be 12 to let the Zigbee node perform a successful CCA. As the maximum value of $\mathrm{CW_{min}}$ is 15, this leads to a very limited number of combinations [12,15] for Zigbee node to have a successful CCA. The situation is different in the case of region R2. Due to

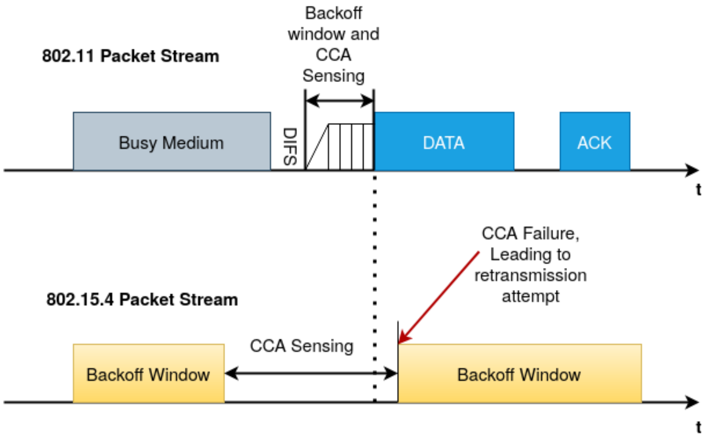


Figure 3.4: **Channel sensing in R1.**

asymmetry in power, the Wi-Fi node can't hear the Zigbee node but not vice versa. During the transmission of the Wi-Fi node, the Zigbee node defers its transmission. However, the Wi-Fi node doesn't defer its transmission during Zigbee transmission, as the signal strength is too weak and the Wi-Fi node treats it as noise. One such scenario is shown in Figure 3.5. This situation leads to packet collision even though the Zigbee node acquired the channel and started its transmission first.

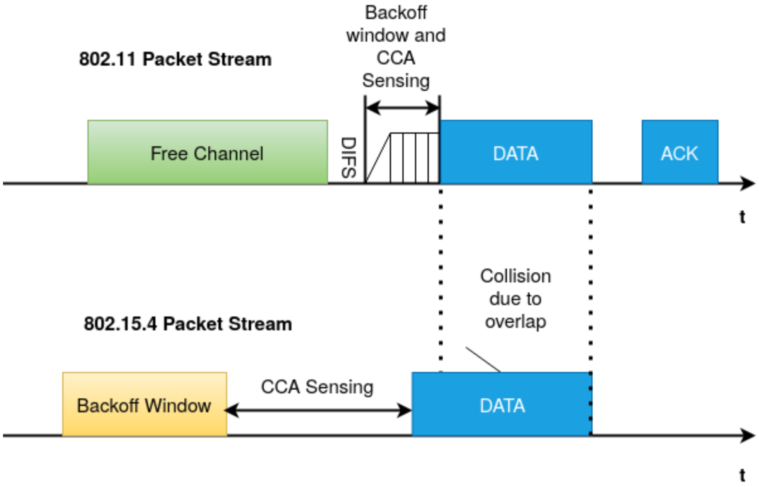


Figure 3.5: **Potential packet collisions in R2.**

In the region R3, since neither node can sense each other, they both transmit freely which can be called blind transmission. However, the large distance between the two nodes makes it hard for either node to corrupt the transmission of the other node, but it is still possible.

As seen from our above discussion, the specifications do not provide a level playing field between Zigbee and Wi-Fi technologies. Low transmit power and high channel sensing times cause Zigbee packets to get delayed or even get lost due to collisions, which decreases its reliability. In the next section, we will review some models which are developed in the literature to help Zigbee mitigate Wi-Fi interference.

3.2 Related Work

Due to the ubiquitous deployment of Wi-Fi and Zigbee networks in daily life, coexistence between them presents a critically significant problem. As we observed from the previous section, Zigbee networks are extremely vulnerable to Wi-Fi interference. The extensive use of high data rate Wi-Fi applications complicates the situation for low duty-cycle Zigbee networks. Both Zigbee and Wi-Fi networks are usually deployed in close proximity to each other, e.g., inside a residential building. The interference between Wi-Fi and Zigbee networks has been extensively studied in both industry and academia. Over the years, many models and possible solutions have been developed in the literature. This section lists major existing Zigbee Wi-Fi coexistence mechanisms and how they typically address this challenge.

Zigbee networks rarely suffer from possible collisions with Wi-Fi under light Wi-Fi interference and can easily recover the loss by means of retransmission[13]. However, Zigbee's performance is severely degraded within the sight of moderate to high Wi-Fi interference as shown in multiple measurement studies [42, 52]. Zigbee can also suffer up to a loss of 85% under Wi-Fi interference even when carrier sensing and packet retransmissions are enabled[14]. On the other hand, Zigbee causes almost no interference to Wi-Fi networks since it targets low duty-cycle applications with low channel occupancy[40].

A trivial way to mitigate Wi-Fi interference is to use a Zigbee channel which is not overlapping with the nearby Wi-Fi networks [59, 60, 43]. In one approach, the authors correlated Wi-Fi signal strength with the packet error rate of the Zigbee network. When the PER (Packet Error Rate) is greater than a certain threshold, the Zigbee network believes it's because of Wi-Fi interference and switches to a different operating channel to avoid it. This approach is practical in low-density environments where the Zigbee network and the Wi-Fi network can operate on free channels. However, it is not extremely effective due to the limited number of orthogonal Zigbee channels available in a densely deployed Wi-Fi environment. As Wi-Fi networks try to operate on orthogonal channels to minimize interference amongst themselves, it leaves the Zigbee network with little or no choice for a free channel. Wi-Fi traffic is generally bursty in nature i.e. it tries to transmit at a high data rate over a short period and then slows down. This mechanism can also be seen while streaming movies/YouTube as these applications buffer the data for the next 30 seconds in advance. Due to this distribution, white spaces get created between these transmissions in which the Wi-Fi node is not transmitting anything or transmitting at a very low bandwidth. If properly tuned, these white spaces can be used by Zigbee networks to transmit their packet with little interference. Authors of [19], show that the CSMA mechanism is surprisingly inadequate to exploit the white spaces present in Wi-Fi transmissions. They analyzed the Wi-Fi network traffic pattern and developed a Pareto model to identify white spaces in Wi-Fi traffic. They changed the Zigbee frame adaptively according to the estimation of the idle interval between Wi-Fi transmissions. This approach prevents collisions as it targets free slots in Wi-Fi transmission, however, it is unsuitable for delay-sensitive applications as Zigbee transmission is stopped during Wi-Fi bursts. A comparable methodology was introduced in [12] to exploit the quiet time frames during which Wi-Fi nodes are inert for Zigbee transmissions.

Since Zigbee and Wi-Fi both use different standards, it becomes challenging for one radio to accurately differentiate other's signals from noise. In [63], the authors developed a framework called ZiFi, using Zigbee radios to identify Wi-Fi signals from noise signals. Since Wi-Fi networks periodically send beacon packets, the authors developed a digital signal processing algorithm that can amplify these obscure intermittent signals in received signal strength. They achieved high accuracy in detecting Wi-Fi signals. However, the authors didn't address the interference mitigation part. A similar approach was presented in [56]. This method employs a special sensing engine either on Zigbee or Wi-Fi or both sides which enables Wi-Fi to detect Zigbee signals. The main drawback of this approach is it increases the complexity of the system by introducing additional hardware.

In [62], the authors presented the idea of a separate Zigbee node called the signaler that possesses more transmission power than ordinary Zigbee nodes. The signaler is deployed close to the Wi-Fi network and is designed to emit a busy tone to inform Wi-Fi nodes when the Zigbee network possesses data to transmit. This makes certain Wi-Fi nodes can also sense a busy tone before data transmission and prevent collisions. This mechanism can effectively coordinate between the two networks. In [18], the authors integrated an 802.11 chip on a Zigbee enabled medical device. Before transmitting a Zigbee packet, the 802.11 chip sends an 802.11 RTS packet to reserve the channel. This prevents nearby Wi-Fi nodes from transmitting and prevents collisions. However, in these two methods, a separate node is needed and also fairness remains an issue for the Wi-Fi network in cases of sustained busy tones.

Models such as [29, 16, 17] also exist that are able to extract valuable information even after packet collisions. The authors argued that on many occasions the only part of a Zigbee packet to be corrupted is the header. They designed the system called BuzzBuzz which uses multiple headers and forward error correction (FEC) to encode the Zigbee packets. They showcased that data can be successfully recovered even if some data errors occur during the transmission. This solution improves the robustness of the Zigbee network. However, the addition of coding technology at both the transmitter and the receiver also increases the overhead. One recent approach proposed a frequency overlay technique called COFFEE (COexist wiFi For zigbEE networks), in which the authors nullified the Wi-Fi subcarriers interfering with Zigbee transmissions [28]. They blocked the subcarriers corresponding to the 2 MHz channel that the Zigbee network was currently using to provide it an interference-free environment. The rest of the channel was used for Wi-Fi transmission. The results indicate an increase in the throughput for the Zigbee network, however, the Wi-Fi throughput is decreased by more than 15%. As there is no coordination between the two networks, the subcarriers are required to be consistently nullified even when there is no Zigbee transmission which causes unfairness to the Wi-Fi network.

To overcome the shortcoming of the IEEE 802.15.4 standard and mitigate interference from Wi-Fi devices, most of the mentioned models either make hardware/MAC changes or introduce an extra node in the network. The secondary node assigns Zigbee priority by cutting off Wi-Fi transmission completely. These methods ensure reliable transmission for Zigbee networks but add complexity and cost to the system. They also don't address the data loss for Wi-Fi networks. Many Wi-Fi applications like VoIP, online gaming require no data loss and low latency. As their transmission is stopped, they no longer can adhere to their QoS requirements, and their quality drops. This produces an extremely detrimental effect on the experience of the end-user.

Chapter 4

Smart Lighting System

Smart lighting encompasses various technologies like LEDs and OLEDs to illuminate indoor and outdoor environments. In the traditional lighting system, the lights need to be controlled manually. However, smart lighting systems allow the lights to be controlled remotely. They can be operated through the means of a remote controller like a mobile phone or a web browser. They can be employed to adjust the intensity of light in a room when an external event occurs like the detection of a person using a motion sensor. Different communication technologies like Zigbee, Bluetooth, or Wi-Fi can be used with smart lighting. The performance metrics of different lights can also be extracted from these systems which open ways for complex data processing and analysis. Such systems organized as lighting networks allow diverse types of lights to interact with each other. Smart lighting provides an efficient management system and can be implemented to make complex lighting applications. One such system is the Philips Hue system described in the following section.

4.1 Philips Hue System

Signify (formerly known as Philips Lighting) uses Zigbee for several wireless lighting propositions, both in consumer (Hue) and professional (Interact) systems. Philips Hue ecosystem is an IoT lighting platform containing four core components: LED lights, a bridge with internet connectivity, apps for smartphones, and web service with an open API to send and receive commands. The Hue bulbs contain three types of RGB LEDs that can produce an array of colors and intensities. The lights are deployed in a mesh topology that enables each light to relay messages to nearby lights making the system more robust and extending its range. Internet connectivity is provided to the system through a gateway that supports the Ethernet stack and the Zigbee protocol Zigbee. The Zigbee standard makes sure lights from other manufacturers can additionally be utilized with the Philips Hue system. Zigbee defines multiple device types, including lights devices, sensors, and switch devices and controller devices. The light devices include on/off light, color light, dimmable light, extended color light, etc. The controller devices may include light switches, remote control unit(s), computing devices, or smartphones. Zigbee networks utilize 16-bit network addresses to identify devices [58]. Groupcasts can be used by a controller to address a subset of devices, e.g., a groupcast to turn on all lights in a room. Apps can control the lights using a RESTful API over the HTTP(S) protocol. Solitary or multiple lights can be controlled by the interface. Each light is assigned a unique local address that can be used to interact with the light. Various parameters of the light like its state, color, brightness can be retrieved or changed by simply making an appropriate GET or PUT request respectively. [50] [51]

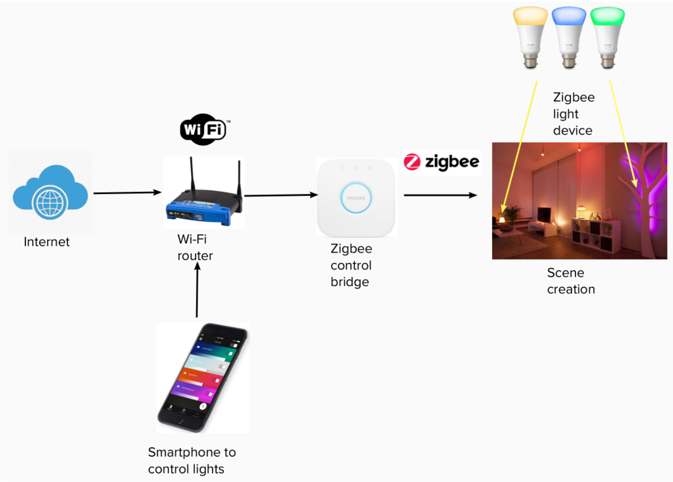


Figure 4.1: **Philips Hue based Wireless Lighting System. Modified from [57]**

A simple Philips Hue system is illustrated in Figure 4.1. The Philips Hue Bridge works as the Zigbee control bridge which is the central controller of the Zigbee network. The Hue bridge is connected to the Wi-Fi AP/router via an Ethernet connection. Single or multiple light devices from the same or different manufacturers can be connected to the Hue bridge via Zigbee. The lights can be deployed either as a Zigbee End Device or Zigbee Router. However, they are typically deployed as a Zigbee Router as it increases the robustness of the system by relaying messages to other lights within their reach. Zigbee provides the possibility to connect and control home lighting from the internet. The Zigbee standard not only allows to control the on/off (start) of the smart lights but also to control the color temperature, hue, saturation, and brightness. The settings can be controlled either through smartphones, tablets, or PCs. All of these messages travel through the router to the Philips Hue bridge which broadcasts or unicasts the message to the target node.

4.2 Hue Beta Program

Data collection and analysis are essential in any industry, more so for companies focusing on IoT applications like a smart lighting solution. It provides practical insights by carefully monitoring the performance of the system. It is employed to extract valuable data like usage patterns of users which can be used to improve their experience and provide better service to them.

Signify conducts a program called Hue beta program to test its new products before its release to the public. Many testers including its employees take part in this program. In this program, useful data is retrieved from these home installations of the Philips Hue system (with the consent of the participants in the beta program). Some of the relevant parameters are extracted from these systems and stored in a cloud back-end for future processing and data analysis. These datasets include general parameters and Zigbee related parameters. General parameters include the current state of the light (on/off), brightness level, hue, saturation, name of the light, manufacturer name, number of lights in the whole system, etc. Zigbee parameters include the current Zigbee channel, packet reception rate for all devices, unicast fails and multicast fails. All of these parameters are stored in the ElasticSearch database [8] and Kibana [9] is used to make meaningful dashboards to visualize the data. These parameters are important as they can be used to assess the performance of the system in real-time. It is extremely useful in the case of troubleshooting a problem as they could indicate what could have went wrong. The visualization help in remote monitoring and provide instant feedback in the case of a change in the system.

