PERFORMANCE EVALUATION OF AN IEEE 802.11 MOBILE AD-HOC NETWORK ON THE RASPBERRY PI

ORANGEL JOSÉ AZUAJE CONTRERAS
DISSERTAÇÃO DE MESTRADO APRESENTADA À FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO EM ÁREA CIENTÍFICA
Performance Evaluation of an IEEE 802.11 Mobile Ad-Hoc Network on the Raspberry Pi

Orangel José Azuaje Contreras

Dissertação realizada no âmbito do
Mestrado em Engenharia da Informação

Advisor: Prof. Ana Aguiar
Co-Advisor: Prof. Pedro Brandão

October 2015
Abstract

The Project Vital Responder 2.0 – Gestão Inteligente de eventos críticos de stress, fadiga e intoxicação pelo fumo no combate a fogos florestais has as main purpose to design a system for decision support in ground-based firefighting operations, in a cooperative manner between the firemen monitoring data (e.g. fatigue, stress, vital signs) and the environmental variables (e.g. soil characteristics, weather, CO levels). To handle the sending of the fireman monitoring data, it has been proposed the deployment of a Mobile Ad-Hoc Network (MANET), where each fireman carries a node and all of them participate in the routing-forwarding mechanism to send the data towards the firemen truck, i.e. the sink.

In this dissertation, we evaluated experimentally the performance of an IEEE 802.11 MANET on a low cost computer, known as Raspberry Pi (RPi), establishing the feasibility of the deployment of an application in the MANET that supports the Project Vital Responder 2.0. We also evaluated theoretically the end-to-end delay that experiences a MANET while changing its topology.

The experimental performance evaluation was addressed at different layers namely physical, network and application layer; hardware performance is also measured in terms of current drawn and CPU usage for the RPi in different test scenarios. Moreover, the MANET connectivity was tested with real world GPS data obtained in the context of the aforementioned project.

The theoretical performance evaluation was conducted with an analytical framework known as Network Calculus. In this evaluation, we explored the characteristics of this theoretical framework for analyzing service guarantees in wireless networks, where Min-Plus Algebra, Moment Generating Function and Finite State Markov Chain model were addressed. Cases of a single fireman and firemen in series are studied through an analytical model.

The experimental results show that the RPi needs a battery that can supply a minimum of 400 mA, the communication range is highly dependent on the WiFi-USB adapter with values between 30 to 150 m and the maximum application level throughput is 4 Mbps for the two-node case. Regarding the theoretical result, we showed how the end-to-end delay bound can change from 264 up to 888 ms while changing the network topology.

Key-words: wireless, nodes, throughput, 802.11, Network Calculus
Acknowledgements

Thanks to Prof. Ana Aguiar for her encourage, suggestions and especially for her trust during the realization of the thesis and the whole Master program.

Thanks to my mother, father and siblings for their support.

Thanks to my wife Yanmeric for being the inspiration of my life.

Thanks to God for make all the things possible.
Index

Index............................................................................................................................................. vii
Figure List ...................................................................................................................................... x
Table List ...................................................................................................................................... xii
Abbreviations and Symbols ......................................................................................................... xiv
Chapter 1 Introduction.................................................................................................................. 1
  1.1 Description of the Problem ................................................................................................. 1
  1.2 Objectives ............................................................................................................................ 2
  1.3 Structure of the Thesis ........................................................................................................ 2
Chapter 2 Mobile Ad-Hoc Networks ............................................................................................. 3
  2.1 What are Mobile Ad-Hoc Networks? .................................................................................. 3
  2.3 IEEE 802.11 .......................................................................................................................... 4
  2.4 Routing Protocols for MANETs............................................................................................ 4
  2.5 Network Device: Raspberry Pi............................................................................................. 6
  2.6 MANET Performance ........................................................................................................... 7
  2.7 Related Work....................................................................................................................... 7
Chapter 3 Network Calculus........................................................................................................ 10
  3.1 Min-Plus Algebra Basics .................................................................................................... 10
  3.2 Quality of Service Guarantees ........................................................................................... 11
  3.3 Stochastic Network Calculus ............................................................................................. 11
  3.4 Moment Generating Function ........................................................................................... 11
  3.5 Modeling the Fading Channel ............................................................................................. 13
  3.5.1 Coherence Time and Doppler Spread ............................................................................ 14
  3.6 Scheduling Discipline and Leftover Service Characterization ........................................... 14
Chapter 4 Experimental Evaluation ............................................................................................. 16
  4.1 Methodology ..................................................................................................................... 17
  4.1.1 Ad-Hoc Network ............................................................................................................. 17
  4.1.2 Current Consumption .................................................................................................... 17
  4.1.3 Communication Range ................................................................................................... 19
  4.1.4 Connectivity ..................................................................................................................... 20
  4.1.5 Throughput ..................................................................................................................... 21
  4.2 Implementation and Results ............................................................................................. 23
Figure List

Figure 2.1 Raspberry Pi ................................................................. 6
Figure 3.1 Network Calculus representation ......................................... 10
Figure 3.2 Equivalent server-Server in series ........................................ 12
Figure 3.3 Gilbert-Elliot model ........................................................... 13
Figure 3.4 Queuing system with through and cross traffic .......................... 15
Figure 3.5 Through-flow in server in series ........................................... 15
Figure 4.1 Resistive-sensing ............................................................... 18
Figure 4.2 High-side current sensing ..................................................... 19
Figure 4.3 Communication range set up ................................................ 19
Figure 4.4 Transitive closure ............................................................... 20
Figure 4.5 One hop-One flow ............................................................. 21
Figure 4.6 Two hop-One client ............................................................ 22
Figure 4.7 Two hop-Two clients .......................................................... 22
Figure 4.8 Ad-Hoc properties .............................................................. 23
Figure 4.9 WiFi-USB adapter (left:N150 Ralink, middle: Wi-Pi, right: TP-Link) 24
Figure 4.10 Test circuit .................................................................... 25
Figure 4.11 Current drawn on the RPi ................................................... 25
Figure 4.12 Current evolution at 100 kbps ............................................ 26
Figure 4.13 Current evolution at 1 Mbps ............................................... 27
Figure 4.14 Packet loss and RSSI – N150 Ralink Adapter .................... 28
Figure 4.15 Packet loss and RSSI - WiPi Adapter ................................. 29
Figure 4.16 Packet loss and RSSI - TP-Link Adapter ......................... 30
Figure 4.17 Communication map ........................................................ 30
Figure 4.18 Distance from a fireman to the sink ..................................... 31
Figure 4.19 Minimum distance from a fireman to any other .................... 32
Figure 4.20 Distribution of time of disconnection (x-axis in log scale) .......... 33
Figure 4.21 Multi-hop configuration .................................................... 34
Figure 4.22 Per-scenario throughput .................................................. 34
Figure 4.23 CPU-usage .................................................................... 35
Figure 4.24 Throughput - Two flows ................................................... 35
Figure 4.25 Throughput - TCP ........................................................... 36
Figure 5.1 One server - One flow ....................................................... 38
Figure 5.2 Two servers - One flow ...................................................... 38
Figure 5.3 One server - Two flows ..................................................... 38
Figure 5.4 Two servers - Two flows .................................................... 39
Figure 5.5 Analytical bound and simulation .......................................... 39
Figure 5.6 Analytical bounds for several scenarios ............................... 40
Table List

Table 4.1 Chipset/Driver in ad-hoc network ................................................................. 23
Table 4.2 RPi and ad-hoc consumption ................................................................. 27
Table 4.3 Consumption at different data rates ....................................................... 27
Table 4.4 Probability of connection with another firemen ........................................ 32
Table 4.5 Probability of connection with the sink in a same second ......................... 33
Table 5.1 Source and channel model parameters ................................................... 38
Table 5.2 Parameters used to validate ................................................................. 39
Abbreviations and Symbols

Access Point             AP
Bits per second          bps
Carrier Sense Multiple Access CSMA
Distributed Coordination Function DCF
Frequency Division Multiple Access FDMA
Global Positional System GPS
Industrial, Scientific and Medical ISM
Inter-Integrated Circuit I2C
Institute of Electrical and Electronics Engineers IEEE
Medium Access Control MAC
Mobile Ad-Hoc Network MANET
Optimized Link State Routing Protocol OLSR
Orthogonal Frequency Division Multiplexing OFDM
Quality of Service QoS
Raspberry Pi RPi
Received Signal Strength Indicator RSSI
Transmission Control Protocol TCP
Time Division Multiple Access TDMA
User Datagram Protocol UDP
Wireless Local Area Network WLAN
Chapter 1

Introduction

The use of sensors to monitor physical or environmental variables such as pressure, sound, temperature or even electrical activity of the heart (electrocardiogram) has increased in recent years, also giving life to the new Internet of Things and wearable technology [1].

These kinds of sensors in a wireless network allow us to imagine different areas where they can be deployed such as emergency, research, military scenarios, among others. One of these deployments is the aforementioned project Vital Responder 2.0 – Gestão Inteligente de eventos críticos de stress, fadiga e intoxicação pelo fumo no combate a fogos florestais. The firemen monitoring data is gathered in a device with network capabilities and then sent it to the sink, where it can be used in a decision support system to react to some events (e.g. life-threatening to a fireman).

Several wireless technologies have been proposed but they were found inappropriate for deployment. Cellular technology was one of the proposals to handle the transmission of the data, however, it is not a solution because commonly there is no coverage in remote areas where fires often take place. Infrastructure mode was another proposal, where an access point (AP) is deployed in the truck and the firemen (nodes) are connected to it, however, it was rejected because of the limited range.

1.1 Description of the Problem

Taking into account the disadvantages of the previous technologies, a new technology was proposed to handle the data transmission: Mobile Ad-Hoc Networks (MANET). This technology does not need a pre-established infrastructure and it overcomes the direct link problem with the ability in each node to act as a router and re-transmit data on behalf of the other nodes, that is, multi-hopping communication.