Some participants of the Hue beta program complained about the unreliable performance of the system, especially during the evenings. The complaints included delays in response from the light bulbs or in the worst case no response i.e. the packet was lost. To better understand the performance of the system on a personal level it was decided to use the dataset to investigate a correlation between the inferior performance and the parameters. One of the parameters that have been extensively used in this work is the ratio parameter which is defined as -

$$Ratio\ (\%) = \frac{Successful\ Zigbee\ Requests}{Total\ Requests} \times 100 \tag{4.1}$$

It is an indicator of the reachability of the system as a whole. The Hue bridge collects this data from every light on an hourly basis and forwards it to the database. The number of successfully received packets is compared with the number of total packets sent towards each light and then aggregated for the entire network. It is further analyzed to see any drops in performance. The variation in the value of the ratio parameter can be used to extract the data pattern of the user. One such data extract from a user is shown in Figure 4.2.

This system consisted of 20 lights, therefore, the maximum count of 100% ratio for any hour can be 20. Here the number of nodes that had a ratio of 100% is displayed on the Y-axis with their timestamp on the X-axis. One observation that could be made is the ratio parameter goes down a little in the morning and takes a plunge during the evening. These drops in performance can be correlated to the daily usage pattern of a house. As Wi-Fi can interfere with Zigbee if they both operate on an overlapping channel (refer Section 3.1.1), we might expect a drop in the evening as the user would be at home during this time and might extensively use Wi-Fi devices which could interfere with Zigbee transmission of the Hue bridge. Some video streaming applications like Netflix and YouTube could require a very high bandwidth of 25 Mbps especially for HD videos [36]. These could severely harm the Zigbee network due to high interference from the Wi-Fi network.

After a general analysis of the data for multiple users, it was decided to perform a survey to request the testers to provide information about the per-

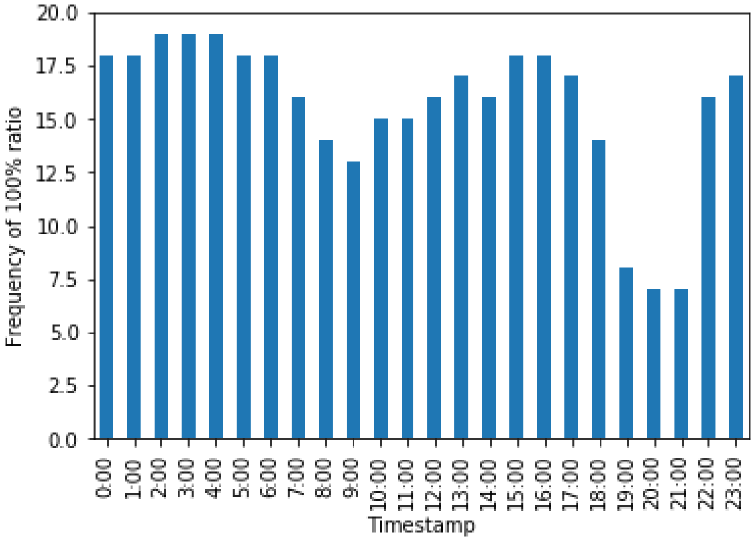


Figure 4.2: **Frequency of 100% ratio parameter for the whole system on an hourly basis.**

formance of their Philips Hue system. Hundreds of users from the Netherlands responded about issues like delay and reachability of the system. It is important to note the Philips Hue systems that were studied in this survey belonged to the beta users who reported reachability-related issues which represents a very small fraction of the total Hue users. Also, some users reported they were happy with the performance of their system and require no changes. For the intended purpose of confirming the hypothesis that performance degradation was indeed caused by Wi-Fi interference, the active participants were invited to provide Wi-Fi screenshots from NetSpot[1] . Screenshots were taken at the problem areas like near the Hue bridge and the unreachable lights. The provided screenshots were carefully cross-checked with the stored data in Kibana for potential frequency overlap with the operating Zigbee channel. In the specific case of channel overlap with Wi-Fi, the users were urged to move to a new Zigbee channel which was at least 25 MHz away from all nearby Wi-Fi channels. The frequency offset of 25 MHz was suggested as a safe separation for no packet loss in [13]. Hence, if there were nearby Wi-Fi networks that were operating on channels 1 and 6, the participant was requested to change the Zigbee channel to 25 to mitigate interference. The performance was assessed again in a few days. The results indicated a considerable increase in the reachability and performance of the system. One such example is demonstrated in Figure 4.3. We can observe an increase in the total number of 100% ratio counts from Day 8 which was the date of change of Zigbee channel. The maximum number of 100% ratio counts that could be achieved for this particular system is 480 (20 lights in total). This clearly shows an improvement in the performance of the system. These results provide further support for the hypothesis that indeed

[1] A software tool for wireless network assessment, wireless scanning and analyzing Wi-Fi networks. It can provide information on nearby Wi-Fi networks by reporting their operating channels and corresponding signal strength.

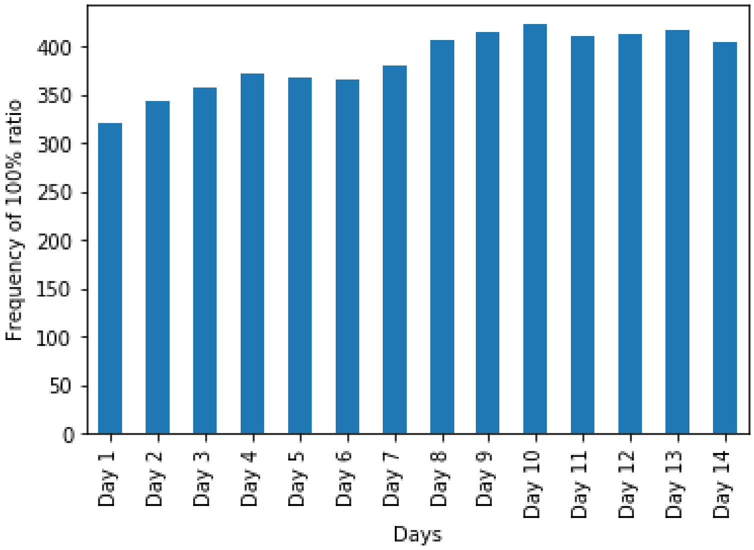


Figure 4.3: **Frequency of 100% ratio parameter for the whole system on a daily basis.**

Wi-Fi interference was the primary cause of the low reachability of nodes. Interestingly, it was also observed that about 50% of the total analyzed systems, were initially operating on an overlapping channel with Wi-Fi networks even when there was a free non-overlapping channel available for Zigbee. The results of this survey clearly show the underutilization of the frequency spectrum. This pattern could be attributed to the consumers using the same default Zigbee channel which gets set during the first installation. Since all consumers might not be tech-savvy to understand the inferior performance may be because of Wi-Fi interference on overlapping channels and moving to a free channel would mitigate interference. This indicated a need for a dynamic system that could scan for Wi-Fi networks and change the default Zigbee channel after detecting Wi-Fi interference, especially during the first installations.

4.3 Dynamic frequency selection system

As concluded from the previous section that Wi-Fi interference can carry out a critical role in degrading Zigbee performance, it was decided to develop an automated dynamic frequency hopping mechanism for the Philips Hue system that could move to a free channel under Wi-Fi interference. Because of different standards utilized by Zigbee and Wi-Fi, both networks can't decode each other's packets. Also due to low transmit power, the Zigbee network in many cases is invisible to the Wi-Fi network. Two approaches can be employed to combat this problem. To begin with, the Zigbee network could forward packets from one node to another and note the PER and latency. If the counters are above a pre-determined threshold, the Zigbee network would move to a new channel as also suggested in [60] and perform the same measurements until it satisfies the condition. A more accurate and easier method would be to use a wireless chip that supports Wi-Fi with the Zigbee network and operate it as a sniffer. The sniffer could be employed to detect the energy levels of different Wi-Fi networks on the basis of beacon packets. The Zigbee network could subsequently use this information and choose the best corresponding channel for its operation. Interestingly, the Hue Bridge Soc has a deactivated wireless function on its SoC. It is not really required as the bridge connects to the Wi-Fi router via an Ethernet connection. However, in this use-case, we propose using the wireless chip[2] in monitor mode to scan energy levels in each Wi-Fi channel to find out the presence of nearby Wi-Fi APs and their signal strength. Afterward, the Philips Hue bridge could move to a free corresponding Zigbee channel. This mechanism is extremely practical during the first installations as the system could find and move to a free channel. This system is also ideal in low-density environments where Wi-Fi and Zigbee could be synchronized to operate on non-overlapping channels as shown in Figure 4.4. The Figure demonstrates the presence of different Wi-Fi APs with their signal strength and operating channel along the Y-axis and X-axis respectively. It was plotted with the information extracted from the setup of a Philips Hue user.

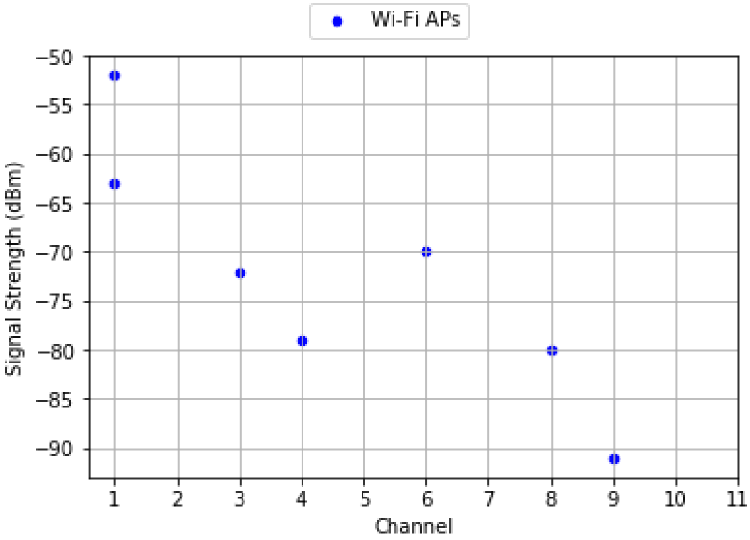


Figure 4.4: **Wi-Fi Channel distribution in a low-density environment.**

In this case, the primary cause of wireless interference would be the AP's operating at Wi-Fi channel 1 as they have the highest signal strength. We on top see that Wi-Fi channel 11 is free as no AP is operating on that frequency. By referring to Figure 3.1 we see that the Zigbee network can use channel 25 and coexist with the Wi-Fi networks in this environment. The Zigbee network would be closest to the AP on channel 9 but its signal strength is too low to cause any substantial interference.

However, this approach has one significant drawback. Because of the limited number of available free channels and the widespread deployment of Wi-Fi networks in residential buildings, it could become extremely challenging to find a

[2] Qualcomm QCA4531 Wi-Fi SoC

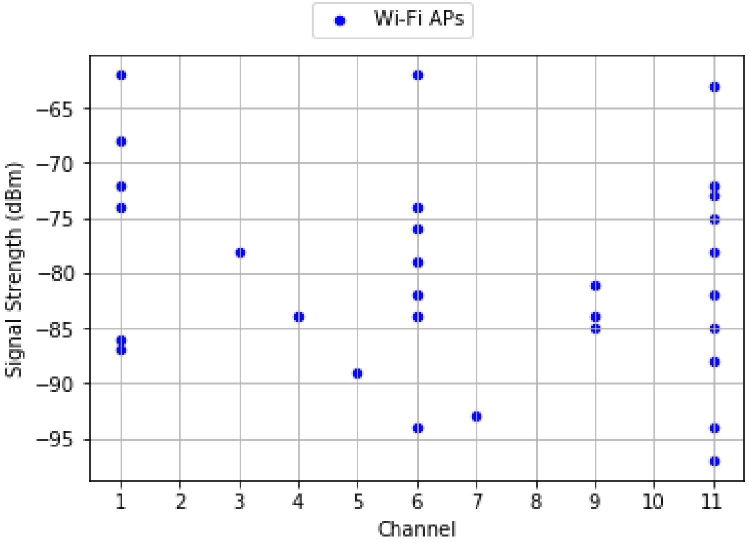


Figure 4.5: **Wi-Fi Channel distribution in a high-density environment.**

free channel. A sample use-case is shown in Figure 4.5. As we can observe there is no free channel for Zigbee to operate on and it can suffer from packet loss and delay if it operates in this environment. All three major Wi-Fi channels are taken by nearby Wi-Fi networks with high signal strength which leaves no space for the Zigbee network.

The method proposed above is efficient only for the 50% of the users of the survey who can move their Philips Hue system to operate on a free non-overlapping Zigbee channel. The other half of the users would not benefit from this approach as there is no interference-free Zigbee channel in the vicinity.

These results and observations motivate us to look at other possibilities in search of a viable solution. In the following chapter, we will introduce our system design which attempts to schedule transmissions in the time domain for both Zigbee and Wi-Fi by using a centralized approach. The central node provides reliability to the Zigbee network meanwhile keeping Wi-Fi loss to a minimum. Since the system design focuses on scheduling packets in the time domain, the Zigbee network can coexist with the Wi-Fi network even operating on an overlapping channel. As we have control over the main source of interference for the Zigbee network i.e. the Wi-Fi router, we assume that our system design will negate the detrimental effect of the Wi-Fi network on Zigbee transmissions.

Chapter 5

Providing reliability to a Zigbee network

High transmit power and low channel sense time of Wi-Fi already makes Zigbee vulnerable to interference. The problem intensifies in a home environment where the Zigbee network not only has to mitigate interference from the home Wi-Fi but also from neighboring Wi-Fi networks. As the different Wi-Fi networks try to operate on one of the three non-overlapping frequencies to prevent overlap between each other, this leaves Zigbee networks with very little choice of available interference-free channels.

As we saw in Section 3.2, to provide reliability for Zigbee transmission many former solutions introduce a gateway node. The gateway node either cuts off Wi-Fi transmission completely during Zigbee transmission or signals a busy tone which is strong enough to make the Wi-Fi node defer its transmission. However, this adds complexity to the system and causes unfairness to the Wi-Fi network. Some solutions try predicting the white spaces present between Wi-Fi transmissions and use that time for Zigbee transmission. However, since this is an estimation, it does not provide reliability to the Zigbee network and adds latency.

The most critical issue that these heterogeneous networks face represents the lack of coordination among themselves. Since they follow completely different standards they can't directly communicate with each other.

Utilizing a centralized network is extremely useful in a Wireless Sensor Network as different nodes can be synchronized by a common coordinator node that leads to effective control of the network. This node possesses a central view of the network which leads to more efficient control of the network. Its importance is considerably more significant in an environment where multiple technologies such as Wi-Fi and Zigbee try coexisting.

In this chapter, we will look at how we can operate a central node as a coordinator node for Wi-Fi and Zigbee network in a home environment. We will look at how IP packets are routed through a router and provide an overview of diverse queuing disciplines that can be applied at the egress interface of a router. Ultimately, we will introduce the working of our proposed system design. We propose a system design that manages the existing infrastructure to provide coordination between the Wi-Fi networks and the Zigbee network of a smart lighting system. The system design provides reliable transmission for Zigbee meanwhile keeping Wi-Fi loss to a minimum.

5.1 Packet routing in IP networks

In a home network, Internet services are provided by the use of a residential gateway which provides network access to LAN hosts by connecting them to a WAN. The WAN is principally operated by the ISP (Internet Service Provider). One such example is the wireless router which can function in a wired LAN, in a wireless-only LAN, or a mixed wired and wireless network. One or more wireless NICs are featured in almost all wireless routers which are either integrated into the main SoC or can be present as separate chips on the PCB.

A router may include multiple interfaces for diverse types of physical layer connections, like fiber optic, copper cables, or wireless transmission. Each network interface allows data packets to be forwarded from one transmission system to another. The interface receiving the data packets is called the ingress interface and the interface forwarding the data packets is called the egress interface. Other hosts can connect to the gateway to access the internet and form a Wi-Fi network.

A subnet is a logical subdivision of the network. Every computing device connected to the LAN interface of the router belongs to the same subnet. They share a common network prefix and have unique host identifiers. For example, 123.12.1.0/24 is the prefix of the IPv4 network starting at the given address, having the first 24 bits allocated for the network prefix, and the remaining 8 bits reserved for host addressing. This notation signifies addresses in the range 123.12.1.0 to 123.12.1.255 belong to this network.

The primary function of a router is to connect multiple networks and route packets destined either for other networks or their own network. It performs this by routing the packets based on the information by extracting the destination IP address from the IP packet. The router is considered a layer-3 device as its forwarding decision is based on the layer-3 IP packet.

"When a router receives a data packet on any ingress interface, it determines where to forward the packet by looking in the forwarding table for the best route to a destination. The router then forwards the data packet toward its destination through the appropriate egress interface"[24]. QoS policies can then be applied at the egress interface which can be used to prioritize, classify, or limit outbound traffic. These policies are necessary as they can be employed to prioritize traffic like VoIP to keep their latency as low as possible.

5.2 Coexistence of a smart lighting system with Wi-Fi in home networks

A fundamental representation of a home network utilizing a smart lighting system is presented in Figure 5.1. The Philips Hue bridge employs the router as a gateway to connect to the internet. Whenever a user requests to turn on/off the light or change its color, the message travels through the common network infrastructure and arrives at the router. The router then forwards the packet to the interface which connects to the Hue bridge (Wired LAN in this case).

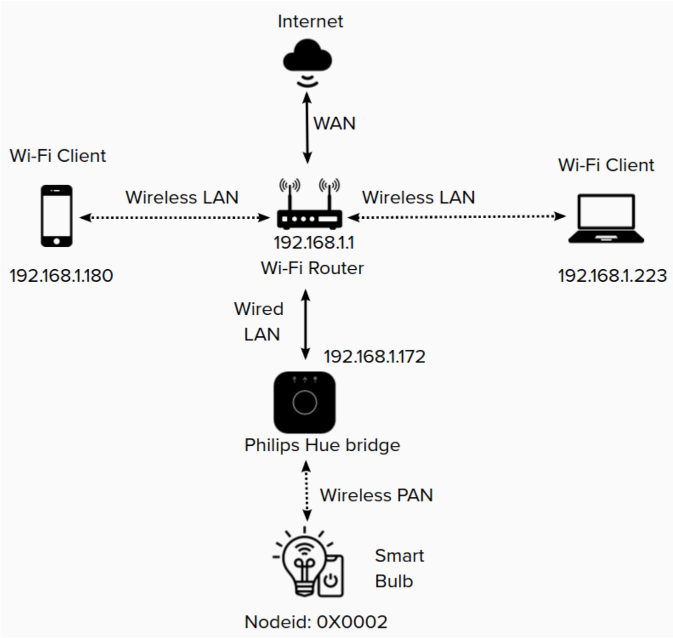


Figure 5.1: **Simple representation of a smart home using Philips Hue ecosystem.**

The Hue bridge then converts the contents of the packet to a Zigbee packet and transmits the message to the target node over Zigbee.

The router has its own pros and cons for the Zigbee network. It is essential for providing internet services to the Philips Hue bridge, which allows users to manipulate the lights with their smartphones/laptops. However, as the Philips Hue bridge is connected to the router via Ethernet, the maximum separation between the router and the bridge is limited. The router due to its large duty cycles and close proximity is often a source of interference, while the clients connected to it have a considerably smaller duty cycle. As the router could be transmitting on the wireless interface when the bridge senses the channel during its CSMA/CA mechanism, the Zigbee network could suffer from radio interference that could lead to high latency or even transmission failures. High bandwidth applications like video streaming and large file transfers saturate the link further and cause the Zigbee network to backoff multiple times and even end in failure.

In the case of a Smart lighting solution that is connected to the Wi-Fi router, this information presents us with a unique opportunity to employ the router as a centralized controller for the two heterogeneous networks. Since the Wi-Fi router acts as a gateway for both networks, they constitute essentially a part of the same IP subnet as shown in Figure 5.1. In this system design, we propose to use the Wi-Fi router as a coordinator node between the two networks. We do not require any hardware/MAC level changes or connect a new node that keeps the system design cost-effective. The router can then be used to provide different QoS to both Zigbee and Wi-Fi networks. This modification will support the Zigbee network to mitigate interference and increase its reliability.

In our system design, we focus on the following important points:

1. Use of a preventive approach rather than a reactive approach. As we saw in Section 3.2, many previous solutions follow a reactive approach i.e. they implement changes to their system when they detect interference. However, in our system design, we propose to adopt a preventive approach, that makes changes just before the possibility of interference. As the packets destined for both Wi-Fi nodes and the Philips hue bridge are forwarded through the router, this observation can be used by the router to prevent interference. The router after detecting the packet destined for the Philips Hue bridge can execute a control algorithm/method that makes the environment favorable for Zigbee transmissions. This approach is critically essential for a system like a wireless lighting system as the frequency of control messages is low but requires superior reliability and low latency.

2. Multiple studies confirmed that Zigbee is uniquely vulnerable to high Wi-Fi traffic [52]. Taking advantage of this observation, we propose employing the router to alter the traffic pattern of Wi-Fi to imitate a low interference environment during Zigbee transmission. In this way, we can provide reliable Zigbee transmission meanwhile keeping Wi-Fi transmissions ongoing albeit at a low data rate. The transient low interference environment decreases the chance of overlap and increases the opportunity to receive the Zigbee packet in a timely manner. This methodology is an improvement and provides a sort of pseudo-fairness to Wi-Fi as compared to other solutions that completely cut-off Wi-Fi transmission and cause unfairness to them. We propose running the control algorithm on the router only after the detection of a Zigbee packet and store the excess Wi-Fi packets at the buffer of the router to prevent potential data loss. By following this process, we make certain that transmission of Zigbee remains reliable and inflict none to minimum damage on Wi-Fi transmissions.

As another advantage, as we control the packet transmission in the time domain, the Zigbee network can operate on an overlapping channel with the home Wi-Fi network. Making an assumption that the nearby Wi-Fi networks would be operating at a separate channel that the home Wi-Fi, this choice will reduce interference from external Wi-Fi networks. In addition, this will lead to a more efficient utilization of the frequency spectrum.

In this system design, to provide Zigbee reliability, we propose to employ the router to carefully limit its data rate during Zigbee transmission. By employing the router and not introducing a new node we keep the system complexity low. Furthermore, in this design, we provide fairness to the Wi-Fi network by not completely stopping its transmission. Many experimental studies test Zigbee performance under different Wi-Fi traffic profiles like web surfing, file downloads, media streaming, etc. The General Consensus is that Zigbee performs poorly under high interference, however, they do not report a threshold that can be used as a benchmark. In this system design, rather than completely stopping Wi-Fi transmission, we focus on carefully limiting the flow of Wi-Fi packets to a threshold data rate that is manageable for reliable Zigbee transmissions with tolerable latency. If we lower the transmission of Wi-Fi, it will give Zigbee more opportunities to successfully acquire the channel and transmit its packet. To ensure minimum loss for Wi-Fi, we propose to only limit its rate when there is a Zigbee transmission. At other times the router is allowed to transmit packets at the standard rate.

To design such a system, we first need to understand which mechanisms are employed by the router to detect and transmit packets. In the following section, we will introduce the firewall application of a router that can be utilized to alter, drop, or detect packets based on predefined rules. Afterward, we will review some of the queuing disciplines that can aid us in designing our system by altering how packets are transmitted from the egress interface.

5.3 Detecting and Filtering Packets

Almost all routers run on a version of the Linux Operating system. The Linux kernel consists of a packet filtering framework called netfilter. It can be used to permit, drop, or modify traffic coming in and out of the interface. Iptables is a command-line interface that builds upon netfilter is used to configure the firewall by adding/modifying rules. They work by linking with the packet filtering hooks of the Linux kernel's networking stack. Each packet that enters the system triggers these hooks which permits programs registered with them to interact with the packet at key moments. The iptables kernel programs ensure the traffic flow conforms to the enforced conditions of the firewall[25]. The kernel modules can register with a total of five hooks. These modules are triggered as the packet progresses through the networking stack[44]. The hooks represent multiple well-defined points in the stack and are mentioned below:

1. NF_IP_PRE_ROUTING: This hook gets triggered as soon as a new packet enters the networking stack. Programs can implement this hook to process this packet even before any routing decision has been made about its destination.

2. NF_IP_LOCAL_IN: This hook gets triggered when the packet has been routed and is destined to the local system i.e. the router in this case.

3. NF_IP_FORWARD: This hook gets triggered when the packet has been routed and is destined for another host.

4. NF_IP_LOCAL_OUT: This hook is triggered when the traffic is generated locally i.e by the router and needs to go out of the system.

5. NF_IP_POST_ROUTING: This hook gets triggered by any forwarded or outgoing traffic just before it's forwarded to the NIC.

Tables are used to organize rules within the firewall. Inside each iptables table, rules are further organized within separate chains. They can include built-in chains and also contain user-defined chains. The chains represent the netfilter hooks which trigger them and determine when rules will be evaluated. The general tables and their respective chains are described below:

1. Filter: It is used to make decisions whether to allow the packet to traverse to its destination or deny it. Its built-in chains are - Input (Packets going to local sockets), Output (locally-generated packets), and Forward (routed packets).

2. NAT: As the packet enters, rules in the NAT table determine whether or how to modify the source or destination address of the packet to impact the way it is routed. Its built-in chains are - Prerouting: designating packets when they come in (for designating incoming packets), Output (locally-generated packets before routing), and Postrouting (to alter packets on the way out).

3. Mangle: It is used to alter the headers of the IP packets. For example, it can adjust the TTL value hence altering the number of hops it can sustain. can be used to change the ToS (Type of Service) field to alter its priority. Its chains include - Prerouting (incoming packets), Postrouting (outgoing packets), Output (locally-generated packets).

4. Raw: This table is used to track connections as soon as the packet enters the network interface. The built-in chains are - Prerouting (packets arriving at the network interface) and Output (locally-generated packets).

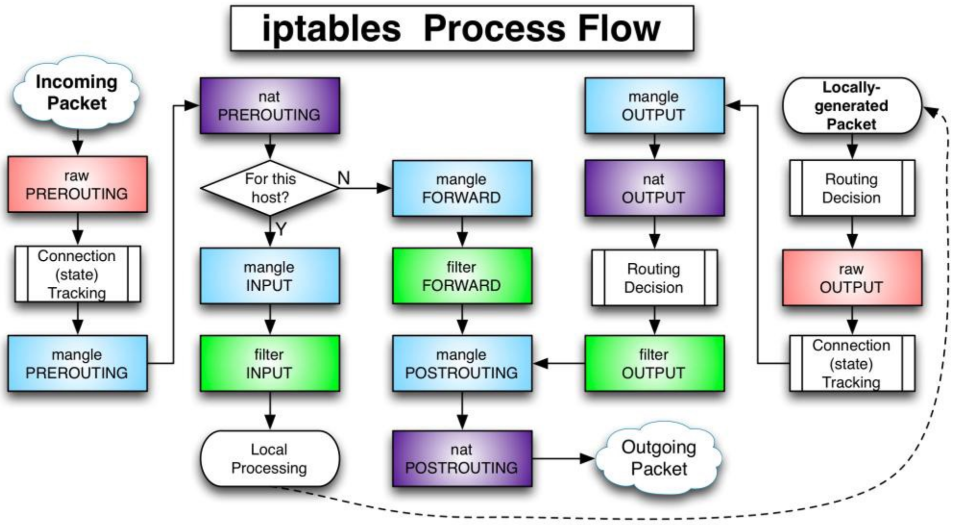


Figure 5.2: **iptables Process Flow.** [41]

The general flow of packets is illustrated in Figure 5.2. The packet after arriving at the ingress port passes through numerous tables where certain rules can be provided to decide the fate of the packet. The packet is checked against each rule within the chain. Each rule supports a matching component and an action component. The matching component specifies the criteria which the packet must satisfy for the corresponding action to be executed. Matching can be performed on the basis of source/destination address, protocol type, source/destination port, etc. The action component could include dropping or allowing a packet, altering its components, logging it in the system, etc. Logging is extremely helpful during troubleshooting to diagnose a problem. An action component called LOG can be used to log all packets which fulfill the matching component of a rule.

As we want to trace packets destined for the Zigbee coordinator as soon as possible we can employ the logging facility of iptables. We can append a rule to the raw table and the prerouting chain to identify the packet as soon as it enters the ingress port i.e. the Prerouting chain in the raw table. The matched packets get appended to the kernel logs which can be used for further processing.

5.4 Overview of queuing disciplines

Queuing disciplines are used to manage how incoming packets are organized at the egress port of an interface. They can be employed to alter the way data is sent by rescheduling, delaying, or even dropping it.

A qdisc is assigned to every network interface. Whenever the kernel needs to forward a packet to an interface, it is enqueued to the qdisc configured for that interface where it is processed according to the rules specified by the qdisc. Afterward, it is forwarded to the network adaptor driver for transmission. The algorithm for processing packets is internal to each network interface and appears transparent to the other devices. qdiscs have two categories, namely classless and classful. Classless qdiscs as the name suggests does not maintain classes. They accept the packets and only reschedules delays or drops the packets. They can have multiple queues but data can't be manipulated to a specific queue. The classful qdiscs can contain classes that are used to divide packets into unique flows by the use of filters to handle them with different priorities. Separate classes can be configured with different QoS to affect their throughput and latency[6].

5.4.1 Pfifo qdisc

The simplest qdisc is the pfifo which is purely First In and First Out. In Pfifo, packets are transmitted according to their arrival. This qdisc contains three queues which are also called bands, numbered from 0-2 with queue 0 receiving the highest priority. It does not allow for any modifications to the qdisc, the bands, or the way packets are queued in them.

The kernel takes care to insert packets with minimum delay requirement in band 0. As long as queue 0 is not empty, queue 1 won't be processed. A similar process is followed for queue 1 and queue 2. This way pfifo provides different QoS to separate packets based on the ToS flag present in the IP header [6, 32]. Naturally, fairness is an issue as high priority flows might starve low priority flows.

5.4.2 Stochastic Fair Queuing

This qdisc divides the total traffic in n different flows. It then tries to provide fairness to all flows by using a hash function to distribute the traffic in multiple FIFOs queues which are dequeued in a round-robin fashion. To maintain fairness among flows, the hash function is changed periodically by using a parameter called perturb[32].