The project establishes the use of network devices with WiFi support, which is a well-established technology based on the 802.11 standards, to build the ad-hoc network, mainly because WiFi devices are ubiquitous and available “off-the-shelf”. At the same time, the network device chosen was the Raspberry Pi (RPI), which is a low cost credit card–sized single-
board computer that it is available in the market and runs Linux-kernel-based operating systems, making it a good alternative to test any product for software or network engineers.

For the success of the project Vital Responder 2.0, it is important to know whether it is possible to implement an IEEE 802.11 MANET on the RPi and if it is possible, what performance can it provide.

1.2 Objectives

This dissertation has as main objective to study the performance of an IEEE 802.11 MANET through an experimental and theoretical evaluation. The experimental evaluation comprises the deployment of the network on the RPi and analyzing the performance through the following metrics: current consumption, communication range, connectivity and throughput. The theoretical evaluation involves the analysis of end-to-end delay bounds to different network topologies using an analytical framework known as Network Calculus. These results will allow us to determine the feasibility of supporting the Vital Responder 2.0 application in a MANET.

We also have secondary objectives in the developing of this dissertation:

- Building a test bed of RPi connected in ad-hoc for future research in areas such as MAC, routing and transport protocols

- Building a connectivity framework, where Global Positioning System (GPS) data of each node in a MANET is given and the software will calculate statistic of the connectivity between nodes and a possible sink.

1.3 Structure of the Thesis

Chapter 2 introduces MANETs with a brief description of the IEEE 802.11 MAC and routing protocols used in this kind of networks, the network device used in the thesis, i.e., the Raspberry Pi, as well as the metrics that will be used to measure the experimental performance of the network. It also presents a survey of works where aspects related to this work have been addressed in other research projects.

Chapter 3 introduces the theoretical framework selected to analyze the changes in the network topology, that is, Network Calculus. We present its use in the study of the service guarantees of wireless communications systems, with special attention to subjects like Min-Plus algebra, Moment Generating Function and Markov Channel Model.

Chapter 4 presents the methodology adopted to measure the performance of the MANET through the following metrics: current consumption, communication range, connectivity and throughput. The chapter also shows the implementation details and experimental results.

Chapter 5 shows the methodology employed to calculate the end-to-end delay bound for different network topologies through Network Calculus. Afterwards, we introduce modeling parameters and related them to our case and finally, implementations and results are presented.

Chapter 6 presents the conclusions and contributions as well as the future work that can be developed from the results of this work.
Chapter 2

Mobile Ad-Hoc Networks

2.1 What are Mobile Ad-Hoc Networks?

A Mobile Ad-Hoc Network (MANET) is a collection of mobile devices (nodes or routers)1 dynamically forming a temporary network, without the requirement of any existing infrastructure or prior network configuration [2]. As the devices are free to move and organize themselves arbitrarily, the network wireless topology may change rapidly and the communication between the nodes might be through single or multiple-hop wireless link.

The notion of ad-hoc can be traced back to the 1970’s, when U.S. Defense Advanced Research Projects Agency (DARPA) implemented the Packet Radio Network (PRNET), which is considered the precursor of MANETs[3]. Since then, MANETs is one of the most active fields in communications and have been mostly used in military and emergence applications.

In summary, the main characteristics of MANETs are [4]:

- Distributed operation: the control of the network is distributed among the nodes because there is no centralized authority. The nodes in the MANET are independent and communicate among themselves, acting as both as a host or a router, establishing multiple-hop communication.

- Dynamic topology: nodes are free to move randomly, thus, the network topology may change in an unpredictable manner along the time. In addition, the nodes can join or leave the network anytime.

- Constrained resources: mobile devices in MANETs are usually powered by batteries that proportion a limited working time for the nodes; making the power source a constraint in this kind of network. Another limitation is bandwidth, where multiple

1 From now on, the terms node and router are going to be used interchangeably.
access, fading, interference, noise, etc. affect the capacity of the shared wireless medium

2.2 - Issues in MANETs

Despite the several benefits of MANETs, as all technologies, it poses some disadvantages and in this case, some of them are inherit from wireless communications and others from the ad-hoc architecture [5]:

- The wireless channel is unreliable and unprotected from external signals.
- The wireless channel has time-varying and symmetric propagation properties.
- The wireless signals suffer from path loss and interference.
- The mobile devices have limited energy resource.
- The Medium Access Control (MAC) protocols are not optimized for multiple-hop environment.
- The routing protocols have to deal with construction and maintenance of paths.

2.3 IEEE 802.11

IEEE 802.11 Distributed Coordination Function (DCF) is an access method for asynchronous, contention-based and distributed access to the wireless channel. DCF implements Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA) technology to share the wireless medium among several nodes [4].

In CSMA/CA, when a node receives a packet that is to be sent, it senses the channel to be sure that is clear, that is, no other node is transmitting at the time. If the channel is clear, the packet is sent, conversely, if the channel is not clear, the node waits for a random time and then re-senses the channel. This random period of time is the so-called backoff factor and it is counted down by a corresponding backoff counter. If the channel is clear when the counter is equal to zero, the node can transmit the packet but if the channel still not clear, the backoff factor is set again and the process is repeated.

2.4 Routing Protocols for MANETs

In MANETs, when a node tries to send data to other nodes that are out of its communication range, the packet should be forwarded via one or more intermediate nodes. In order to make this multiple-hop communication possible, a routing protocol is needed for a node to find the communication route.

In order to perform properly, the routing protocols for MANETs have to confront the following issues [6]:

- Route failure and recovery: the protocol should respond quickly to any failure in the network (e.g. broken link) to guarantee low traffic delay.
- Bandwidth efficient: the protocol should minimize the number of periodic updates to use properly the limited bandwidth.
• Energy efficient: the protocol should have the minimum possible of processing and transmission requirements to use in an optimal manner the battery of the device.

2.4.1 – Classification of MANET Routing Protocols

Depending on the routing discovery process implemented, MANETs routing protocols are typically classified into proactive, reactive and hybrid.

In proactive or table-driven routing protocol, each node has to maintain routing information to the other nodes and they keep it up to date by exchanging route updates throughout the network. These protocols lead us to a minimal routing delay given that the node can determine quickly if the destination is reachable or not at the expense of heavy routing overhead [5]. Proactive protocols include Destination-Sequenced Distance-Vector (DSDV), Optimized Link State Routing Protocol (OLSR), Babel, among others.

On the other hand, a reactive routing protocol, also called on-demand, does not maintain routing information if there is no communication. The route is discovered when a node wants to send a packet to a certain destination, that is, only when it is needed. As a result, these kind protocols produce less routing overhead but there is always a delay in the communication because of the route discovery process [5]. Reactive protocols include protocols such as Dynamic Source Routing (DSR) and Ad-Hoc On-Demand Distance Vector (AODV).

Hybrid routing protocols is a combination of proactive and reactive characteristics, where usually the network is partitioned and depending on the zone, proactive or reactive protocol may be used. Hybrid protocols include Zone Routing Protocol (ZRP), Zone-Based Hierarchical Link State (ZHLS) Routing Protocol, Hybrid Ad Hoc Routing Protocol (HARP), among others.

2.4.2 OLSR Routing Protocol

Optimized Link State Routing [7] is a table driven protocol developed for MANETs that is based on link-state protocols, where the only information passed between nodes is connectivity related. OLSR is optimized for MANETs in the sense that compact the size of the information in the messages and reduces the number of retransmission to flood these messages.

To minimize the flooding, OLSR uses the concept of Multipoint Relay (MPR) and Multipoint Relay Selector (MPR Selector). First, each node in the network selects a set of nodes in its neighborhood to diffuse its message; these selected nodes are called MPRs of that node. The MPRs are selected in such a manner that the set covers all the nodes that are two hops away. Second, each node maintain a list of the nodes, called MPR Selectors, from which has been selected as MPR and that node will retransmit only the messages received from the nodes in its MPR Selector set.

The main steps in the OLSR operation are [8]:

• Neighbor sensing: each node detects its neighborhood by periodically broadcasting HELLO messages, containing the information about its neighbors and their link status. These HELLO messages allow each node to get the information of its neighbor up to two nodes and then, based in this information the MPR selection is performed.
TC messages dissemination: each node broadcast its link state information using control messages called Topology Control (TC), in order to build the routing table. TC messages are forwarded in the network, taking the advantage of MPRs.

Routing table calculation: the nodes which receive a TC message will look into the connected pairs in the TC message list and then, they will compute the route by tracking in descending order.

2.5 Network Device: Raspberry Pi

The Raspberry Pi (RPi) [9] is a low-cost credit-card size Linux computer developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of computer programming in schools.

The RPi has fully filled all the expectations at the education level, but it has also found a place in the electronic and computer hobbyist and lately in the research area. Its price, operating system and hardware properties fit perfectly in the design of prototypes, test bed and in some case, final products.

There have been different versions of the RPi: model A, model A+, model B, model B+ and the last one, released in February 2015, Raspberry Pi 2. In this dissertation, we will focus in the RPi model B+ of the Figure 2.1

![Raspberry Pi](image)

**Figure 2.1 Raspberry Pi**

2.5.1 Hardware specifications

The RPi has the following hardware specifications:

- Processor: The RPi has integrated circuit (IC) that incorporates all components of a computer into a single chip, better known as a System on Chip (SoC). The Raspberry Pi is based on the Broadcom BCM2835, which includes a 700 MHz ARM11 processor.

- RAM: The BCM2835 SoC also includes a 512MB RAM, which should be shared between the Central Processing Unit (CPU) and Graphics Processing Unit (GPU).

- Networking: The RPi has an 8P8C (“RJ45”) Ethernet port provided by a built-in USB Ethernet adapter and also can be connected to a network using an external WiFi-USB adapter, which can be connected in one of the 4 available USB ports.