5.4.3 Token Bucket Filter qdisc

TBF is a classless qdisc specially designed for traffic control. It can be used to control the rate of traffic flow. It allows the passage of packets arriving at

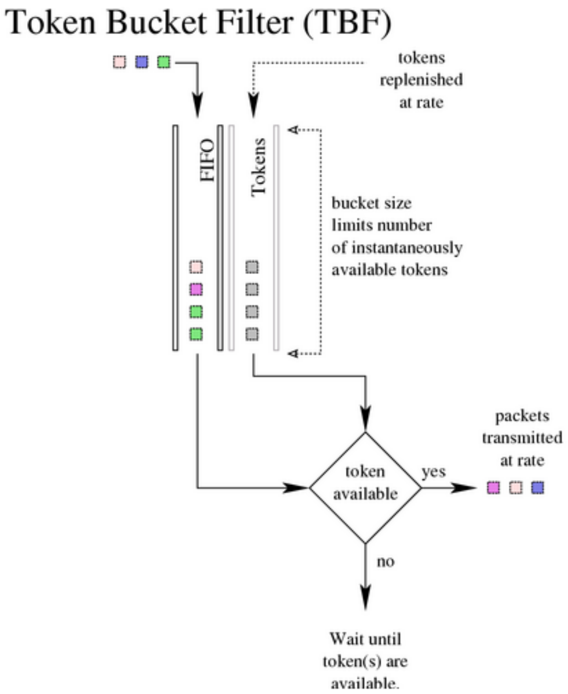


Figure 5.3: **Working of Token Bucket Filter (TBF) qdisc [32]**

a rate that is not exceeding a preconfigured set rate. It is network/processor friendly and is extremely precise. As shown in Figure 5.3, this qdisc is built on the classical concept of tokens and buckets.

The bucket is continually filled with a few virtual pieces of data called tokens, at a particular token rate. Each token compares to a byte of data. The bucket measure is extremely important for this qdisc as it is the number of tokens it can retain at any instance. Every token associates itself with one approaching packet from the queue and is subsequently removed from the bucket. TBF can be tuned to the requirement of a user by manipulating various knobs like the rate of arrival of tokens, buffer size, and latency which defines the maximum time a packet can sit in the queue. As the incoming rate of data packets and the token rate could be different, this can produce three scenarios:

1. The rate of data arrival in TBF is break even with the rate of approaching tokens. In this case, each approaching bundle receives its coordinating token and it endures no delay.

2. The rate of data arrival in TBF is smaller than the token rate. In this case, only a fraction of tokens are used and the remaining tokens are accumulated up to the bucket limit. These tokens can at that point be utilized to transmit information at a rate that surpasses the token rate which happens in a short data burst.

3. The rate of data arrival in TBF is greater than the token rate. In this

case, the TBF sends out data at the required rate until the tokens get depleted. Packets over this limit are queued as they wait for tokens to arrive, this situation is called an overlimit situation. [5]

The last scenario is significant, as it allows us to limit the outgoing traffic on an interface. In the case of overlimits, the waiting packets can be stored in a fixed-size buffer. More significant shaping rates typically require a larger buffer. If a packet arrives and the buffer is full then it is dropped. In this way, TBF provides accurate control over the bandwidth assigned to an interface.

5.4.4 Hierarchy Token Bucket qdisc

Similar to TBF, HTB uses the concept of tokens and buckets. It is a classful qdisc that permits complex and granular control over data flow by the use of several knobs. Filters are responsible for classifying traffic, which enters the qdisc, into its child classes. The responsibility to classify data into the correct child class lies with the filters. The packet is placed into one of the child classes after the decision from the filter. Shaping only occurs in leaf classes, while the responsibility of distributing the available tokens is taken up by the inner or root classes. It is much more sophisticated than TBF qdisc as it allows for more complex traffic shaping based on a borrowing model as shown in Figure 5.4.

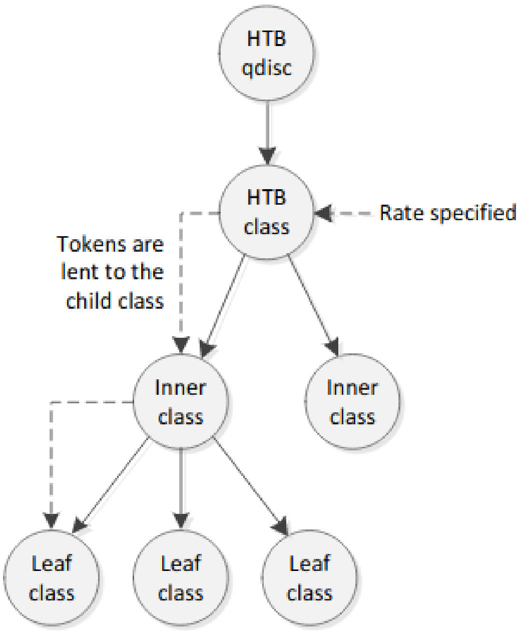


Figure 5.4: **HTB qdisc hierarchy with rate control.** [4]

It permits the creation of classes in a hierarchical fashion. The bandwidth is guaranteed by using a TBF in each class. Every child class maintains a pre-configured rate and a maximum rate (ceil rate) which it can achieve by borrowing tokens from their parent class if available and required. After reaching the maximum ceil rate, all incoming packets start getting buffered as they wait for tokens. The classes can in addition be offered a priority to ensure the packet from the higher priority class is dequeued first [4].

5.5 Final system design

In the previous sections, we highlighted the importance of a centralized system in the coexistence of Zigbee and Wi-Fi in a home environment. Unfair MAC parameters for Zigbee carry out a critical role in making it vulnerable to Wi-Fi interference. We proposed an idea to use the Wi-Fi router to effectively manage the existing infrastructure and synchronize transmissions between the two technologies. We reviewed how we can detect packets instantaneously and use it to trigger a custom algorithm. We also observed how we can alter the rate of transmission at an interface of the router using custom qdisc. In this system, we focus on limiting the flow of Wi-Fi packets to a manageable rate for successful Zigbee transmissions with tolerable latency. To provide priority to Zigbee traffic and cause minimum performance degradation to Wi-Fi, we lower the Wi-Fi bandwidth only during Zigbee transmission. The low bitrate gives Zigbee a more realistic chance to successfully acquire the channel and transmit its packet. During Zigbee transmission, as the router is still transmitting but at a low bitrate, the throughput of the Wi-Fi network is not zero as in the case of previous solutions.

Figure 5.5 illustrates the control of the flow of packets for the Wi-Fi router in our system design. As mentioned in Section 5.2, we adopt a preventive approach rather than a reactive approach. In the first step, the router keeps searching for a packet destined for the Zigbee bridge. If the router detects this packet based on the destination IP on the packet, it lowers the data rate of the wireless interface. This is performed by adjusting the rate of the qdisc to rate R for time T. If the router does not detect a packet for the Zigbee bridge, it keeps transmitting Wi-Fi packets at the standard rate. During the time of rate-limiting, if the router receives packets at a more rapid rate than the configured rate R, the excess packets would be stored in the buffer of the queue. After Time T, we increase the rate of the wireless interface to a high rate which allows all the buffered packets to be transmitted in the next instant. The most relevant parameters for this system are the allowed bandwidth for Wi-Fi during Zigbee transmission and the time for this application. If the time is extremely small, the router will prematurely start transmitting with a high bitrate which will cause more interference for the currently transmitting Zigbee node. If the time is extremely large, the Wi-Fi network will suffer from unnecessary packet losses and increased latency. By optimizing the values of the rate R and Time T, we can minimize data loss for Wi-Fi meanwhile ensuring reliable Zigbee performance.

HTB and TBF qdiscs are both ideal for limiting the data rate at an interface. TBF is simple as it uses a single queue and provides no prioritization, treating every packet equally. This behavior can be harmful to some applications like VoIP, online gaming as they require strict QoS parameters like low latency and zero loss. TBF can block high priority traffic due to the presence of lower priority traffic in front of it. TBF will also start dropping high priority packets if the buffer gets full. HTB is more complex in its operation as it can be used to provide priority to unique flows. It can be employed to make a separate

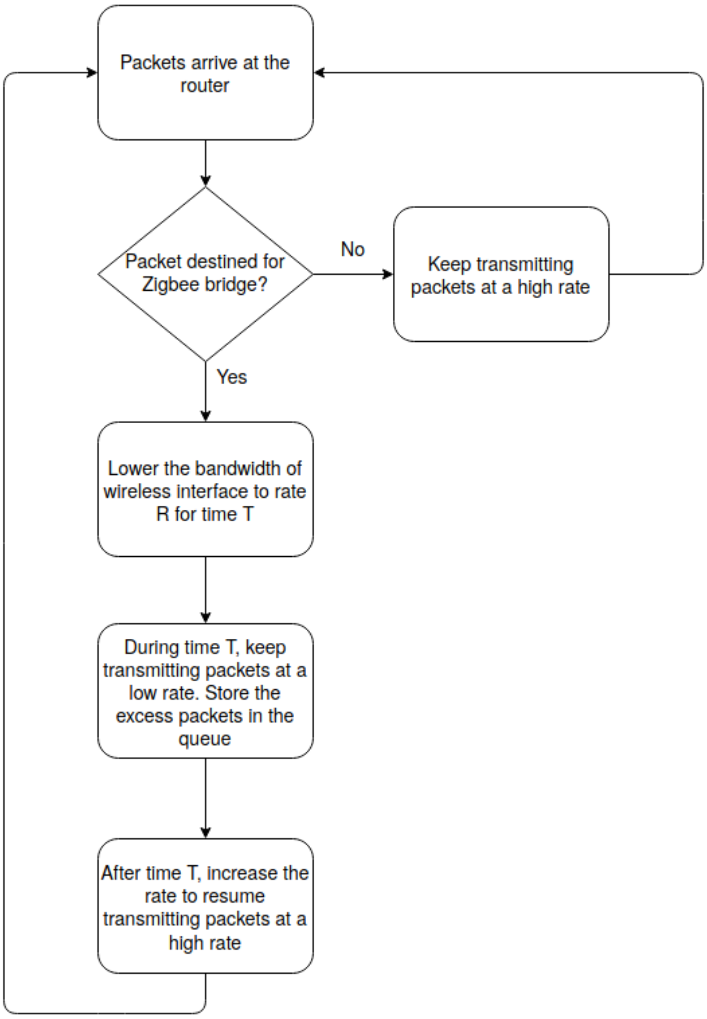

Packets arrive at the router
Packet destined for Zigbee bridge?
No
Keep transmitting packets at a high rate
Yes
Lower the bandwidth of wireless interface to rate R for time T
During time T, keep transmitting packets at a low rate. Store the excess packets in the queue
After time T, increase the rate to resume transmitting packets at a high rate

Figure 5.5: **System design**

class (queue) for high priority traffic which ensures an independent buffer space for them. These two qdiscs can have their application in different scenarios, however, their operation will cause the same effect on Zigbee transmission if the allowed bandwidth (R) is the same for both of them.

Table 5.1 illustrates the difference in approaches of some of the major methods introduced in Section 3.2 and our system design. As we can see, these methods do provide reliability for Zigbee transmissions but at the cost of added complexity and Wi-Fi loss. The COFFEE approach provides reliability to Zigbee transmissions but its complexity is high due to Wi-Fi subcarriers nulling corresponding to the frequency of the Zigbee network. However, it still takes into account of Wi-Fi loss as it doesn't completely cut off Wi-Fi transmissions, unlike other solutions. Buzzbuzz makes multiple MAC/PHY level changes which increase its complexity. It provides high reliability for Zigbee transmissions but doesnt consider its adverse effect on the Wi-Fi network. Cooperative Busy tone introduces a new Zigbee node to transmit a busy tone during Zigbee transmissions. This method adds to the cost and introduces PHY level changes to the system. This method also does not provide consideration for the Wi-Fi network and can lead to high Wi-Fi data loss.

Our system design is more feasible as compared to these techniques as it provides central control to both networks with the help of a gateway node. It helps in reducing collisions and providing ease of channel access for Zigbee transmissions meanwhile keeping Wi-Fi loss at a minimum. It does not require any changes to the standard or node which keeps its complexity and cost low. However, it does requires the Zigbee bridge to be compatible with the Wi-Fi router. We also need to make changes to the software of the router which enables it to detect Zigbee packets and trigger mechanisms to help Zigbee transmissions.

Features	COFFEE	BuzzBuzz	CBT	Our System Design
PHY Layer modification	Yes	Yes	Yes	No
MAC Layer modification	No	Yes	No	No
Complexity	High	High	High	Medium
Cost	Low	Low	High	Low
Reliability	Medium	High	High	High
Centralized	No	No	No	Yes
Requires Wi-Fi router compatability	No	No	No	Yes
Wi-Fi Loss	Low	Medium	High	Low

Table 5.1: **Comparison between different interference mitigation techniques**

In the upcoming chapters, we will present the implementation of our system. We will find out the optimal rate and time for our system design and perform its performance analysis.

Chapter 6

Implementation

In this chapter, we provide a description of the specific hardware and software tools used in this work. Tools for both Zigbee and Wi-Fi networks are described separately.

6.1 Zigbee Network

Two EFR32MG12 Mighty gecko wireless SoC from Silicon Labs were used in this work for establishing the Zigbee network. The EFR32MG family includes wireless networking stacks for Zigbee, Thread, and Bluetooth Low Energy and is ideal for designing energy-friendly, wirelessly connected devices. These devices include support for proprietary wireless protocol development and were employed to establish the Zigbee network. The SoC module consists of the actual radio board and an additional Web Services Took Kit (WSTK) board which is used for flashing the firmware, debugging, resetting, and various other purposes [47]. Table 6.1 indicates some of the specifications of these devices. These devices provide high energy efficiency, a scalable power amplifier, and fast wake-up times.

An IDE provided by Silicon Labs called Simplicity Studio was used to configure these SoC. It includes a powerful suite of tools for configuration, network analysis, energy profiling, and software examples. It contains an app builder that provides multiple features under different tabs. Some of the features used in this work include:

- Network Analyzer - This feature allows for the use of a packet tracer for analysis of the wireless connection between different SoC.
- RTOS - Micrium OS, a real-time operating system that can be used to run tasks with priorities, triggering task execution, etc.
- Serial interface - feature to send/receive commands using serial USB.
- General - displays the hardware architecture and the application configuration for different boards.
- Plugins - On the basis of application requirements, multiple manufacturer implementations can be optionally included using this feature.

Table 6.1: **SiLabs EFR32MG12 Hardware Specifications**

Processor	32-bit ARM Cortex M4
Programmable Flash Memory	1 MB
RAM	256 KB
Operating Voltage	1.8 to 3.8V
Clock Speed	40 MHz
Compatibility	BLE, Zigbee, Thread protocols

6.2 Wi-Fi Network

A router is a Layer 3 device that performs decisions on how to transmit the packets based on the contents of the IP packet. In our system design, we propose to employ the router as a central device for both Zigbee and Wi-Fi networks. In order to do so, we need to have total control over the router and its decision-making rules. Most firmware designed by manufacturers of home routers allows for little to no customization. They operate a version of the Linux operating system, however, they don't allow access for making changes or optimizing the firmware for a particular application. To maintain absolute control over the functionality of the router, it was decided to use OpenWrt. It is an open-source Linux based operating system for embedded devices, primarily oriented for networking support [39]. OpenWrt provides much more freedom in terms of installing utilities and packages which can be used to get much more network information out from the router then typically exposed. They additionally allow us to use utility programs like iptables to configure the IP packet filter rules of the Linux kernel firewall. These rules come in handy if the user wants to drop or block traffic from a particular host or a service. Packages like tc [6], can also be installed which allows for the use of different queuing disciplines for diverse applications. In this work, we have used a common home router with the OpenWrt operating system. We have added custom iptables rules that allow for the detection of packets destined for the Zigbee node. These rules are then used with the tc functionality to configure a rate-limiting queueing discipline in this work. Specifications of the EA6350 Linksys router [30] is indicated in Table 6.2 which was flashed with the appropriate OpenWrt firmware. One or more laptops were used in the wireless node to complete the Wi-Fi network.

Table 6.2: **LINKSYS EA6350 Hardware Specifications**

Feature	Specification
Architecture	ARM
System-On-Chip	Qualcomm IPQ4018
CPU/Speed	ARM Cortex A7 @ 717MHz 4 cores
Flash size	Winbond 128MB SLC NAND Flash
RAM	Samsung DDR3 256MB RAM
Wireless	IPQ4018 2.4GHz / 5GHz 802.11an+ac
Ethernet	1000 Mbit/s w/ vlan support

Chapter 7

Performance Analysis

In this chapter, we evaluate our system design in terms of reliability and latency of the Zigbee network. The Zigbee performance is assessed by various unicast packet transmissions. We also present the effect of our system design on the Wi-Fi network.

7.1 Zigbee Unicast Transmission

Whenever a Zigbee node wants to transmit data to another node, it generates data at the top-most APL layer. From there the data packet travels down to the NWK layer, the MAC layer, and finally reaches the PHY layer as shown in Figure 7.1. Packets can be queued when a node tries to transmit multiple packets.

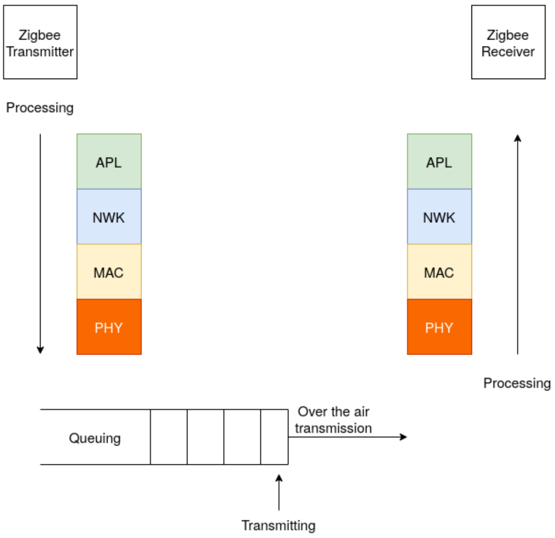


Figure 7.1: **Packet transmission [31]**

The packets are transmitted one by one until the queue is empty. The receiving node processes the packet from the bottom layer to the top layer. Also, the node uses the CSMA/CA mechanism as discussed in Section 2.2 to prevent collisions from other interferes.

Before transmitting the packet, the Zigbee transmitter node backs off for a random amount of time and senses if the shared channel is idle. If the channel is busy, it backs off and tries once more after some time. After sensing the channel to be idle, it starts transmission of the packet and receives an acknowledgment from the receiving node after successful transmission.

RTT (Round Trip Time) for one Zigbee packet is shown in Figure 7.2. It is calculated from when a packet is generated at APL of the sender node, to the time APL receives a response from the target node.

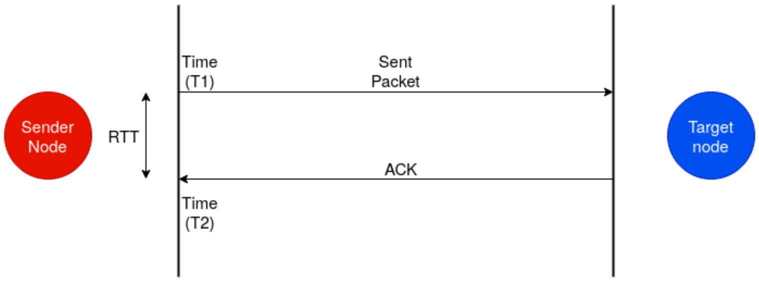


Figure 7.2: **Timing experiment setup.**

It includes processing time for headers at different layers, time spent in sensing the channel, transmission time (250 Kbps), and the actual propagation time over the air. In the case of collisions, re-transmissions also contribute to the whole RTT. They are dealt with in all three layers -APS, MAC, and NWK layer.

The packet is forwarded from the APS layer to the NWK layer which starts a timer for 250 ms and sends the packet to the MAC layer. The MAC layer starts the CSMA/CA process to sense the channel. After successfully accessing the channel, the MAC layer transmits the packet and waits up to 864 μs (54 symbol period) for an acknowledgment. If the ACK is received within this time, the MAC layer reports success to the NWK layer and the transmission is successful. However, if the ACK is not received, the MAC layer retries the entire CSMA/CA transmit process up to three more times. In the total four attempts, if the ACK is not received, the MAC layer reports failure to the NWK layer. NWK layer waits for 16 ms (the default for EmberZnetpro stack) and asks the MAC layer to transmit again. NWK layer then attempts until either it receives a success or the timer reaches 250 ms. If the timer reaches 250 ms, it reports failure to the APS layer, and the packet is lost [53].

7.2 Test setup

An experimental set-up was designed to determine packet loss and latency using RTT during data transmission between two Zigbee devices. Two Mighty Gecko wireless SoCs from Silabs described in Section 6.1 were used.

They were placed at a distance of 10m from each other. This distance was assumed to be the maximum distance between the Hue bridge and the first-hop neighbor[26]. In the case of multihop mesh networks, it is important that the first node successfully receives the message as it can then be relayed further to other distant nodes. A single Zigbee packet was transmitted from one device to another. After receiving the packet the node replies back to the source node. The test was performed 100 times i.e. 100 Zigbee Unicast messages were sent between the two Zigbee devices to observe variation between multiple attempts. The different measurements were independent. During any one of the test cases, if there was a re-transmission of a Zigbee packet, it did not have an influence on the transmission of the next packet. The software was developed using Simplicity Studio and deployed on the two SoC.

Throughput library plugin along with Network Analyzer from Simplicity Studio was used to perform the experiment. The command shown in Listing 7.1 was used to send out packets from one node to another.

Listing 7.1: **Zigbee Packet Configuration**

```
plugin throughput set-all 0x0002 1 0 100 1 0 100000
```

Here 0x0002 represents the destination node_id, 1 is the number of packets, and the interval field was set to 0 since we were only sending a single packet. The packet size was kept at 100 Bytes, 1 represents the packets in flight, APS security was off, and the timeout at the default value of 100000 ms.

After successfully receiving the acknowledgment the source node reports the RTT and different diagnostic counters like CCA fails and retries. If the transmission is unsuccessful, the node reports a packet loss. Packet loss was used to indicate the reliability of Zigbee transmission. Higher packet loss meant lower reliability of the system.

7.3 Analysis of Zigbee packet transmission

In this section, we will analyze the transmission of Zigbee packets under three scenarios:

1. Under no Wi-Fi interference
2. Under Wi-Fi interference using different loads
3. Under Wi-Fi interference using different loads utilizing our system design

7.3.1 Zigbee performance under no Wi-Fi interference

To benchmark the performance of Zigbee transmission, it was decided to conduct the experiment under no external radio interference. In this scenario, 100 Zigbee packets were transmitted one by one from one Zigbee node to another under no Wi-Fi interference. This test was conducted in a basement with thick walls to ensure the environment was interference-free from Wi-Fi and other networks. Figure 7.3 illustrates the distribution of the RTT for different observations. No packet losses were observed which could be credited to no external interference. We can see that the average RTT remains at 5 ms while we in some attempts we have an RTT of 6 ms and 7 ms. This high value can be due to either one or both nodes choosing a high number from the [0, $\mathrm{CW_{min}}$] distribution during their backoff process. In the following section, we will see how Wi-Fi affects the latency and throughput of our Zigbee network.

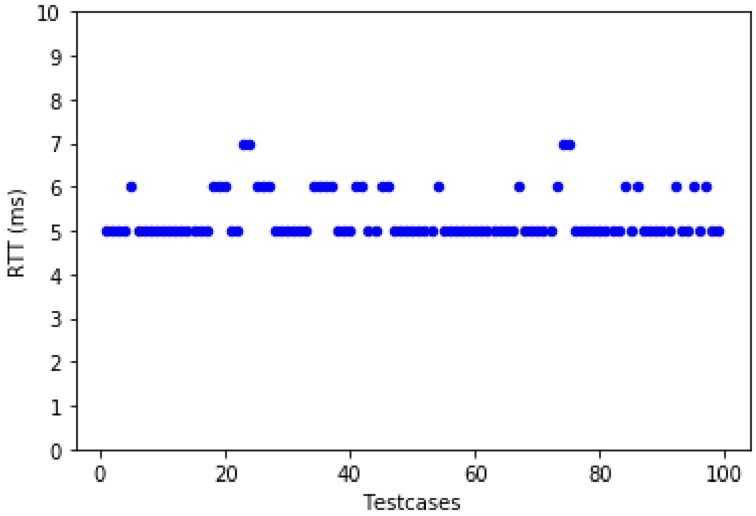


Figure 7.3: **RTT of single Zigbee packet under no Wi-Fi interference**

7.3.2 Zigbee performance under varying Wi-Fi interference

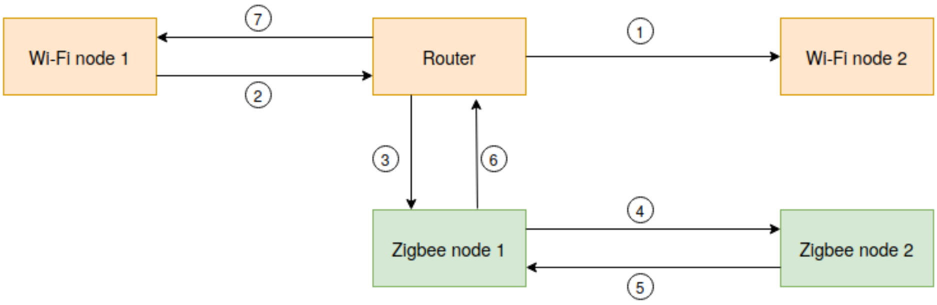


Figure 7.4: **Message Sequence**

In this section, we will try quantifying the effect of Wi-Fi interference on Zigbee transmission under different loads. To understand how Wi-Fi interference affects the latency and reliability of a Zigbee network, we performed the same experiment under increasing Wi-Fi loads. The Zigbee network was operating on an overlapping channel (Zigbee channel 25) with Wi-Fi operating in 802.11n mode (Wi-Fi channel 11) to ensure maximum interference. The message sequence is shown in Figure 7.4.

Just like the Philips Hue bridge, Zigbee node 1 was connected to the router via an ethernet connection and placed close to it. To send a control message and receive a diagnostic message from this Zigbee node wirelessly, a PuTTY[1] application was used to connect to its virtual COM port using TCP/IP socket

[1] a free and open-source terminal emulator, serial console, and network file transfer application.

4901[49]. An Iperf3[2] application was installed on the router which transmits UDP traffic at a constant rate to a Wi-Fi node to create Wi-Fi interference. Distinct levels of Wi-Fi interference can be produced by adjusting the bitrate of the Iperf3 application. UDP was chosen instead of TCP protocol as it has no congestion control mechanism. Since UDP does not require ACK from the receiver it has no way to know about packet loss. Even in the case of packet loss, the sender will continue sending packets at the same rate while TCP in this case will ask the sender to slow the rate in order to reduce congestion. As we want to maintain a constant rate of interference UDP was chosen.

The steps involved during each iteration are listed below:

1. The router starts transmitting packets to Wi-Fi node 2 to create Wi-Fi interference. The bitrate of UDP traffic was changed under different cases.

2. During this time, a control message is sent from Wi-Fi node 1 to Zigbee node 1. This message instructs Zigbee node 1 to transmit a unicast packet to Zigbee node 2 and report counters back to Wi-Fi node 1. The message first arrives at the router.

3. The router after checking the destination IP of the packet forwards it to Zigbee node 1.

4. After receiving the packet, Zigbee node 1 transmits a unicast message to Zigbee node 2 and waits for an ACK.

5. After successfully receiving the packet, Zigbee node 2 replies back with an ACK.

6. Zigbee node 1 calculates the RTT and reports other counters such as CCA fails and retries. The node then sends the message to the router.

7. The router forwards the counters to Wi-Fi node 1 where it is stored and further analyzed.

Since situations such as whether the Zigbee nodes are in a LOS of each other or not could affect their performance, it was decided to evaluate the setups under both scenarios and with increasing distance between the two Zigbee nodes. For both cases, the two nodes were placed in a straight line with varying distances from each other (3 m, 6 m, and 10 m). During the data collection from users in the Hue beta program as mentioned in Section 4.2, it was observed that some users placed their devices (both Hue bridge and Wi-Fi router) in a closed cabinet or a drawer. We choose this setting for the NLOS case, in which we placed the devices (Wi-Fi router and Zigbee node 1) in a cabinet, and the position of Zigbee node 2 was varied with distance while being in a straight line as shown in Figure 7.5. The dashed box around the router and Zigbee node 1 represents the cabinet in which the two nodes were placed for the NLOS scenario. For all cases, only the position of Zigbee node 2 was changed, whereas all other nodes were left untouched. For every distinct Wi-Fi bitrate, the experiment was performed for 100 test cases. Tables 7.1 and 7.2, illustrates the different RTT values measured during these test cases for LOS and NLOS scenarios respectively.

[2] a widely used tool for network performance measurement and tuning.