---

2 Raspberry Pi Model B+
2.6 MANET Performance

Usually the performance of a network is just measured by the amount of data that it “can carry on it”. However, if we look at the MANET as a whole, we can measure the performance through the following metrics:

- **Hardware-based**: the MANET is built in mobile devices with network capabilities, thus, we have to consider physical aspects of the device such as the current drawn and/or CPU usage.

- **Communication range**: it is the range (with respect to the transmitting node) within which a transmitted packet can be successfully received. The communication range is mainly determined by the transmission power and the radio propagation properties.

- **Connectivity**: it is the capacity of nodes to communicate because they are in each other’s communication range or via multiple-hop.

- **Throughput**: it is the rate of successful message delivery over a communication channel, in our case, the wireless channel. Throughput is usually measured in bits per second (bps) or in packets per second (pps).

- **End-to-end delay**: it is the time taken for a packet to be transmitted across a network from source to destination. In MANETs, given the continuous changes in topology, it will experience a range of values that will affect the Quality of Service (QoS).

2.7 Related Work

In [10] A. Paramanathan et al. used the software library KODO in the Raspberry Pi model B to design a testbed for network coding protocols. Despite the fact that the RPi are stationary in their experiments, they implemented Better Approach To Mobile Adhoc Networking (B.A.T.M.A.N.) as routing protocol and reported for the RPi an average current measurement of 447.5 mA in idle state.

In [11] G. Anastasi et al. investigated the performance of IEEE 802.11b ad-hoc network with a set of measurements. The measurement test-bed was based on laptops running the Linux-Mandrake 8.2 operating system and equipped with D-LinkAir DWL-650 cards. Their work revealed that different transmission ranges and carrier-sensing ranges may exist at the same time in the network due to different transmission rates for control and data frames. They stated that simulation studies (e.g. ns-2 simulation tool) usually neglect these aspects and also, consider much larger values than achievable in practice for transmission ranges. They reported values of transmission range between 110 – 130 meters with a data rate of 1 Mbps, 90 – 100 meters at 2 Mbps, 70 meters at 5.5 Mbps and 30 meters at 11 Mbps. Finally, from the results, they conclude that the transmission range is highly dependent on the data rate and also highly variable even during the same experiment, depending on several factors such as whether condition, propagation conditions, etc.

In [12], M. Abolhasan et al. investigated the performance of three mesh routing protocols: OLSR, B.A.T.M.A.N. and Babel. They focused the investigation on the multi-hopping performance and the ability to recover from link failures. This performance evaluation was conducted in an indoor multi-hop mesh network testbed of Wireless Router Application
Platform (WRAP) boards from PCEngines with Voyage Linux OS. They concluded that B.A.T.M.A.N. achieves the highest level of stability and packet delivery, while BABEL offers the highest multi-hop bandwidth and the fastest route repair time. Both B.A.T.M.A.N. and BABEL outperformed OLSR in all performance metrics examined.

In [13] P Ng and S. Liew attempted to identify the maximum throughput that can be sustained in an 802.11b multi-hop network considering the impact of hidden nodes and carrier sensing. The first study was with a simulation using ns2.1b9, where all the nodes communicate through a half-duplex wireless radio on the 802.11 DCF with data and basic rates set at 11 Mbps, the RTS/CTS mechanism was turn off, the nodes where stationary with a transmission range of 250 m and carrier-sensing range of 550 m; in a chain of 8 nodes, they reported an average throughput of 1.83 Mbps at the first hop and 1.13 Mbps at the last hop. They also set up a real 6-node multi-hop network with six symmetric DELL Latitude D505 laptop PCs with 1.5 GHz Celeron Mobile CPU, 512 MB RAM and Buffalo WLI2-CF-S11 IEEE 802.11b Wireless LAN card and repeated the same single-flow investigation, obtaining a maximum sustainable throughput of 1.25 Mbps, that is, offered load beyond 1.25 Mbps was unsustainable and high packet loss rate for the flow would occur.

Regarding specific solutions for emergency services, the authors in [14] present the BREathing rate MONitoring (BREMON), a smartphone-based framework that monitors the breathing activities of patients in disaster situations. BREMON involve the following phases: there is a sensing phase, where a paramedic situates a smartphone on the patient and with the accelerometer measure the “contractions and relaxations of the diaphragm muscle”. The second phase, which they called “computing”, it is just translate the raw data from the accelerometer to Breathes Per Minute (BPM) and the third phase is “communications”, where the data is sent to all of the paramedic’s smartphone using multi-hop Bluetooth network. One aspect to note in this work is the use of an architecture called SPontaneous Information and Resource sharing InfrasTructure (SPIRIT), which is the responsible of network connectivity, that is, finding and maintaining connection with other nodes and sensors sharing, that is, share the information collected by the sensor of the smartphone. They implemented the design on Android Nexus One phones, using the Android Bluetooth API for connecting them in ad-hoc mode and evaluated both, the paramedic device and the patient device. For the one-hop case, it is reported for the service-seeking client a Service Discovery Latency and Service Access Latency of 2423 ms and 1576 ms, respectively and until 7434 ms and 5510 ms for the three-hop case, while for the service provider, the BREMON app consumes a maximum of 16% of CPU usage and 29020K of memory usage. The main difference between their work and ours is the technology used for the ad-hoc network but in any case, it gives some references in latency for emergency response system.

In [15] S. Feese et al. presented CoenoFire, another smartphone based sensing system aimed at capturing performance indicators of firefighting missions (e.g. timing measures, team effort, team coordination, etc.) in order to provide meaningful data that could be used in fireman trainings or “post-incident” analysis. CoenoFire comprises sensing and visualization modules. For the sensing module, they used an Android phone, specifically a Sony Xperia Active Smart because of “its robust design”, where it was installed an app that collected the data from sensors such as accelerometer, barometer, GPS, among others and saved the raw data to the memory card. On the visualization module, they run two webservers, one that received and saved the data from the phones and other that provided a real-time user interface to monitor the firemen team. It is important to mention that there is no
communication between the phones (i.e. it is not an ad-hoc network) and that the features are transmitted directly from the phones to the server via the mobile network. They deployed the app in a professional fire brigade over a period of six weeks in which 71 firemen used the system in 76 real-world mission totaling to over 148 hours of data. Finally, they concluded that with the analyzed metrics cannot determine the whole performance of the fireman during the mission but they pointed out that the information extracted is “valuable for incident commanders and training instructors”.
Chapter 3

Network Calculus

Network calculus is a theory for performance guarantee analysis of communication networks. It is based on using alternate algebras, particularly the min-plus and max-plus algebra, to transform complex non-linear network systems into analytically tractable linear systems [16].

Network calculus can be seen as the system theory that applies to computer networks. Figure 3.1 depicts the network calculus representation of a network node. Applying the previous analogy, the input function of system theory can be seen as the arrival process of a traffic source, the transfer function as the service offered for the network and the output function is the departure process of the network.

![Network Calculus representation](image)

3.1 Min-Plus Algebra Basics

In min-plus algebra, the addition and multiplication of conventional algebra becomes computation of the infimum and addition, respectively. We will not cover all of the properties of this “new” algebra but, continuing with the analogy of system theory, we will present the definition of the min-plus convolution and deconvolution because these two operations will allow us further on, to concatenate server in series and to compute service guarantees.

Min-plus convolution of real-valued, bivariate functions \( x(s, t) \) and \( y(s, t) \) and a corresponding min-plus deconvolution are defined for \( t \geq s \geq 0 \) as

\[
(x \otimes y)(s, t) = \inf_{\tau \in [s, t]} [x(s, \tau) + y(\tau, t)]
\]
where inf is the infimum and sup is the supremum.

In the following sections, we will apply this algebra to data flows and network elements, using the concepts of arrival, server and departures processes.

### 3.2 Quality of Service Guarantees

Quality of Service (QoS) refers to the nature of the packet delivery service provided by the network and is the collective effect of service performance indicating the degree of satisfaction of a user of the service.

A service guarantee is either deterministic or stochastic. A deterministic service guarantee guarantees that all packets of a flow meet its required performance metric such as throughput, loss and delay bounds, whereas a stochastic service guarantee allows the QoS objectives specified by a flow to be guaranteed with a probability smaller than one, that is, they can be violated.

In wireless networks, the capacity of the wireless channel is time varying in a random manner due to the impact of propagation phenomena, contention and others factors. Additionally, in multi-access networks such as CSMA networks, the server capacity seen by a user is dependent on the traffic characteristics of other users [16]. Because of that, we will focus on stochastic network calculus for the analysis and provision of service guarantees in wireless networks.

### 3.3 Stochastic Network Calculus

Arrival and departures process are defined as $A(0,t)$ and $D(0,t)$, respectively and represent the amount of data detected in the interval $(0,t]$ with $t \in \mathbb{N} = \{0,1,...\}$. These functions are real-valued cumulative functions, therefore, are nonnegative and increasing in $t$. In addition, we can represent the amount of data detected in the interval $(s,t]$ by the bivariate functions $A(s,t) = A(0,t) - A(0,s)$ and $D(s,t) = D(0,t) - D(0,s)$.

From [17], we use the following definition: Assume $A(0,t)$ and $D(0,t)$ are the arrival and departure process of a lossless server, respectively. Let $S(s,t)$ for $t \geq s \geq 0$ be a random process that is nonnegative and increasing in $t$. The server is called a dynamic server $S(s,t)$ if for any fixed sample path it holds for all $s \geq 0$ that

$$D(0,t) \geq (A \otimes S)(0,t)$$

### 3.4 Moment Generating Function

In [18] a network calculus with moment generating function (MGF) is developed using the concept of dynamic servers. Throughout this work we assume that arrival and service are described by statistically independent, stationary random process. Under this assumption $A(s,s+t)$ equals $A(0,t)$ in distribution for all $s \geq 0$. The MGF of a stationary random process $A$ is defined as

$$M_A(\theta, t) = E e^{\theta A(0,t)}$$

\[ (x \ominus y)(s,t) = \sup_{\tau \in [0,s]} [x(\tau, t) - y(\tau, s)] \]
where $E$ is the expected value of a random variable.