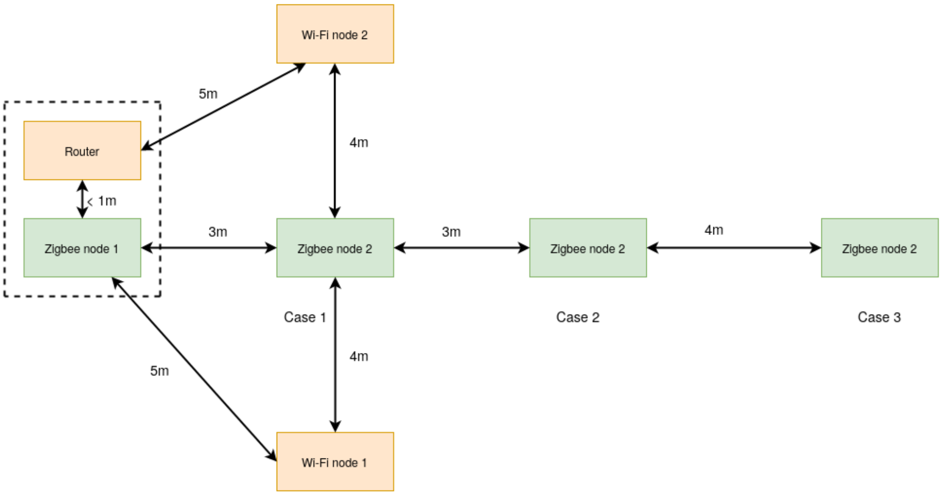


Figure 7.5: **Measurement Setup for both LOS and NLOS scenarios**

Bitrate	Average(3m)	Max(3m)	Average(6m)	Max(6m)	Average(10m)	Max(10m)
1Mbps	14ms	92ms	14ms	85ms	15ms	96ms
2Mbps	18ms	131ms	20ms	173ms	19ms	170ms
3Mbps	22ms	173ms	24ms	175ms	25ms	182ms
4Mbps	25ms	181ms	29ms	180ms	33ms	190ms
5Mbps	36ms	206ms	34ms	212ms	42ms	210ms
10Mbps	57ms	245ms	54ms	237ms	65ms	239ms
15Mbps	65ms	241ms	81ms	241ms	83ms	247ms
20Mbps	78ms	243ms	86ms	239ms	89ms	243ms
25Mbps	89ms	246ms	92ms	241ms	90ms	232ms
30Mbps	94ms	248ms	98ms	246ms	97ms	249ms

Table 7.1: **Effect of different Wi-Fi loads on RTT of Zigbee packet in LOS condition**

The observed minimum RTT for all cases was 5 ms and hence not reported in the tables. This value is also the same for the case of no Wi-Fi interference. This behavior could be attributed to the Zigbee node acquiring the channel before Wi-Fi which can happen when the Wi-Fi node randomly selects a high value during its backoff process as shown in Equation 3.1. However, we note an increase in Max and average RTT with increasing Wi-Fi bitrate for both cases. This happens due to an increase in the frequency of Wi-Fi transmissions for the same time period. Since the CCA time of Wi-Fi is multiple times lower than Zigbee, this gives Wi-Fi a priority which requires the Zigbee node to wait longer for its transmission. Naturally, these Zigbee packets also had a considerable number of CCA fails and retries which contributed to the high RTT. The observed RTT values for shorter distance are generally lower as compared to the larger distance, however, the difference is not very big.

Packet loss percentage is given by Equation 7.1.

Bitrate	Average(3m)	Max(3m)	Average(6m)	Max(6m)	Average(10m)	Max(10m)
1Mbps	15ms	92ms	14ms	90ms	17ms	100ms
2Mbps	22ms	167ms	27ms	178ms	28ms	182ms
3Mbps	25ms	189ms	30ms	171ms	32ms	188ms
4Mbps	33ms	233ms	36ms	214ms	39ms	246ms
5Mbps	42ms	240ms	47ms	243ms	50ms	247ms
10Mbps	60ms	245ms	59ms	245ms	68ms	249ms
15Mbps	79ms	249ms	87ms	241ms	88ms	243ms
20Mbps	85ms	244ms	90ms	239ms	95ms	249ms
25Mbps	101ms	243ms	115ms	241ms	117ms	246ms
30Mbps	104ms	248ms	112ms	246ms	109ms	245ms

Table 7.2: **Effect of different Wi-Fi loads on RTT of Zigbee packet in NLOS condition**

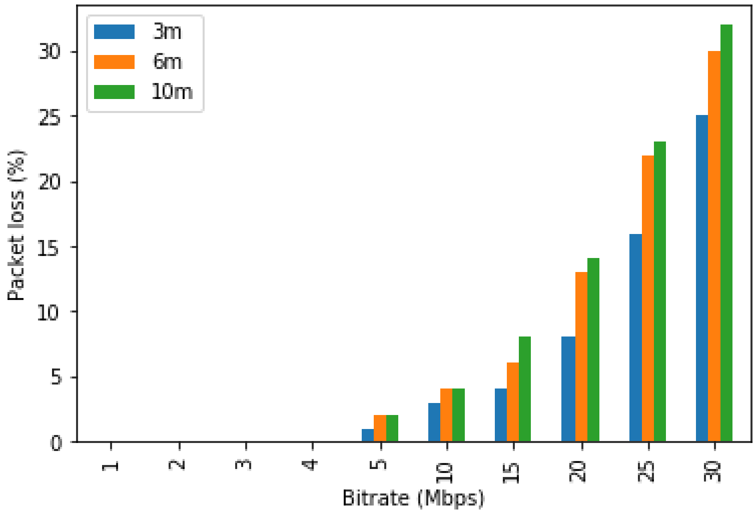


Figure 7.6: **Packet loss for Zigbee under LOS conditions**

$$Packetloss(\%) = \frac{Number\ of\ Packets\ lost}{Number\ of\ Packets\ sent} \times 100 \tag{7.1}$$

As we can observe from Figures 7.6 and 7.7, packet loss also increases with the increase in Wi-Fi interference. Even with CSMA/CA backoff procedures, we observe an increase in the number of lost packets with increasing Wi-Fi interference which points to a decrease in the reliability of the Zigbee network. In the case of LOS, the observed packet loss started from a Wi-Fi data rate of 5 Mbps for all three scenarios. However, in the case of NLOS, the packet loss started from a lower data rate of 3 Mbps indicating a high interference medium than in the case of a LOS setting. This could be attributed to more interference from signals that bounced off the surface of the cabinet.

We observe no packet loss up to 2 Mbps and almost a 50% loss at 30 Mbps for the NLOS case. For the LOS case, no packet loss was observed up to 4 Mbps

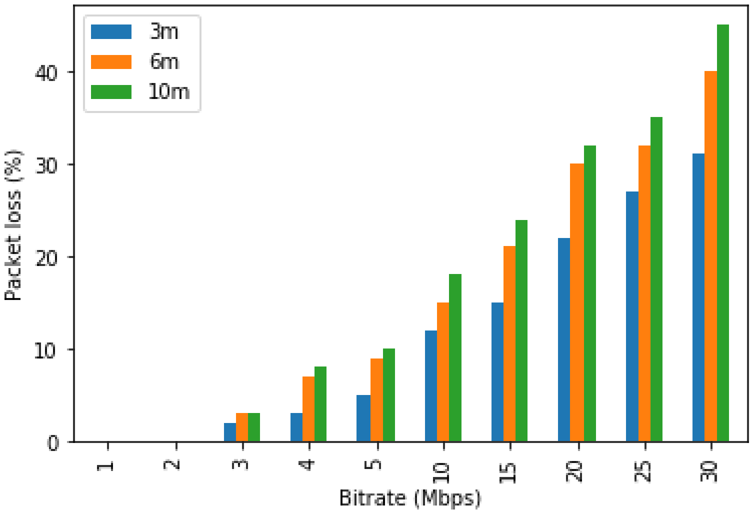


Figure 7.7: **Packet loss for Zigbee under NLOS comditions**

and almost a 35% loss at 30 Mbps.

According to this experiment, for our test setup, we observe no packet loss till 2 Mbps and 4 Mbps for the NLOS and LOS conditions respectively which is critically important for a reliable system. For these cases, the RTT is still manageable for these Wi-Fi thresholds as the average is under 35 ms and the maximum is under 200 ms. However, increasing the load degrades Zigbee's performance severely. For heavy loads, there were multiple instances in which the transmission was unsuccessful in the default limit of 250 ms.

To evaluate the effect of the interference on the second hop transmission of the Zigbee packet, a similar experiment was performed. In this scenario, the distance between the sender Zigbee node (Zigbee node 1) and the Wi-Fi router was varied which corresponds to the second hop for the Zigbee transmission. The results of the average RTT are shown in Figure 7.8. When the distance is very small i.e. 1m we observe the same results as we observed for the first hop transmission. However, as the physical distance increases between the Wi-Fi node and the Zigbee nodes, the effect decreases substantially and we record similar results for all considered distance even under heavy Wi-Fi load. This behavior confirms the first-hop transmission is of utmost importance (due to the close proximity of the Zigbee coordinator to the Wi-Fi router) and further transmissions are not affected by the Wi-Fi network. Similar results were presented in [13] in which the authors claimed a distance of 2m as a safe distance between the Wi-Fi network and the Zigbee network for reliable transmission.

These experiments confirms the Zigbee network is robust against the Wi-Fi network at small loads. However, its performance degrades under heavy load. To make our system reliable and decrease latency under heavy Wi-Fi loads, we choose a rate of 2 Mbps as an ideal choice for the NLOS case and a rate of 4 Mbps for the LOS case for our qdisc during Zigbee transmission. Since we record the maximum RTT of 190 ms for our experiment, we choose 200 ms as a reasonable choice for rate-limiting Wi-Fi transmission. Also, due to

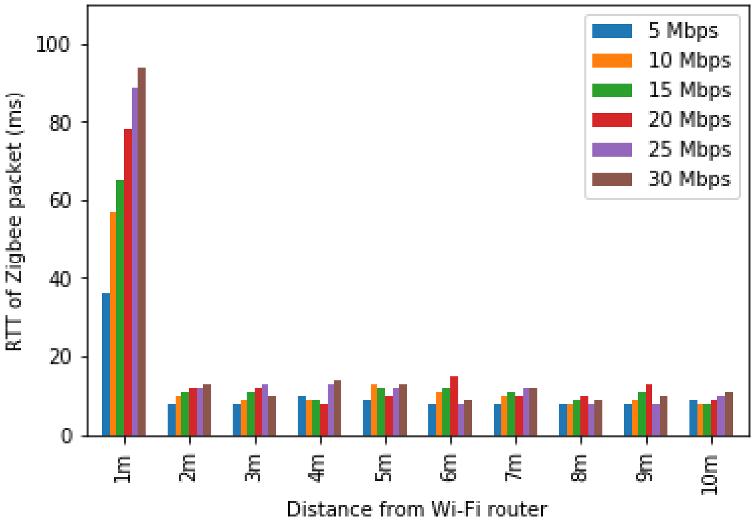


Figure 7.8: **RTT for second hop Zigbee packet**

the dynamic nature of the wireless medium and different settings, these values might change according to the medium. In that case, different thresholds could be employed to get the best performance. The combination of 2 Mbps and 200 ms can be considered as the safest option. For evaluating our system design, we have considered the case of the NLOS scenario in which the distance between the Zigbee nodes is 10 m since we have the highest observed Zigbee RTT and packet loss in this case. In other cases, the appropriate Wi-Fi threshold (R) and time (T) could be chosen to produce similar results. With higher thresholds, the Wi-Fi loss would be even less as more Wi-Fi packets would be allowed to be transmitted by the configured qdisc.

7.3.3 Analysis of the proposed system design using TBF qdisc

For the effective use of our system design, we require three parameters:

1. Safe Wi-Fi bitrate (R) for reliable Zigbee performance
2. The time of application (T) of the system design to minimize Wi-Fi data loss
3. A reliable and real-time method of detecting the arrival of the packet destined for the Zigbee node at the Wi-Fi router.

We experimentally found out the values of R and T for our system in the previous section. We still need a method to instantaneously detect the Zigbee packet at the router. This time is critical as it dictates the start of the rate-limiting at the router. We present our approach and results for the case of NLOS and when the distance of the Zigbee receiver node is 10m. The results for other cases are not presented as they were similar to the above case.

To achieve this functionality we make some additions to the firewall setting of the router. A firewall is a network security system that monitors and controls arriving packets based on a policy as introduced in Section 5.3. The policy consists of an ordered list of rules, executed from top to bottom and typically the first rule that matches the packet is performed [10]. Some of these policies can be used to drop packets from malicious sources such as a hacker attempting to send IP spoofed packets using raw packets or protect the network from TCP SYN flooding and DDOS attacks. Firewalls can also be configured to log such events for future analysis or for triggering a different event as introduced in Section 5.3.

To detect the arrival of the Zigbee packet we added a LOG rule to the kernel firewall of the router as shown in Listing 7.2.

Listing 7.2: **Firewall rule to detect Zigbee packet**

```
iptables −t raw –A PREROUTING −destination 192.168.1.177
    ↪ −j LOG −log−prefix ‘‘ZigbeePacket”
```

We make use of the iptbales utility to configure a rule that adds a LOG element for a packet destined for IP address 192.168.1.177 (IP of Zigbee node 1)[44]. The PREROUTING chain is the first chain that the packet encounters after entering the ingress port of the router. We trace the packet at the earliest possible time even before the routing decision has been made for the packet. As many firewalls don't have rule in place to record LAN-LAN traffic we also added a rule to enable the logging.

After the detection of the packet, the firewall makes an entry in the log file which could be read using the logread module. We run a script that continuously monitors the log file. After successful detection of a Zigbee packet, a script is run which configures the TBF on the wireless interface of the router to a low rate of 2 Mbps for a period of 200ms. This creates a transient low interference environment for Zigbee which it can take advantage of. By using this method we synchronize the Zigbee transmission with the application of rate-limiting at the router.

We first examine our system design using the TBF qdisc. The TBF was configured as below-

Listing 7.3: **TBF configuration**

```
tc qdisc change dev wlan0 root tbf rate 2mbit burst 8000
    ↪ latency 250ms
```

We use the tc (Traffic control) utility of the Linux operating system to configure the qdisc at an interface of the router [6]. This command attaches the TBF qdisc to the wireless interface "wlan0" and sets its data rate to 2 Mbps. The burst parameter represents the size of the bucket in bytes. Latency is the maximum amount of time a packet can be queued in the TBF.

To test our system design we use the same setup as described in Figure 7.4 with some changes. First, instead of using the router as an Iperf3 client, we use Wi-Fi node 1. This is done to imitate reality as close as possible. The Wi-Fi packets sent from Wi-Fi node 1 first arrive at the router which are then forwarded to Wi-Fi node 2 to create Wi-Fi interference. We add the packet detection functionality at the router which can change the data rate of the wireless interface whenever required. We start testing our system model by sending UDP traffic from Wi-Fi node 1 to the Wi-Fi node 2.

In between Wi-Fi transmissions, a control message to transmit a unicast message was sent from Wi-Fi node 1 to Zigbee node 1. After the successful detection of the packet at the router, the router rate limits its outgoing traffic to 2 Mbps for 200 ms. This time is used by Zigbee to complete its transmission under low Wi-Fi interference. After the transmission, we look at the latency of the Zigbee packet and for potential packet loss. We also analyze how our system design affects the performance of the Wi-Fi network.

Average RTT of Zigbee packets under different Wi-Fi interference is shown in Figure 7.9. For comparison, we also include the RTT under the normal operation of the router. We can see that the RTT under normal operation increases with increasing Wi-Fi interference. Under our system design the RTT remains almost constant and under 35 ms for every case. This result shows that the system design is effective even under high Wi-Fi interference. Even under 50 Mbps, our system design provides 3.5 times less RTT for the Zigbee packet.

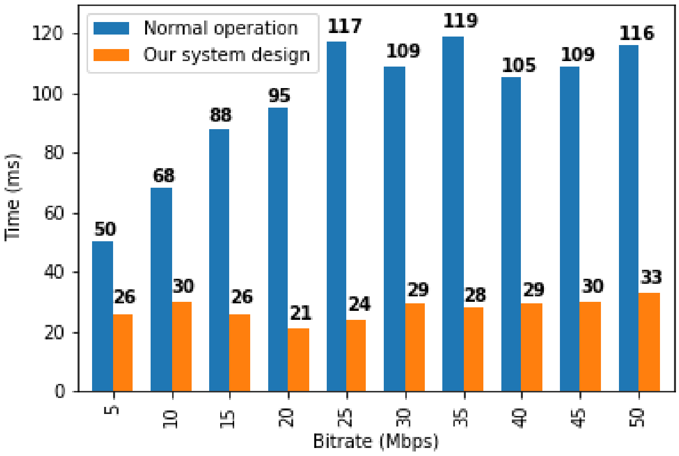


Figure 7.9: **Average RTT for Zigbee packet under normal operation and with our system design under increasing Wi-Fi interference**

Packet loss for both scenarios is shown in Figure 7.10. Under high Wi-Fi interference and normal operation of the router, we observe a packet loss of 67%, while our system design provides a 7% loss in the same setting. This result shows that our system design can provide reliability to the Zigbee network even under high Wi-Fi load.

Due to the limitations applied to the Wi-Fi router, it is allowed to send at a maximum data rate of 2 Mbps which translates to 0.4 Mb in 200 ms. This means for sender bitrates over 2 Mbps, the router will only be able to send -

$$Packets\ sent = \frac{Bitrate \times T_{\text{application}}}{8 \times Packetsize} \tag{7.2}$$

where bitrate is the bitrate of the sender, Packetsize was kept at 1400 bytes and $\text{T}_{\text{application}}$ was set at 0.2 sec.

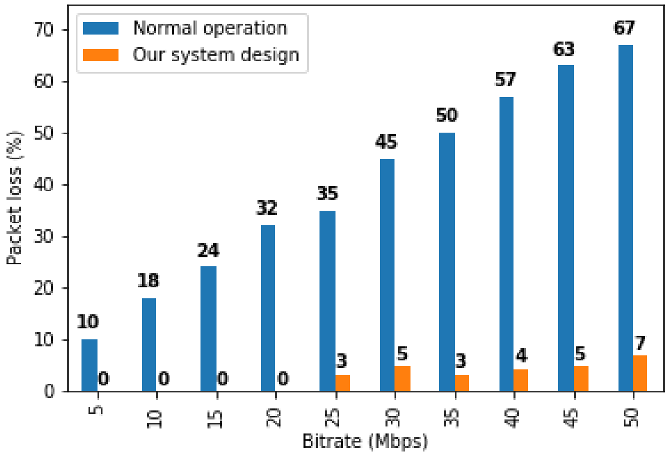


Figure 7.10: **Zigbee packet loss under normal operation and with our system design under increasing Wi-Fi interference**

According to Equation 7.2, for our system design, the router will only be able to transmit around 37 packets in the time period of 200 ms.

Data loss can occur at an interface when the amount of data arriving at the interface is more than the amount of data allowed to exit the interface. However, data loss can be reduced by using buffers which store the excess packets. Hence data loss occurs only when-

$$\mathrm{D_{in}} - \mathrm{D_{out}} > \text{Buffer Size} \tag{7.3}$$

where $\mathrm{D_{in}}$ represents the amount of data arriving at an interface, $\mathrm{D_{out}}$ represents the amount of data leaving the interface and Buffer Size is the size of the buffer of the interface.

If we consider the case of a bitrate of 40 Mbps, this translates to a total data arrival of 8 Mb per timeslot of 200 ms. The same can be seen in Figure 7.11.

This figure illustrates the number of packets transmitted by the router on the wireless interface in timeslots of 200 ms in the case of 40 Mbps bitrate. We see that during the first three timeslots the router is transmitting packets corresponding to 40 Mbps. In the fourth timeslot, the router rate-limits its egress interface to 2 Mbps making no changes to the ingress interface. During this time period, only 36 packets are allowed to leave the interface and the excess packets from Wi-Fi node 1 are stored in the buffer. In the next timeslot, the limitations are lifted and the router increases its bitrate to almost 80 Mbps to transmit the queued packets and the packets which arrive during this timeslot. As the buffer is empty now, in the subsequent timeslots the router keeps transmitting at a bitrate of 40 Mbps.

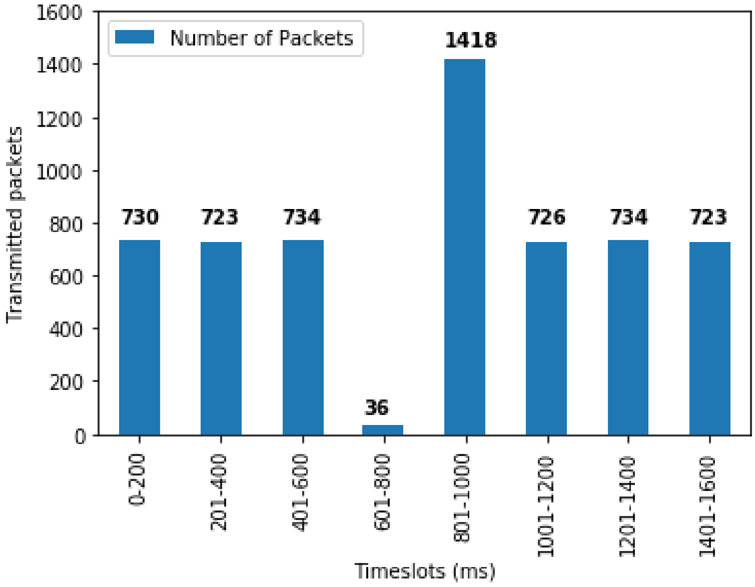


Figure 7.11: **Packet transmission from the router under our system design for the case of 40 Mbps bitrate**

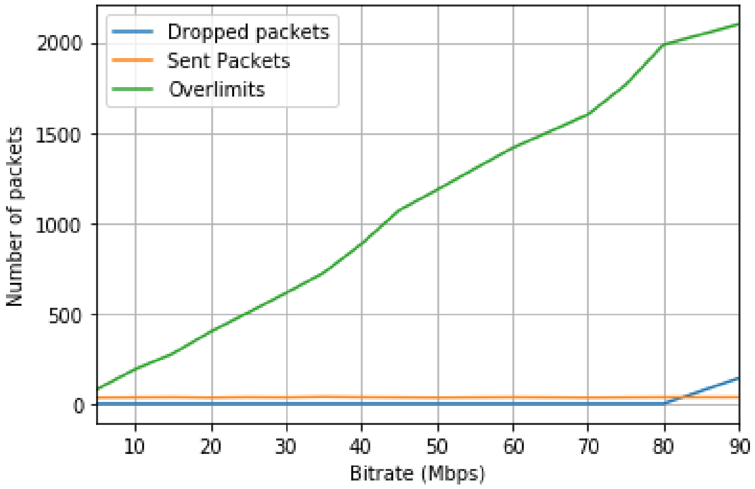


Figure 7.12: **Packet loss for Wi-Fi under proposed system design using TBF**

The results for different Wi-Fi bitrates are shown in Figure 7.12. The size of the buffer was 16 Mb for the TBF qdisc. We do not record any data loss until the bitrate of 80 Mbps as the buffer is large enough to store the excess packets according to Equation 7.3. As video streaming even in Ultra HD requires a maximum throughput of 25 Mbps, we can confidently say that our system design causes no packet loss even under high loads.

This QoE aspect is extremely important for the end-user as data losses or excessive delay will lower the overall acceptability of an application as perceived by the end-user. Our system design can cope with high loads with zero packet loss with the addition of little delay. We see as we increase the bitrate further we get packet losses as there is no space to place the packets and they get dropped. For bitrates over 2 Mbps, the TBF enters an overlimit situation as introduced in Section 5.4.3. We observe a linear increase in the number of overlimits with increasing bitrates. This can be attributed to the increase in the number of packets waiting for tokens for their transmission. As we increase the bitrate more packets will have to wait for tokens which increases the number of overlimits. The transmitted packets report a jitter of 6 ms which conforms to the recommended value being less than 30 ms[21]. These stored packets are transmitted as soon as the rate of TBF is increased i.e. after 200 ms.

By using this system design we can make certain Zigbee can transmit its packet in a reliable manner with keeping Wi-Fi data loss zero until 80 Mbps. However, the queued Wi-Fi packets do suffer an added latency of 200 ms.

7.3.4 Analysis of the proposed system design Using HTB qdisc

Since TBF can't provide priority to any flows due to the use of a single queue, we decided to also test our system design with an HTB qdisc which can provide priority between different flows. The HTB was configured as follows -

Listing 7.4: **HTB configuration**

```
tc qdisc change dev wlan0 root handle 1: htb default 10
tc class add dev wlan0 parent 1: classid 1:1 htb rate 2
   ↪ mbit
tc class add dev wlan0 parent 1:1 classid 1:10 htb prio 1
   ↪  rate 1kbit ceil 2mbit
tc class add dev wlan0 parent 1:1 classid 1:20 htb prio 0
   ↪  rate 2mbit ceil 2mbit
```

The first command attaches HTB qdisc to the interface wlan0 and gives it an identifier called "handle" 1: . The handle and classid numbers are qdisc and classid identifiers of the structure major:minor. Classes who share a parent have to have the same major number, and have a separate minor number. A queuing discipline's minor number is always zero, and can thus be omitted during declaration [4]. The default 10 means that any traffic that is otherwise unclassified will be assigned to class with classid 1:10. Next, we create a root class with classid 1:1 under the qdisc 1: and rate limit it to 2 Mbps. Next, we add two child classes to the root qdisc with classid 1:10 and 1:20. The prio parameter ensures that the class with higher priority (lower number) is served first. We give the low priority class a guaranteed bitrate of 1 Kbps and the ceil rate of 2 Mbps. The high priority class is configured with a guaranteed rate of 2 Mbps and the ceil rate of 2 Mbps. The ceil parameter is used such that each class can attain the maximum bandwidth of 2 Mbps by borrowing tokens from the parent class whenever needed. The lower priority class is given a very low guaranteed rate so that it doesn't increase the overall bitrate of the root qdisc.

To imitate the transmission of high priority packets, we also installed a client application on Wi-Fi node 2 as described in Figure 7.4. This application sends UDP traffic to Wi-Fi node 1 via the router. We want to assign this traffic priority over the traffic sent from Wi-Fi node 1 to Wi-Fi node 2. To filter high priority packets to class 1:20 we use the command shown in Listing 7.5.

Listing 7.5: **u32 Filter Configuration**

```
tc filter add dev wlan0 parent 1:1 u32 match ip dst
    ↪ 192.168.1.180 flowid 1:20
```

It employs the u32 filter which enqueues packets with destination IP as 192.168.1.180 (Wi-Fi node 1) to the high priority class 1:20 of the parent HTB qdisc[34]. In actual implementations, the u32 filter can be used to filter on the commonly known port numbers (3478-3481) used by high priority applications such as VoIP, video conferencing, online gaming, etc. This could be configured as below -

Listing 7.6: **dport matching using u32**

```
tc filter add dev wlan0 parent 1:1 u32 match ip dport
    ↪ 3478:3481 flowid 1:20
```

The full configuration makes sure that both flows can individually achieve the maximum rate of 2 Mbps if required. However, if the higher priority class keeps receiving packets then it will starve the lower priority class.

We used the same setup as described in Section 7.3.2 with the addition of Wi-Fi node 2 sends traffic to Wi-Fi node 1. The router treats all packets going towards Wi-Fi node 1 with high priority and enqueues it to the class 1:20 as configured in Listing 7.5. The results are presented in Figure 7.13.

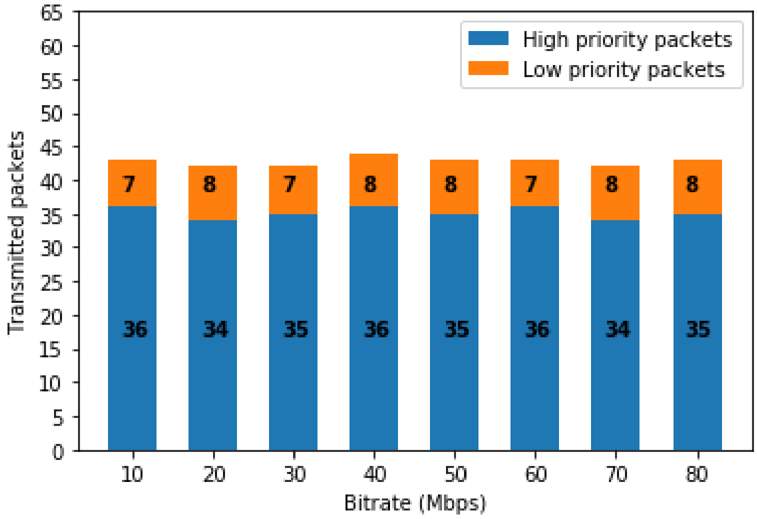


Figure 7.13: **Sent Wi-Fi packets under our system design using HTB**

The class 1:20 achieves the maximum rate of 2 Mbps and send out the corresponding packets. As its guaranteed rate is sufficient enough it does not need to borrow tokens from the root qdisc. The higher priority packets are sent first which decreases their latency. As there could be instances when the high priority class has not enough tokens to transmit its packet, the low priority class can transmit its packets by borrowing token from the root qdisc. Since the root qdisc is also rate-limited it can only lend a certain amount of tokens. This results in a marginal increase of bitrate for the whole system from 2 Mbps to around 2.4 Mbps. As compared to TBF configuration, we observed similar results for Zigbee transmission in terms of RTT and packet loss as the effective bitrate and the time of application of rate-limiting is the same. For that reason, we have not included those results. After 200 ms the higher priority packets are sent first which further decreases their latency, unlike TBF which can't differentiate between these flows. By employing HTB, our system design not only enjoy the benefits of the simple TBF qdisc but can also provide high QoS to selected traffic.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

The plethora of Wi-Fi devices present in homes and their extensive use put excessive pressure on the performance of Zigbee networks. In this work, we first investigated the source of inferior performance of the Zigbee network by conducting a survey and analyzing stored Zigbee data from these systems. After validating the source of interference was indeed the Wi-Fi network, we presented two approaches for a Wireless Lighting System (Philips hue) to mitigate it. In low-density environments, the lighting system can use the inbuilt Wi-Fi chip to find a free non-overlapping channel to operate on. The Wi-Fi chip in monitor mode can scan for nearby Wi-Fi channels and choose an appropriate free Zigbee channel instead of using the default static Zigbee channel set during the first installations.

In high-density environments, we have presented a new centralized approach that can provide coordination between the two heterogeneous networks. By employing custom qdsics and IP packet filtering applications of the Wi-Fi router, our system design can effectively convert the wireless environment to favor Zigbee transmission. By performing multiple experiments for both LOS and NLOS scenarios, we find the rate at which Zigbee can coexist with Wi-Fi traffic under different cases. We also confirm the main source of interference for the Zigbee network is the Wi-Fi router due to its close proximity and high duty cycle. Our flexible system design not only ensures reliable Zigbee performance but also keeps Wi-Fi degradation to a minimum by storing extra packets in the buffer of the router. We show we can reduce packet loss for Zigbee from 67% to 7% even under high interference environments with an average RTT of under 35 ms. The system design is preventive in nature and is only activated during Zigbee transmission which provides no hindrance to Wi-Fi under normal operation. It can also be used with multi-hop networks as the main cause of interference is the router and it is controlled by our design. Our system is robust even under high Wi-Fi loads (up to 80 Mbps) and inflicts little delay (200ms) for Wi-Fi transmission. The system design along with HTB configuration not only allows us to provide reliable transmission for Zigbee but also provide high QoS to high priority traffic. We only observe any data loss when the combined bitrate is over 80 Mbps which is close to the maximum achievable bitrate in home environments using the IEEE 802.11n standard with 20 MHz channel width[7].