We use the same notation as [18] to define $\bar{M}_S(\theta, t) = M_S(-\theta, t)$. MGFs have some interesting properties. For additive and multiplicative constants $a$ and $b$ it is known for all $\theta$ that

$$M_{a+bA}(\theta, t) = e^{\theta a} M_A(b\theta, t)$$

and for the addition of two statistically independent random processes $A$ and $B$ it holds that

$$M_{A+B}(\theta, t) = M_A(\theta, t) M_B(\theta, t)$$

**Theorem 1 (Concatenation):** Consider $S_1(s, t)$ and $S_2(s, t)$ dynamic servers in series as depicted in Figure 3.2. There exist an equivalent, single dynamic server $S(s, t)$ for $t \geq s \geq 0$ where

$$S(s, t) = (S_1 \otimes S_2)(s, t)$$

Assuming as [18] that $S_1(s, t)$ and $S_2(s, t)$ are statistically independent, stationary and have MGF $\bar{M}_{S_1}(\theta, t)$ and $\bar{M}_{S_2}(\theta, t)$ respectively. The MGF of the equivalent server is upper bounded for $t \geq 0$ and all $\theta$ according to

$$\bar{M}_S(\theta, t) \leq \sum_{\tau=0}^{t} \bar{M}_{S_1}(\theta, \tau) \bar{M}_{S_2}(\theta, t - \tau)$$

(1)

**Backlog and delay:** Let $A(0, t)$ and $D(0, t)$ be the arrival and departures process of a lossless server, respectively. The backlog at time $t \geq 0$ is

$$b(t) = A(0, t) - D(0, t)$$

Assuming first-come first-served ordering the delay at time $t \geq 0$ is upper bounded according to

$$d(t) = \inf [s \geq 0: A(0, t) - D(0, t + s)] \leq 0$$

**Theorem 2:** Consider a dynamic server $S(s, t)$ with arrival process $A(s, t)$. The backlog at time $t \geq 0$ is upper bounded according to

$$b(t) \leq (A \ominus S)(t, t)$$

Assuming first-come first-served ordering the delay at time $t \geq 0$ is upper bounded according to

$$d(t) \leq \inf [s \geq 0: (A \ominus S)(t + s, t)] \leq 0$$
The departure process is upper bounded for any $t \geq s \geq 0$ according to
\[ D(s, t) \leq (A \circ S)(s, t) \]

Performance bounds [19] follow with Chernoff’s theorem. Consider a dynamic server $S(t)$ with arrival process $A(t)$, which are statistically independent, stationary and have MGF $M_A(\theta, t)$ and $M_S(\theta, t)$, respectively. Assuming first-come first-served ordering an upper delay bound that are violated at most with probability $\epsilon \in (0,1]$ are
\[
d = \inf_{\theta > 0} \left[ \inf \left[ \tau: \frac{1}{\theta} \log \sum_{s=\tau}^{\infty} M_A(\theta, s - \tau) M_S(\theta, s) - \log \epsilon \right] \leq 0 \right] \tag{2}
\]

Given this framework, obtaining the delay bounds for a specific network topology boils down to selecting the violation probability, modeling the arrival and service process through corresponding MGFs and searching the values of $\theta$ and $\tau$ for which the previous equation holds.

### 3.5 Modeling the Fading Channel

The use of network calculus and the derivation of quality of service guarantees for wireless systems has not been an easy problem to address because of the modeling of the wireless medium. In order to apply the network calculus theory, it is crucial to present probabilistic service curves that describe fading channels.

One of the traditional techniques for modeling fading channel consists in the use of Markov chain. Markov chain permits to consider memory between consecutive transmissions, which is a key factor for high-layer models, that is, models that do not the channel representing the received signal strength or related process at the physical layer. One of these models is the classical Gilbert-Elliott model [20] [21], which is based on a discrete time 2-state Markov chain, in which the data can be decoded error-free if the channel is in “on state”, or cannot decoded correctly at the receiver if the channel is in “off state”.

![Figure 3.3 Gilbert-Elliot model](image)

The Gilbert-Elliot model of Figure 3.3 is used to describe a random process $S(s, t)$ that is defined as the service offered by a fading channel in the interval $(s, t]$. Assuming that the service is non-idling and uses the entire available service to serve backlogged data (i.e. work-conserving server), it can be seen as dynamic server.

From [18], we used the following definition: Consider an irreducible, homogeneous Markov chain with $n$ states and stationary distribution $\pi$. Workload is process with rate $h$ when in
state \( i \). Let \( H(\theta) \) be the diagonal matrix \( \text{diag}(e^{\theta h_1}, e^{\theta h_2}, \ldots, e^{\theta h_N}) \) and \( Q \) be the transition probability matrix, where \( q_{ij} \) is the transition probability from state \( i \) to state \( j \) and \( \mathbf{1} \) is a column vector of ones. For all \( t \geq 0 \) and all \( \theta \) the MGF of the corresponding random process \( S \) is

\[
\bar{M}_S(\theta, t) = \pi(H(-\theta)Q)^{t-1}H(-\theta)\mathbf{1}^3
\]  

(3)

At this point, we have a characterization of the fading channel with a packet-base model that depends of the transition probabilities between the states and the rate at which the channel processes the workload. The expression (3) will be used on (2) to determine delay bounds.

### 3.5.1 Coherence Time and Doppler Spread

Time varying nature of the channel is caused by either relative motion between the receiver and transmitter, or by movement of objects in or around the channel. Doppler spread and coherence time are parameters which describe the time varying nature of the channel in a small-scale.

Doppler spread \( B_D \) is a measure of the spectral broadening caused by the time rate of change of the mobile radio channel and is defined as the range of frequencies over which the received Doppler spectrum is essentially non-zero. When a pure sinusoidal tone of frequency \( f_c \) is transmitted, the received signal spectrum, called the Doppler spectrum, will have components in the range \( f_c - f_d \) to \( f_c + f_d \), where \( f_d \) is the Doppler shift. The amount of spectral broadening depends on \( f_d \) which is a function of the relative velocity angle of the movements. If the baseband signal bandwidth is much greater than \( B_D \) the effects of Doppler spread are negligible at the receiver. This is a slow fading channel [22].

Coherence time \( T_C \) is the time domain dual of Doppler spread and is used to characterize the time varying nature of the frequency dispersiveness of the channel in the time domain. A popular rule of thumb for modern digital communications is to define coherence time as [22]

\[
T_C \approx \frac{0.423}{f_m}
\]

where \( f_m \) is the maximum Doppler frequency shift. At the same time, \( f_m \) is given by

\[
f_m = \frac{v}{\lambda}
\]

where \( v \) is the velocity and \( \lambda \) is the wavelength.

Later on, we will model the wireless channel as a slow fading channel and we will use these definitions to support our analysis.

### 3.6 Scheduling Discipline and Leftover Service Characterization

The analysis of performance bounds to multiplexed flow is also possible inside the network calculus framework. However, as we want to analyze the end-to-end delay per-flow, we will consider a single system with stationary service curve \( S(t) \), stationary through-traffic arrival \( A^t(t) \), stationary cross-traffic arrivals \( A^c(t) \) and corresponding departures \( D^t(t) \) and \( D^c(t) \), respectively as shown in Figure 3.4.

---

3 Respective proof can be found in Chang 2000
To analyze the delay of a through-flow is important to isolate the remaining service that is left over by cross-traffic at each of the systems, that is, the service curve that actually would see the through-flow (Figure 3.5). Then, we can use the concatenation theorem with the so-called leftover service curve of each sub-system to obtain the equivalent network service curve for the through-traffic flow [23]. The left over service is strongly related with the scheduling model that is implemented in the network. Following are the expressions of left over service using the General Scheduling Model.

**General Scheduling Model:** Suppose flows with stationary arrival process $A_1(t)$ and $A_2(t)$, which are scheduled at a stationary work-conserving server with service process $S(t)$. $A_2(t)$ sees a dynamic server [19]

$$S_2(t) = \max[0, S(t) - A_1(t)]$$

Assuming $A_1(t)$ and $S(t)$ are statistically independent and have MGF $M_{A_1}(\theta, t)$ and $\bar{M}_S(\theta, t)$, respectively. The MGF of the server seen by $A_2(t)$ is upper bounded for $\theta \geq 0$ by

$$\bar{M}_{S_2}(\theta, t) \leq \min\left[1, \bar{M}_S(\theta, t)M_{A_1}(\theta, t)\right] \tag{4}$$

It is important to note that the general scheduling model “does not make any assumptions about the order in which flows are served with respect to each other”, in other words, as we do not have any other information of the system, the bound that we can find is actually for a preemptive priority scheduler where flow $A_2(t)$ has low priority.
Chapter 4

Experimental Evaluation

In this chapter we will first present the methodology, implementation details and results for the experimental performance evaluation of an IEEE 802.11 ad-hoc network on the Raspberry Pi.

To analyze this performance, we will use the following metrics:

- **Current consumption**: Network nodes have limited energy resources because they are powered by batteries. A limited energy budget constrains the computation and communication capacity of each node. Thus, it is beneficial to quantify the current consumption to be able to determine the autonomy of the system and possible techniques for optimization.

- **Communication range**: communication among nodes is limited to a certain range due to the loss of power in the transmitted signal in a wireless channel. Therefore, it is important to investigate this limit in the ad-hoc network.

- **Connectivity**: given that the nodes are free to move, the topology will change along the time. Knowing properties of the link’s temporal behavior in MANETs, that is, statics of time that the nodes in the network are (dis)connected, it will allow us to prepare the network to possible failures.

- **Throughput**: In the context of communication networks, throughput is the rate of successful message delivery over a communication channel and it is usually measured in bits per second (bps). These messages may be delivered through a single or multiple links.