The overall complexity and cost of the system is kept low by utilizing the resources which are already present (Wi-Fi chip) and only making certain changes to the software of the router to accommodate custom filtering and queuing techniques.

8.2 Future Work

The implemented system design shows a working proof-of-concept, but could still be improved and customized for every home setup by utilizing different Wi-Fi thresholds. Some of the points that still need to be addressed include -

1. Dynamic channel change: We proposed to use the Wi-Fi chip to select a free channel in a low-interference environment during the initial installation of the lighting system. However, this method might not always be effective as some Wi-Fi routers are set to automatic channel selection which implies that the channel that was free one day might not be the next day. Continous wireless sniffing is not suggested as it will consume valuable resources and might lead to recurring frequency changes which will result in downtime of the system and cause trouble to the user. One proposed method is to scan the medium once during the night, inform the user about a potential channel switch (if required) and guide the user through the process.

2. Granular packet filtering: This work makes a fundamental assumption that every message going towards the Zigbee Bridge is a control command. This assumption is not universally true as the Wi-Fi router or the mobile device used by the user might send synchronization packets which will not lead to any Zigbee traffic but will result in our system design triggering the Wi-Fi router to lower its data rate. This will lead to unnecessary throttling of the Wi-Fi network. Granular filtering needs to implemented to achieve this. One method could be to filter on the PUSH flag present in the TCP packet. Filtering for PUSH flag will ensure the triggering of the system design will only occur when some data is actually being transferred and not just synchronization/ARP messages [55].

3. Autonomous Zigbee packets: One more area to improve the system design is to include a method to include messages sent by the Zigbee bridge autonomously such as timed schedules and button presses by the user. For these scenarios, the router would not be aware of the Zigbee traffic and our system design won't be triggered. This situation can be tackled by transmitting dummy packets to the router by the bridge to indicate a pre-warning to trigger the system design and lower the bitrate in the future.

4. Duplex communication: As this work only considers singlex communication between the coordinator and the corresponding router, the case of duplex communication also needs to be addressed i.e. from Zigbee router to Zigbee coordinator.

Bibliography

[1] Ieee standard for information technology– local and metropolitan area networks– specific requirements– part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low rate wireless personal area networks (wpans). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, pages 1–320, 2006.

[2] Muhammad Sajjad Akbar, Muhammad Saleem Khan, Kishwer Abdul Khaliq, Amir Qayyum, and Muhammad Yousaf. Evaluation of ieee 802.11n for multimedia application in vanet. In *Procedia Computer Science*, volume 32, pages 953–958, 2014.

[3] Ata Elahi and Adam Gschwender. Introduction to the zigbee wireless sensor and control network. `https://www.informit.com/articles/article.aspx?p=1409785`, 2009. Last accessed: Jun. 15, 2020.

[4] Bert Hubert. Classful queuing disciplines . `https://tldp.org/HOWTO/Traffic-Control-HOWTO/classful-qdiscs.html`, 2002. Last accessed: May. 09, 2020.

[5] Bert Hubert. Simple, classless queueing disciplines. `http://www.tldp.org/HOWTO/Adv-Routing-HOWTO/lartc.qdisc.classless.html`, 2002. Last accessed: May. 09, 2020.

[6] Bert Hubert. Tc linux man page. `https://linux.die.net/man/8/tc`, 2004. Last accessed: May. 09, 2020.

[7] Buffalo Technology. Understanding and optimizing 802.11n. `https://www.lmi.net/wp-content/uploads/Optimizing_802.11n.pdf`, 2011. Last accessed: May. 09, 2020.

[8] Elastic NV. Elastic search. `https://www.elastic.co/`, 2020. Last accessed: May. 09, 2020.

[9] Elastic NV. Kibana dashboard. `https://www.elastic.co/kibana`, 2020. Last accessed: May. 09, 2020.

[10] Errin W. Fulp. Chapter e74 - firewalls. In John R. Vacca, editor, *Computer and Information Security Handbook (Third Edition)*, pages e219 – e237. Morgan Kaufmann, Boston, 2013.

[11] R. G. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti. Experimental assessment of the coexistence of wi-fi, zigbee, and bluetooth devices. In

2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, pages 1–9, 2011.

[12] Stefan Geirhofer, Lang Tong, and Brian Sadler. Cognitive radios for dynamic spectrum access - dynamic spectrum access in the time domain: Modeling and exploiting white space. *IEEE Communications Magazine*, 45(5):66–72, 2007.

[13] Gilles Thonet, Patrick Allard-Jacquin and Pierre Colle . Zigbee wifi coexistence - white paper and test report. `http://vip.gatech.edu/wiki/images/8/8e/Zigbee_WiFi_Coexistence_-_White_Paper_and_Test_Report.pdf`, 2008. Last accessed: May. 14, 2020.

[14] Ramakrishna Gummadi, Hari Balakrishnan, and Srinivasan Seshan. Metronome: Coordinating spectrum sharing in heterogeneous wireless networks. In *2009 First International Communication Systems and Networks and Workshops*, pages 1–10, 2009.

[15] W. Guo, W. M. Healy, and M. Zhou. Impacts of 2.4-ghz ism band interference on ieee 802.15.4 wireless sensor network reliability in buildings. *IEEE Transactions on Instrumentation and Measurement*, 61(9):2533–2544, 2012.

[16] Jan-Hinrich Hauer, Andreas Willig, and Adam Wolisz. Mitigating the effects of rf interference through rssi-based error recovery. In *Proceedings of the 7th European Conference on Wireless Sensor Networks*, EWSN10, page 224239, Berlin, Heidelberg, 2010. Springer-Verlag.

[17] Anwar Hithnawi. Exploiting physical layer information to mitigate cross-technology interference effects on low-power wireless networks. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, SenSys 13, New York, NY, USA, 2013. Association for Computing Machinery.

[18] James Hou, Benjamin Chang, Dae-Ki Cho, and Mario Gerla. Minimizing 802.11 interference on zigbee medical sensors. In *Proceedings of the Fourth International Conference on Body Area Networks*, BodyNets 09, Brussels, BEL, 2009. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[19] Jun Huang, Guoliang Xing, Gang Zhou, and Ruogu Zhou. Beyond coexistence: Exploiting wifi white space for zigbee performance assurance. In *The 18th IEEE International Conference on Network Protocols*, pages 305–314, 2010.

[20] Hongwei Huo, Youzhi Xu, Celal Can Bilen, and Hongke Zhang. Coexistence issues of 2.4ghz sensor networks with other rf devices at home. In *2009 Third International Conference on Sensor Technologies and Applications*, pages 200–205, 2009.

[21] I. Rec. y. 1541: Network performance objectives for ip-based services, international telecommunication union, itu-t, 2003.

[22] IEEE SA. 802.15.4-2015 - ieee standard for low-rate wireless networks. https://standards.ieee.org/standard/802_15_4-2015.html, 2015. Last accessed: May. 29, 2020.

[23] IHS Markit. Iot trend watch 2018. https://cdn.ihs.com/www/pdf/IoT-Trend-Watch-eBook.pdf, 2018. Last accessed: May. 25, 2020.

[24] Juniper Networks. Router data flow overview. https://www.juniper.net/documentation/en_US/junos/topics/concept/firewall-filter-router-data-flow-overview.html, 2020. Last accessed: Jun. 15, 2020.

[25] Justin Ellingwood. A deep dive into iptables and netfilter architecture. https://www.digitalocean.com/community/tutorials/a-deep-dive-into-iptables-and-netfilter-architecture, 2015. Last accessed: Jun. 15, 2020.

[26] Peter Kacz, Ondrej Hyneica, Petr Fiedler, Zdenek Bradaeora, and Pavel Kucera. Range test with zigbee in indoor environments. *IFAC Proceedings Volumes*, 39(21):447 – 451, 2006.

[27] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In *IECON 2007 - 33rd Annual Conference of the IEEE Industrial Electronics Society*, pages 46–51, 2007.

[28] Ping Li, Yubo Yan, Panlong Yang, Xiang-Yang Li, and Qiongzheng Lin. Coexist wifi for zigbee networks with fine-grained frequency approach. *IEEE Access*, 7:135363–135376, 2019.

[29] Chieh-Jan Mike Liang, Nissanka Bodhi Priyantha, Jie Liu, and Andreas Terzis. Surviving wi-fi interference in low power zigbee networks. *SenSys '10: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, 15(4), November 2010.

[30] LINKSYS Inc. . Linksys ea6350 wi-fi router. https://www.linksys.com/us/p/P-EA6350/, 2018. Last accessed: May. 12, 2020.

[31] Kui Liu. Performance evaluation of zigbee network for embedded electricity meters. Master thesis, Kungliga Tekniska Hgskolan Royal Institute of Technology, Stockholm, Sweden, 2009.

[32] M. A. Brown. Traffic control howto version 1.0.2. http://tldp.org/HOWTO/Traffic-Control-HOWTO, 2006. Last accessed: May. 09, 2020.

[33] Metageek . Zigbee, wi-fi coexistence chart. https://www.metageek.com/training/resources/zigbee-wifi-coexistence.html, 2020. Last accessed: May. 09, 2020.

[34] Michael Kerrisk. tc-u32. https://man7.org/linux/man-pages/man8/tc-u32.8.html, 2004. Last accessed: May. 09, 2020.

[35] Isbat Uzzin Nadhori, Amang Sudarsono, and Ridho Luberski. Analysis of slotted and unslotted csma/ca wireless sensor network for e-healthcare system. In *2014 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*, pages 53–57, 2014.

[36] Netflix Inc. . Internet connection speed recommendations. `https://help.netflix.com/en/node/306`, 2019. Last accessed: May. 12, 2020.

[37] NXP Laboratories UK. Zigbee light link user guide. `https://www.nxp.com/docs/en/user-guide/JN-UG-3091.pdf`, 2016. Last accessed: May. 29, 2020.

[38] NXP Semiconductors. Zigbee stack 3.0 user guide. `https://www.nxp.com/docs/en/user-guide/JN-UG-3113.pdf`, 2018. Last accessed: May. 14, 2020.

[39] OpenWrt Community . Welcome to the openwrt project. `https://openwrt.org/`, 2020. Last accessed: May. 12, 2020.

[40] Marina Petrova, Lili Wu, Petri Mahonen, and Janne Riihijarvi. Interference measurements on performance degradation between colocated ieee 802.11g/n and ieee 802.15.4 networks. In *Sixth International Conference on Networking (ICN'07)*, pages 93–93, 2007.

[41] Phil Hagen. iptables processing flowchart. `https://stuffphilwrites.com/2014/09/iptables-processing-flowchart/`, 2014. Last accessed: Jun. 15, 2020.

[42] Sofie Pollin, Ian Tan, Bill Hodge, Carl Chun, and A.hmad Bahai. Harmful coexistence between 802.15.4 and 802.11: A measurement-based study. In *2008 3rd International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CrownCom 2008)*, pages 1–6, 2008.

[43] Hariharan Rahul, Nate Kushman, Dina Katabi, Charles Sodini, and Farinaz Edalat. Learning to share: Narrowband-friendly wideband networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):147158, August 2008.

[44] Rusty Russell. Packet filtering howto. `https://netfilter.org/documentation/HOWTO//packet-filtering-HOWTO.html`, 2002. Last accessed: May. 29, 2020.

[45] Semtech Corp. Wireless rf products. `https://www.semtech.com/products/wireless-rf/24-ghz-transceivers`, 2020. Last accessed: Jun. 15, 2020.

[46] Z. Shah, S. Rau, and A. Baig. Throughput comparison of ieee 802.11ac and ieee 802.11n in an indoor environment with interference. In *2015 International Telecommunication Networks and Applications Conference (ITNAC)*, pages 196–201, 2015.

[47] SI Labs . Efr32mg12 gecko multi-protocolwireless soc family data sheet. `https://www.silabs.com/documents/public/data-sheets/efr32mg12-datasheet.pdf`, 2016. Last accessed: May. 12, 2020.

[48] SI Labs . Managing coexistence between wi-fi, zigbee, thread, and bluetooth. https://www.silabs.com/products/wireless/learning-center/wi-fi-coexistence, 2017. Last accessed: May. 09, 2020.

[49] SI Labs . Ug261: Efr32mg12 2.4 ghz 10 dbmradio board user's guide. https://www.silabs.com/documents/public/user-guides/ug261-brd4162a.pdf, 2017. Last accessed: May. 12, 2020.

[50] Signify NV. Meet the hue. https://www.philips-hue.com/nl-nl, 2020. Last accessed: May. 11, 2020.

[51] Signify NV. Web api philips hue. https://developers.meethue.com/develop/get-started-2/, 2020. Last accessed: May. 11, 2020.

[52] A. Sikora and V. F. Groza. Coexistence of ieee802.15.4 with other systems in the 2.4 ghz-ism-band. In *2005 IEEE Instrumentationand Measurement Technology Conference Proceedings*, volume 3, pages 1786–1791, 2005.

[53] SILICON LABS. Zigbee retry in emberznet stack of silabs. https://www.silabs.com/community/wireless/zigbee-and-thread/knowledge-base.entry.html/2012/06/29/how_does_the_emberzn-po1M, 2012. Last accessed: Jun. 15, 2020.

[54] SILICON LABS. Ug103.2: Zigbee fundamentals. https://www.silabs.com/documents/public/user-guides/ug103-02-fundamentals-zigbee.pdf, 2017. Last accessed: Jun. 15, 2020.

[55] Stretch. Tcp flags: Psh and urg. https://packetlife.net/blog/2011/mar/2/tcp-flags-psh-and-urg/, 2011. Last accessed: Jul. 30, 2020.

[56] Lieven Tytgat, Opher Yaron, Sofie Pollin, Ingrid Moerman, and Piet Demeester. Avoiding collisions between ieee 802.11 and ieee 802.15.4 through coexistence aware clear channel assessment. *EURASIP Journal on Wireless Communications and Networking*, 15(4), August 2012.

[57] Jianfeng Wang. Zigbee light link and its applicationss. *Wireless Communications, IEEE*, 20:6–7, 2013.

[58] Jianfeng Wang. Zigbee light link and its applicationss. *Wireless Communications, IEEE*, pages 6–7, Aug 2013.

[59] Lei Yang, Wei Hou, Lili Cao, Ben Y. Zhao, and Haitao Zheng. Supporting demanding wireless applications with frequency-agile radios. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation*, NSDI10, page 5, USA, 2010. USENIX Association.

[60] Peizhong Yi, Abiodun Iwayemi, and Chi Zhou. Developing zigbee deployment guideline under wifi interference for smart grid applications. *IEEE Transactions on Smart Grid*, 2(1):110–120, 2011.

[61] Wei Yuan, Xiangyu Wang, and Jean-Paul M. G. Linnartz. A coexistence model of ieee 802.15.4 and ieee 802.11b/g. In *2007 14th IEEE Symposium on Communications and Vehicular Technology in the Benelux*, pages 1–5, 2007.

[62] Xinyu Zhang and Kang S. Shin. Enabling coexistence of heterogeneous wireless systems: Case for zigbee and wifi. *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, May 2011.

[63] Ruogu Zhou, Guoliang Xing, Yongping Xiong, Limin Sun, and Jian Ma. Zifi: wireless lan discovery via zigbee interference signatures. *Proceedings of the annual international conference on mobile computing and networking (MOBICOM)*, September 2010.