For carry on the tests, we set up an ad-hoc network on the RP using a WiFi-USB adapter and then we implemented specific experiments to obtain the metrics. Following is the description of the methodology that we used.
4.1 Methodology

4.1.1 Ad-Hoc Network

The RPi does not have a built-in wireless card as a consequence we had to use a WiFi-USB adapter in one of the USB available ports in order to provide wireless capabilities. There are several WiFi-USB adapters in the market that differ in factor form (size), power transmission and chipset/driver. This last aspect has main importance because the deployment of the ad-hoc depends on the driver module installed in the operating system (OS) of the RPi.

There are two manners to put the RPi in ad-hoc mode: manual and Debian method. In the manual method, it is used the ifconfig command and the Wireless tools for Linux. This process consists on setting the wireless channel, assigning an Extended Service Set Identifier (ESSID) to the network, changing the operating mode of the device, setting an IP address and (optional) setting a Wired Equivalent Privacy (WEP) [24].

However, every time the RPi is booted, it is necessary repeat the same process. The Debian method solves this problem, that is, it makes the wireless interface configuration permanent by editing the `/etc/network/interfaces` file [24].

There are some IEEE 802.11 properties at the moment of the ad-hoc deployment that we have to define:

- **SSID**: SSID stands for Service Set ID and it is a 32-character (maximum) alphanumeric key identifying the name of the WLAN.

- **Cell ID**: Also called Basic Service Set ID (BSSID), it is the identifier of the access points within a WLAN. In infrastructure mode, the MAC address of the AP is used as the BSSID but in ad-hoc networks, there is no physical AP so the network generates a 48-bit string of numbers that looks and functions just like a MAC address. It is important to mention that for the successful connection between nodes in the ad-hoc network, all of the nodes must have the same cell ID.

- **Frequency**: It is the channel selected in the 2.4 GHz range for the deployment of the ad-hoc network.

Resuming, the steps that we used to set up the ad-hoc network on the RPi were:

- Install Raspbian [25], which is a free operating system based on Debian optimized for the RPi hardware, it is considered the “default” OS for RPi and it comes with updated drivers that support several wireless adapters.

- Set up the ad-hoc via the Debian method and verify its properties namely Cell ID.

- Test the ad-hoc capability of different WiFi-USB adapters (i.e. different chipset/drivers) using the ping command.

4.1.2 Current Consumption

There are two basics methods for measuring current. The first method is called magnetic current sensing and consists in measuring the magnetic field around a current-carrying...
conductor. The magnetic sensors present high-sensitivity, small size and do not introduce loss to the circuit; however, it is relatively expensive and also susceptible to temperature-coefficient error and effects of nonlinearity [26].

The second method is called resistive sensing and consists of the addition of a small resistor in series with the power supply and the load. The voltage that drops in the sense resistor is later amplified and the current is computed with the Ohm’s law. The resistive-sensing technique is cost-effective and provides accurate results for the most of applications [27].

Resistive-sensing can be implemented in two ways, depending on the placement of the low-valued resistor. In low-side current sensing, the sense resistor is placed between the ground (0V) and the load (Figure 4.1-left-side), while in high-side current sensing, it is placed between the load and the supply (Figure 4.1-right side).

![Figure 4.1 Resistive-sensing](image)

We used in this work the high-side current sensing method via the Adafruit INA219 Current Sensor Breakout Board [28]. This breakout board contains the INA219B chip, which is a “high-side current shunt and power monitor” [29] and a sense resistor of 0.1 Ω. To be able to read the current and bus voltage register of the INA219B, we have to communicate with the chip via an Inter-Integrated Circuit (I2C) interface.

Figure 4.2 shows the circuit used to measure the current drawn by the Raspberry Pi, where one RPi is the load and another is running a Python program to read the registers via I2C. The common-mode voltage in the INA219B was around 5V (power supply), less than the 26V of maximum common-mode voltage reported in the datasheet. Since the maximum input difference is ±320mV, the resistor was 0.1Ω and the Analog to Digital Converter (ADC) was configured to 12-bit, the circuit could measure up to ±3.2A with a resolution 0.8mA

---

4.1.3 Communication Range

To determine the communication range of the nodes, we used the concepts of packet loss and Received Signal Strength Indicator (RSSI). The packet loss is the fraction of not received packets from a total number of transmitted packets and the RSSI is “an expression of the signal-to-interference-and-noise ratio (SINR)” [30], therefore, the higher the RSSI, the stronger the signal.

There is no standardized relationship of any particular physical parameter to the RSSI reading [31]. The 802.11 standard does not define any relationship between RSSI value and power level in mW or dBm. Vendors and chipset makers provide their own accuracy, granularity, and range for the actual power (measured as mW or dBm) and their range of RSSI values.

Nevertheless, all the chipsets present a RSSI reading, some sort of “threshold” value that can be related with high packet loss in the link and therefore, with the communication range. The procedure to determine this threshold was:

- Put two nodes (RPi\textsubscript{1} and RPi\textsubscript{2}) at a known distance $d$.
- Start an User Datagram Protocol (UDP) server in the RPi\textsubscript{1}.
- Start an UDP client in the RPi\textsubscript{2} sending data for a known rate and time.
- Report the packet lost in RPi\textsubscript{1}.
- Read the RSSI in RPi\textsubscript{1}.
- Repeat the steps with at a different distance $d’$. 
The initial distance between the RPis was set to 5m and then, it was incremented 5m for each next reading until we got a 100% packet lost. We considered that this granularity was appropriated for this work. Additionally, it is worth to say that all the measures were performed 10 times.

### 4.1.4 Connectivity

Connectivity is the capacity of nodes to communicate because they are in each other’s communication range or via multi-hop in MANETs. We analyzed the connectivity of the MANET through the temporal behavior of links, that is, the distribution of time of connection and disconnection between any nodes.

This analysis was performed in a dataset proportioned by Vital Responder 2.0 Project partners of GPS traces extracted from a firemen training session in the locality of Baião, Porto, Portugal. The traces were generated by Android phones placed in each fireman and a generic GPS device placed in the truck. A total of 5 devices (4 fireman, 1 truck) reported 90814 positions, where a position is reported by each device every 1 second.

The distribution of time of connection between firemen and truck (i.e. the sink) was computed namely through the transitive closure, which is an algorithm applied to directed graphs that finds out if vertex j (node in our case) is reachable from another vertex i for all vertex pairs (i,j) \[32\] as shown in Figure 4.4.

![Figure 4.4 Transitive closure](image)

The method to analyze the connectivity is presented following:

* Interpolate: in general the GPS devices take a sample of the position coordinates each second, however, when the GPS receiver does not see enough satellites, no position can be estimated, leading to a second without position sample. To cope with this, we decided to interpolate in a linear fashion until a maximum number of samples.

* Compute distance: we computed the distance between each pair of nodes with the haversine formula \[33\]. This formula is important in navigation and gives us the shortest distance between two points on the surface of a sphere from their longitudes and latitudes. For a point \(x_1\) with \((\varphi_1, \beta_1)\) pair of latitude and longitude coordinates, respectively and a point \(x_2\) with \((\varphi_2, \beta_2)\), the distance \(d\) using haversine formula is given by

---

\[5\] Skiena 2009
where $R$ is the radius of the Earth with a value of 6371 $Km$.

* Connection matrix: using a specific radio communication range, we built a binary matrix called the connection matrix, where there is 1 if the nodes are able to communicate in a direct link, 0 if not and an optional $-1$ if we do not have the data for that time sample.

* Nodes and sink connection: for one-hop connection between firemen, we look at the connection matrix for all direct links between any pair of nodes, whereas for the sink connection, we used the transitivity closure algorithm.

* Statistics properties: we reported the connectivity to any other node and the connectivity to the sink via the mean, variance and the distribution for the time of connection.

The whole method was implemented in a MATLAB script that reads the GPS traces as input and outputs the mentioned statistical properties.

### 4.1.5 Throughput

Throughput analysis in IEEE 802.11 ad-hoc network is often done using simulations [34] but in this work, we performed an experimental evaluation where we considered multiple flows on the ad-hoc network. In addition, because the transport protocol influences the throughput when multiple flows are present, we also considered different transport protocols namely UDP and Transmission Control Protocol (TCP).

As we wanted to see the dimension of the impact of multiple flows sharing the available channel in a multihop scenario, where intermediate hops also generate traffic, we performed the following scenarios:

**Scenario 1:** This scenario aims to study the throughput of one wireless link. To do so, one netbook ASUS Eee PC 1015PEM will play the role of the sink and one RPi will send data towards it at different data rates (Figure 4.5). Both devices will be static at a fixed distance with line of sight (LOS) between them. The distance was chosen such that the RSSI reading was sufficient to have a good quality link.

**Scenario 2:** This scenario is used as objective study the throughput in a multi-hop environment. One netbook ASUS Eee PC 1015PEM will play the role of the sink and one RPi will send data towards it at different data rates. However, there will not be a direct link between

$$a = \sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos(\varphi_1) \cdot \cos(\varphi_2) \cdot \sin^2\left(\frac{\beta_2 - \beta_1}{2}\right)$$

$$d = 2R \cdot \arctan(\sqrt{a}, \sqrt{1 - a})$$
them and the data has to be forwarded by another RPi (Figure 4.6). The three devices will be static at a fix distance (same criterion as scenario 1) and they will be running OLSRD [35], a daemon implementation of the OLSR protocol to compute the route between nodes.

![Figure 4.6 Two hop-One client](image)

**Scenario 3:** This scenario aims to study the achievable throughput in each node, when all nodes generate traffic. It is similar to the previous one, the only difference in that the RPi\(_1\) is not just forwarding data, but also generating it (Figure 4.7). This scenario emulates the first real implementation of the Vital Responder Project with a team of two firemen.

![Figure 4.7 Two hop-Two clients](image)

The procedure to determine the throughput in all the scenarios was:

- Put the nodes at a known distance.
- Start a server in the netbook.
- In the scenario 1:
  - Start an UDP client in the RPi\(_1\) for a known data rate and time.
  - Report the throughput of RPi\(_1\).
- In the scenario 2:
  - Start an UDP client in the RPi\(_2\) for a known data rate and time.
  - Report the throughput of RPi\(_2\).
- In the scenario 3:
  - Start an UDP client in the RPi\(_1\) and RPi\(_2\) for a known data rate and time.
  - Report the throughput of RPi\(_1\) and RPi\(_2\).
- Repeat the steps with a different data rate.
- Repeat the steps with TCP protocol.
Given that the project establish that the network of firemen is composed by 5 nodes, we wanted to perform more scenarios, however, it was not possible due to the lack of manpower. It is also worth to say that all the measures were performed 10 times.

4.2 Implementation and Results

4.2.1 Ad-Hoc Network

We installed Raspbian OS version from 2015-02-16 (available at [9]) according to details in [36]. Then we set up the ad-hoc network in two RPIs via the Debian method as explained in section 4.1.1. Figure 4.8 shows the properties of the network built.

![Figure 4.8 Ad-Hoc properties](image)

With the RPi configured in ad-hoc mode, we tested various WiFi-USB adapters to prove the ad-hoc capability by pinging between RPIs. Table 4.1 shows the adapters, their driver and chipset and whether or not was possible to implement an ad-hoc network.

<table>
<thead>
<tr>
<th>Adapter</th>
<th>Chipset</th>
<th>Driver</th>
<th>Ad-Hoc mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Fi</td>
<td>Ralink RT5370</td>
<td>rt2800usb</td>
<td>Yes</td>
</tr>
<tr>
<td>N150</td>
<td>Ralink RT5370</td>
<td>rt2800usb</td>
<td>Yes</td>
</tr>
<tr>
<td>TP-LINK TL-WN722N</td>
<td>Atheros AR9271</td>
<td>ath9k_htc</td>
<td>Yes</td>
</tr>
<tr>
<td>Asus USB N10</td>
<td>Realtek RTL8188SU</td>
<td>r8712u</td>
<td>No</td>
</tr>
<tr>
<td>Asus USB N10 nano</td>
<td>Realtek</td>
<td>rtl8192cu</td>
<td>No</td>
</tr>
<tr>
<td>Asus USB N13</td>
<td>Realtek RTL8192CU</td>
<td>rtl8192cu</td>
<td>No</td>
</tr>
<tr>
<td>N150 Realtek</td>
<td>Realtek RTL8188CUS</td>
<td>rtl8192cu</td>
<td>No</td>
</tr>
<tr>
<td>IOGEAR</td>
<td>Realtek RTL8191SU</td>
<td>r8712u</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 4.1 Chipset/Driver in ad-hoc network*

From the table we can see that the ad-hoc mode works in chipsets Ralink RT5370 and Atheros AR9271 on RPI with Raspbian version from 2015-02-16. The drivers rt2800usb and ath9k_htc are installed in the Kernel and they did not need any modifications.

On the other hand, all the chipsets Realtek did not work on ad-hoc mode. Realtek chipsets and drivers do not support ad-hoc mode. Still, we tried compiling different versions of the drivers, but we always encountered at least one of the following issues:
• Failure packets: The wireless interface on each RPi appeared as connected to the ad-hoc network but when you send a ping command from one node, there is no reply from the other. We could determine that the problem was that the RPi could not handle the broadcast message of the ARP protocol to find the MAC address of an unknown IP address. This is a sign of driver issue and it has been reported in the forum of the RPi official website in [37], [38].

• Different cell ID: The RPis show different cell IDs, therefore, there is no communication between them. This behavior was presented in two situations:

  o Boot the RPis inside each other’s range and each of them takes from boot on a different cell ID and it is impossible to change it.

  o Boot the RPis inside each other’s range, the RPis take the same cell ID so there is connection. Move away the RPis (i.e. out of each other’s range), each of them takes a different cell id (normal behavior) but when they are brought back together (i.e. into each other’s range), they remain with different cell IDs (unexpected behavior). In other words, when the RPis are connected and get disconnected due to the distance, they are unable to re-connect.

We can conclude that the combination Ralink RT5370/rt2800usb and Atheros AR9271/ath9k_htc work in ad-hoc mode on the RPi with Raspbian version from 2015-02-16, while the chipsets Realtek tested (i.e. RTL8188SU, RTL8192CU, RTL8188CUS and RTL8191SU) do not work in ad-hoc mode. Figure 4.9 shows the adapter with ad-hoc capability, that is, the Wi-Pi, N150 Ralink and TP-Link adapter.

![Figure 4.9 WiFi-USB adapter (left:N150 Ralink, middle: Wi-Pi, right: TP-Link)](image)

4.2.2 Current Consumption

Test the sensor in a controlled environment

To obtain the measures from the breakout board INA219, we wrote a program in Python which uses the INA219 library [39] to read the values measured from the bus voltage and current register of the chip. This breakout board was originally designed for Arduino microcontrollers but some efforts have been made to change the register address on the library to be able to use it in the RPi [40].

To prove the correct working of the program and the circuitry of the INA219, we designed the test circuit of Figure 4.10, where using the Ohm’s law, we could know beforehand the value of the current, that is, \( i = \frac{V}{R} \).
As it was expected the breakout board reported a current of 50mA for the test circuit. Once knowing that our set up was working, we moved on to measure the current of the RPi.

**Connect the INA219 to the RPi (load)**

We assembled the circuit of Figure 4.2 described in section 4.1.2 and measured the current drawn by the RPi with no WiFi-USB adapter, to serve as reference for other scenarios. Then, we measured the current drawn by the Wi-Pi, N150 Ralink and TP-LINK TL-WN722N in ad-hoc mode.

The set up for this experiment was:

1. At $t = 0$, run the measuring program with the RPi (load) on “steady state”, i.e. turned on a long time ago.

2. At $t = 30$, restart the RPi (load).

The evolution of the current versus time is shown in Figure 4.11

![Figure 4.11 Current drawn on the RPi](image)

We can see the average current draw for the RPi itself is 191mA and that the Wi-Pi, N150 Ralink and TP-Link in ad-hoc mode cause an increase in the average current consumption to 286mA, 296mA and 354mA, respectively. Other aspects to note are the current peaks, which reach 570mA in the case of the TP-Link adapter. Given the results of this experiment, we can conclude that the addition of the Wi-Pi, N150 Ralink and TP-Link in ad-hoc mode causes an increase of around 50%, 55% and 85% respectively, in the base current drawn by the RPi.
Run the program for data scenarios

This part aimed to verify the impact in the current consumption while the RPi was sending data through the wireless interface, to do so, we used the software Iperf 2.0.5.

Iperf [41] is a network testing tool that can create UDP and TCP data streams and measure the throughput, jitter and delay of a network. Iperf uses client and server to measure the throughput between the two ends, either unidirectionally or bi-directionally and additionally, when UDP is selected, Iperf reports the packet loss.

In each of the following scenarios, we used Iperf in UDP mode, transmitting for 60 seconds with a payload of 1470 bytes:

- Send data at 100kbps

The set up for this experiment was:

1. At $t = 0$, run the measuring program with the RPi in ad-hoc mode on “steady state”.
2. At $t = 30$, start the Iperf client with data rate set at 100kbps.

The evolution of the current versus time is shown in Figure 4.12.

![Figure 4.12 Current evolution at 100 kbps](current_evolution.png)

The Wi-Pi, N150 Ralink and TP-Link in ad-hoc mode sending at 100kbps average current consumption are 289 mA, 299 mA and 362 mA, respectively. We can see that there are peaks of current at the beginning of the transmission in all the adapters and also, how the peaks in the TP-Link are more constant during the transmission time, reaching values of 585 mA.

- Send data packets at higher rates

We repeated the same scenario as before but with higher data rates. In Figure 4.13 is the evolution of the current versus time for 1Mbps.
The Wi-Pi, N150 Ralink and TP-Link in ad-hoc mode sending at 1Mbps average current consumption are 299mA, 310mA and 369mA, respectively. We can see similar behavior to the previous scenario, with peaks of current at the beginning of the transmission in all the adapters, but this time there are more peaks in the transmission time. As we expected, the higher the data rate, the higher additional current.

We carried out additional experiments with data rates at 500kbps, 5Mbps and 10Mbps, whose results are shown in Annex A. Table 4.2 shows the current consumption for the RPi without adapter and with adapters in ad-hoc mode, while Table 4.3 shows the current consumption for all the data rates tested.

<table>
<thead>
<tr>
<th>Adapter</th>
<th>Current (mA)</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>191</td>
<td>Minimum current drawn by the RPi</td>
</tr>
<tr>
<td>Wi-Pi</td>
<td>286</td>
<td>Increase the consumption 95mA</td>
</tr>
<tr>
<td>N150 Ralink</td>
<td>296</td>
<td>Increase the consumption 105mA</td>
</tr>
<tr>
<td>TP-Link</td>
<td>354</td>
<td>Increase the consumption 163mA</td>
</tr>
</tbody>
</table>

Table 4.2 RPi and ad-hoc consumption

<table>
<thead>
<tr>
<th>Adapter</th>
<th>100kbps</th>
<th>500kbps</th>
<th>1Mbps</th>
<th>5Mbps</th>
<th>10Mbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Pi</td>
<td>289 mA</td>
<td>292 mA</td>
<td>299 mA</td>
<td>335 mA</td>
<td>375 mA</td>
</tr>
<tr>
<td>N150 Ralink</td>
<td>299 mA</td>
<td>304 mA</td>
<td>310 mA</td>
<td>337 mA</td>
<td>371 mA</td>
</tr>
<tr>
<td>TP-Link</td>
<td>362 mA</td>
<td>362 mA</td>
<td>369 mA</td>
<td>387 mA</td>
<td>399 mA</td>
</tr>
</tbody>
</table>

Table 4.3 Consumption at different data rates

We can conclude that the current consumption of the RPi is highly dependent on the WiFi-USB adapter and the data rate used with values reported between 286-399mA but in any case, the percentage of hardware consumption (i.e. the RPi plus the adapter) represent 76%,
80% and 89% in the Wi-Pi, N150 Ralink and TP-Link, respectively, which according to Amdahl’s law [42], it is the maximum that might be improved with changes to the networking protocol.

4.2.3 Communication Range

The communication range is affected by propagation phenomena. Thus, it was important to choose a setting for experimentally assessing it that would resemble the envisioned application environment, i.e. a forest. We opted for Parque da Cidade do Porto with its grassy areas, rocks, soil and trees was the environment chosen to perform these new sets of experiments.

The experiment to determine the communication range of each WiFi adapter was implemented according to section 4.1.3. To compute the packet loss, we used lperf in UDP mode, transmitting for 10 seconds with a payload of 1470 bytes at 1Mbps and to take the RSSI readings of the wireless link, we used the iwlist command.

Figure 4.14 shows the results for the N150 Ralink adapter. We can see that we have a packet loss of lower than 5% when the RSSI reading is higher than −80 dBm, which corresponds to distances lower than 10 m between the RPi₁ and RPi₂. In the range 10 – 20 m, the RSSI reading is slightly below than −80 dBm, which makes the packet loss highly variable in this region. For distances larger than 25 m, the packet loss reaches values higher than 90% while the signal strength decreases to values smaller than −81 dBm.

We will consider the communication range the distance at which the RSSI reading is below the “threshold” and the packet loss is continuously increasing, e.g., in the N150 Ralink adapter case the communication range is 30 m.
We repeated the same experiment with the Wi-Pi and TP-Link adapters. The results are presented in Figure 4.15 and Figure 4.16. The Wi-Pi adapter shows a similar performance that the N150 Ralink adapter. The first observation is that different adapters reported different values of RSSI, for example, the N150 Ralink reported an average of $-80$ dBm for 10 m, while the Wi-Pi reported $-71.2$ dBm and the differences are higher for higher distances.

In the Wi-Pi case, we have a packet loss lower than 38% when the RSSI reading is higher than $-72$ dBm, which corresponds to distances lower than 10 m. The unstable region comprises distances between 10 – 20 m, corresponding to the $-72, -74$ dBm RSSI interval. For distances larger than 20 m, the RSSI is smaller than $-74$ dBm and the connection is full of losses. Therefore, the communication range is around 20 m.

![Figure 4.15 Packet loss and RSSI - WiPi Adapter](image)

The TP-Link adapter shows a very different behavior, comparing it with the previous two adapters. The packet loss reported until the maximum distance that we could performed the experiment, i.e. 150 m, was 0%. The RSSI reading at 5 m was $-43.4$ dBm, a much larger value than the other adapters at the same distance. Another aspect to mention is that we could measure until $-85.3$ dBm without losses, which means that the reception sensitivity is beyond this value. It is clear that all these advantages are related to the use of the antenna, which gives an additional gain of 4 dBi.
We also did an additional experiment just with the Wi-Pi and N150 Ralink adapters, where we used the ping command to check the connectivity between both RPIs. In this experiment, as we ping the RPi₂ from RPi₁, the communication range is defined as the distance at which the RPi₂ does not get consecutive icmp_req. In Figure 4.17 we can see the map of all the experiments on this communication range part.

We can appreciate two characteristics with the ping experiment, first the distance between the RPi₁ and RPi₂ is the same with both adapters, which confirms once again their similarities and second, this distance is larger than the Iperf experiment. This behavior is because the IEEE 802.11 cards transmit at a constant power, so decreasing the number of
bytes or data rate allows the packaging of more energy per symbol, resulting in a larger communication range.

4.2.4 Connectivity

The connectivity analysis between nodes in the ad-hoc network allows us determining properties such as the probability of connection between nodes, the maximum distance between nodes and distribution of the time of connection. To evaluate these metrics, we built a MATLAB framework that received as input the dataset proportioned by Vital Responder 2.0 project partners. The total number of samples after the pre-processing and interpolation of the dataset was 18926.

Distance between nodes

Using the haversine formula, we computed the distance between all the firemen and the sink with GPS data available. The Figure 4.18 corresponds to the distance between a fireman and the sink.

![Figure 4.18 Distance from a fireman to the sink](image)

We can appreciate that the distance between a fireman and the sink can reach values up to 820 m, which means that either we use a communication technology that covers this kind of range (e.g. WiMAX) or multiple hop connection (e.g. IEEE 802.11) to obtain connectivity. In addition, there are two aspects to note in the figure. First, the distance to the sink in the four firemen increases constantly after the sample 12000, which means that either the truck or the four firemen moves away at the same time. Second, after the sample 15125 of the processed data, we do not dispose of any other GPS position for the sink and for the rest of the analysis, we assumed the last GPS position registered.
Figure 4.19 Minimum distance from a fireman to any other

Figure 4.19 corresponds to the minimum distance between a specific fireman and any other, that is, the distance that it would have to cover the links of a hypothetic multiple hop connection. We can appreciate that the maximum distance of the closest fireman in the processed data is around 250 m. Another aspect to note is that the only fireman with complete GPS data in all the analysis is Fireman 3.

Connection and distribution of connectivity

Establishing the communication range to 150 m, we computed the probability of connection between a fireman and any other. Results are shown in Table 4.4. In the Annex B is also illustrated the probability of connection between firemen.

<table>
<thead>
<tr>
<th>Fireman</th>
<th>Probability of connection</th>
<th>Time connected (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.97</td>
<td>18325</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.88</td>
<td>16619</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.82</td>
<td>15553</td>
</tr>
<tr>
<td>Fireman 4</td>
<td>0.92</td>
<td>17469</td>
</tr>
</tbody>
</table>

Table 4.4 Probability of connection with another firemen

With the previous results, we can appreciate that there is a high probability that at least one direct link exist for any fireman at the moment of fighting fire. However, the direct connection with another fireman does not guarantee that the data is going to arrive to the sink.

To qualify the connection towards the sink, taking into account that multiple hops connection may occur, we computed with the transitive closure algorithm the probability of connection between a fireman and the sink. The results are presented in Table 4.5.
<table>
<thead>
<tr>
<th>Fireman</th>
<th>Probability of connection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.28</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.24</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.43</td>
</tr>
<tr>
<td>Fireman 4</td>
<td>0.23</td>
</tr>
</tbody>
</table>

*Table 4.5 Probability of connection with the sink in a same second*

As we can see, the high probability of connection between firemen does not imply high probability of connection with the sink. There is low probability of connection with the sink for all the firemen, that is, there is a high probability of not get the data in the sink in the same second.

We can also analyze the distribution of the time of disconnection between firemen and sink. This value is of importance to the development of mechanisms that can handle disconnection of nodes in a network (e.g. saving data in a buffer). Because the disconnection distributions are alike, we show the results of Fireman 1 in Figure 4.20 (x-axis in log scale). In the Annex B are the results for the others.

![Figure 4.20 Distribution of time of disconnection (x-axis in log scale)](image)

We can see that the frequency of disconnection between Fireman 1 and Fireman 2 are the highest in the figure. The other distributions do not present high frequency but the time of disconnection is appreciable, reaching up to 3120 samples in the Fireman 3 subfigure.

In all the firemen distributions, the disconnections are not frequent but we do find larger times of disconnection. However, this large time is not just caused by the communication range on the script; it is also affected by the samples where we do not dispose GPS data.
4.2.5 Throughput

The throughput experiment was performed in an indoor environment due to we could control the establishment of routes between the nodes in the ad-hoc network. The traffic characteristics in the scenario 1 (i.e. one flow-one hop), scenario 2 (i.e. one flow-two hops) and scenario 3 (i.e. two flows-two hops) were the following:

- **UDP test**: payload of 1470 bytes, transmission time of 30 seconds at 1, 2, 5.5, 11 and 24 Mbps.
- **TCP test**: payload of 1448 bytes and transmission time of 30 seconds at maximum TCP data rate.

Regarding the distance, in the scenario 1 was set 7.5 meters between the netbook and the RPi1. In the scenarios 2 and 3, we had to put the nodes in the configuration of Figure 4.21 to assure that OLSRD built the routing table in each node the way we needed it, that is, RPi2 connected by one-hop to RPi1 and by two-hops to the sink.

![Figure 4.21 Multi-hop configuration](image)

The results of the three scenarios are depicted in Figure 4.22. The bar shown in scenario 3, it is the lowest throughput between the two client nodes. We decided to show this value because is the throughput that we can guarantee for all the nodes in the network.

![Figure 4.22 Per-scenario throughput](image)
In the first scenario, UDP outperforms TCP with average values of 16.95 Mbps and 14.41 Mbps. However, we can establish that both of these limits are due to hardware limitation and not network limitation. This statement is evidenced in the CPU usage for the 24 Mbps UDP and TCP test depicted in Figure 4.23. We can see that during transmission times the CPU of the RPi is completely overloaded and it is impossible to send more data.

![Figure 4.23 CPU-usage](image)

In the second scenario, UDP still shows better performance than TCP, reporting average values of 6.13 Mbps and 4.21 Mbps, respectively. We can see that adding an intermediate node decreases the value of the throughput by around 10 Mbps in both, TCP and UDP cases.

In the third scenario, UDP again presents better performance than TCP. We can see that if the intermediate node also generates data, the throughput decreases to 3.91 Mbps and 1.11 Mbps. However, it is interesting see two effects in this scenario:

- **UDP optimal offered load**: we can see from Figure 4.24 that until the 2Mbps test, the offered load in both RPis was sustainable, that is, not high packet loss for any flow. However, in the 5.5Mbps test, the flow of RPi2 experienced an increment of the packet loss. In this sense, we defined optimal offered load as the offered load at which neither flow experience high packet loss, therefore for this scenario, the UDP optimal offered load is 2Mbps.

![Figure 4.24 Throughput - Two flows](image)
TCP unfairness: Figure 4.25 shows one the running for the TCP case. As we can see, the RPi₁ (one-hop away) starts with a throughput of around 15Mbps until around $t = 4$ sec that appears the TCP flow of RPi₂ (two-hops away). The throughput of RPi₁ decreases but it is considerable higher than the throughput of RPi₂ in the remaining of the test. The average throughput is 12.44 Mbps for the RPi₁, more than ten times the 1.11 Mbps of the RPi₂. This behavior could be more related with RPi resource constraints than TCP protocol constraints.

![Figure 4.25 Throughput - TCP](image)
Chapter 5

Delay Bounds

In this chapter we will first present the methodology for the analysis of the end-to-end delay in different network topologies using Network Calculus. After that, we will show their corresponding implementation and results for the different study cases.

5.1 Methodology

To derive the service guarantees we have to define the traffic source of the nodes in the network and the properties of the channel described by the discrete-time Markov model of Figure 3.3. In this work, we will use an independent periodic traffic source to model the data that will be generated by the network device attached to the firemen. This type of source generates \( \sigma \) units of workload at times \( \{U + nT, n = 0,1,\ldots\} \) where \( T \) is the period of the source and \( U \) is the initial start time which is uniformly distributed in the interval \([0,1]\). For all \( t \geq 0 \) and \( \theta \geq 0 \) it is known that the MGF is given by [43]

\[
M_A(\theta,t) = e^{\theta \sigma \frac{t}{T}} \left( 1 + \frac{t}{T} - \frac{t}{T} \right) (e^{\theta \sigma} - 1)
\]  

(5)

Analyzing the traffic properties of the Project Vital Responder 2.0, we have that the data rate is 30 kbps and the block of sensor data consists on 3600 bytes. Assuming a physical layer data rate of 1 Mbps for the 802.11 ad-hoc network and a workload of 600 bytes, we have that the wireless channel can process this workload in 4.8 ms, which it is considered as the time slot of the discrete time model. Consequently, the rate \( h = 1 \) used for the channel model corresponds to one unit of workload served in each time slot. Simultaneously, for the traffic source, the time to transmit 3600 bytes at 30 kbps is 960 ms, which leads to a period of the source \( T = 200 \) and units of workload \( \sigma = 6 \).

Regarding the channel properties, we have a 2.412 GHz carrier, which gives us a wavelength \( \lambda = 0.1244 \) m and assuming an average speed value for a running fireman of 6.71 m/s, it will be faced a maximum Doppler shift of \( f_m = 59.94 \) Hz. The baseband signal bandwidth of 802.11 is 22 MHz and given that this value is much greater than 59.94 Hz, the channel can be seen as a slow fading channel.
The channel model is specified by the transition rate from OFF to ON state $p$ and the steady state error rate $q / (p + q)$, where for a given $p$, the transition rate from ON to OFF $q$ can be obtained. As discussed in [44], we modeled the slowly fading channel choosing $p = 0.1$ and $q / (p + q) = 0.1$. Table 5.1 shows the complete set of parameters used for the source and channel models.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q / (p + q)$</td>
<td>Steady state block error probability</td>
<td>0.1</td>
</tr>
<tr>
<td>$p$</td>
<td>Transition probability from OFF to ON</td>
<td>0.1</td>
</tr>
<tr>
<td>$h$</td>
<td>Channel rate</td>
<td>1</td>
</tr>
<tr>
<td>$T$</td>
<td>Period of the source</td>
<td>200</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Burst size</td>
<td>6</td>
</tr>
</tbody>
</table>

*Table 5.1 Source and channel model parameters*

In the first scenario, we examined the delay of a single wireless link (Figure 5.1) through the stochastic delay bound of equation (2) with the values of the Table 5.1 but first, we decided to use the values of [18] in order to compare the results and thus, verifying the correct application of the network calculus framework.

![Figure 5.1 One server - One flow](image)

A second scenario addressed the delay of a flow that traverse 2 nodes in chain topology (Figure 5.2), where we assumed that the channel condition between the network elements are all alike.

![Figure 5.2 Two servers - One flow](image)

And to finish, we analyzed the delay that experiments a through flow when a server has to deal with cross traffic. This experiment comprises two scenarios, first the traffic flow is considered as another periodic source (Figure 5.3) and second, the traffic flow is considered as the departure process of a network element, whose input was another periodic source (Figure 5.4). We used the General Scheduling Model, which is conservative for most scheduling disciplines to compute the delay bound of flow $A_1$.

![Figure 5.3 One server - Two flows](image)
5.2 Implementation and Results

As part of the validation, we implemented the delay bound of equation (2) (i.e. analytical model) and a MATLAB script that simulated one periodic source and a Markov channel using for both the values of Table 5.2 (see [18] for discussion). The infinite sum of the delay bound expression is calculated for the first 2000 units of time and the tail is estimated using geometric series, which is a similar approach as [18].

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{q}{p + q} )</td>
<td>Steady state block error probability</td>
<td>0.1</td>
</tr>
<tr>
<td>( p )</td>
<td>Transition probability from OFF to ON</td>
<td>0.1</td>
</tr>
<tr>
<td>( h )</td>
<td>Channel rate</td>
<td>4</td>
</tr>
<tr>
<td>( T )</td>
<td>Period of the source</td>
<td>8</td>
</tr>
<tr>
<td>( \sigma )</td>
<td>Burst size</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 5.2 Parameters used to validate

Figure 5.5 shows the delay results of the single wireless link analytical and simulation model. We can see that the analytical bound is actually the worst case that the system can face for the given properties and because of that, it is the expected behavior that the analytical curve presents higher values than the simulation curve. When comparing them with the results of M. Fidler, we could validate that our expressions and code are correct because we obtained exactly the same results. This validation allowed us to move on to the experiment with the values of source and channel model for the Project Vital Responder 2.0.
Figure 5.6 shows the delay bounds results for the one server-one flow, two servers-one flow, one server-two flows and two server-two flows with parameters of Table 5.1. As it was expected, the two servers-one flow case (red curve) presents higher delays bounds than the one server-one flow case (blue curve), incrementing the delay bound an average of 160 ms in all the violation probabilities.

In the same way, the one server-two flows case (green curve) shows how the presence of other flow impacts the delay bound, presenting higher delay bounds than the blue curve because the flow of interested is served with the leftover service. And lastly, scheduling with cross traffic flow that has already pass through a server (black case) is the highest delay in the figure because that flow is bounded already with the properties of the first server when it arrives to the second one.

We wanted to find the delay bounds for other topologies with more than two wireless links, however, we could not do it because the search for \( \tau \) and \( \theta \) becomes too hard in terms of computational resources. For instance, for the analysis of the one flow-three server, the delay bound equation to solve is

\[
d = \inf_{\theta > 0} \left\{ \frac{1}{\theta} \left( \log \sum_{s=1}^{2000} M_d(\theta, s - \tau) \sum_{t=0}^{\tau} \left( \sum_{t_1=0}^{t} M_S1(\theta, \tau_1)M_S2(\theta, \tau - \tau_1) \right \} M_S3(\theta, s - \tau) - \log e \right) \leq 0 \right\}
\]

which for a value of \( \tau = 200 \), leads to the first sum to be from 200 to 2200, which in turn leads to a sum from 0 to each of the previous values, which in turn leads to another sum from 0 to each of the previous values. In addition, this process needs to be repeated until you find the optimal \( \theta \).
Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis showed the performance of an IEEE 802.11 ad-hoc network at different levels by means of an experimental study. First, we were able to deploy an ad-hoc network in the low-cost computer, known as Raspberry Pi, establishing the combination chipset/driver to accomplish the communication between the nodes, where Atheros and Ralink chipsets proved to be ad-hoc capable.

The current consumption in the Raspberry Pi in ad-hoc mode is mainly due to hardware (Raspberry itself and WiFi-USB adapter), which leaves a small margin of improvement from the network point of view to minimize current consumption. In addition, for field deployment of the RPi is needed a battery that supplied a minimum of 400 mA.

The communication range in a wireless environment is highly variable, even during the same experiment, and it decreases as the data sizes or data rate increases. We measured communication range from 30 m to more than 150 m, where the last distance was achieved by the effect of a 4 dBi external antenna.

The connectivity script written in MATLAB allowed to see what would be the time of disconnection in a real world implementation and how they are distributed. The results obtained show that firemen are usually connected throughout the firefighting but it is not necessary the case with the truck. However, a dataset with a larger amount of positions is needed to make more reliable conclusions.

The throughput obtained confirmed other results previously reported for UDP traffic and also, how it is affected while the number of nodes increases in the network. We also could establish the limits of transmission in the RPi due to hardware limitations.

Network Calculus proved to be a complex tool that allows the systematic study of the service guarantees for wireless network. We could build and study the end-to-end delay of the
basic blocks of any multi-hop topology, therefore any more complex topology can be analyze from our results but power computational resources will be required.

### 6.2 Future Work

Use the RPi testbed to study MAC, routing and transporting protocols in MANETs with Linux-based Kernel.

Extend the connectivity MATLAB script to simulate the behavior of the most popular routing algorithms and compute more statistical properties of the nodes connection.

Study the end-to-end delay bound of more complex topologies, in especial, the 5-node case of the Vital Responder 2.0 Project.

Model other schedulers to improve the analytical bounds when cross traffic is considered and describe the service process with physical fading channel parameters, i.e., signal to noise ratio rather than high layer models.
Bibliography


Annex A - Current Measures

In the next figures are presented the evolution of the current versus time for the Wi-Pi, N150 Ralink and TP-Link adapter in ad-hoc mode sending at 500kbps, 5Mbps and 10Mbps, respectively.
Annex B – Connectivity

In the following tables is presented the probability of connection between firemen.

<table>
<thead>
<tr>
<th>Fireman 1</th>
<th>Probability of connection</th>
<th>Time connected (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.63</td>
<td>11939</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.61</td>
<td>11520</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.82</td>
<td>15483</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fireman 2</th>
<th>Probability of connection</th>
<th>Time connected (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.63</td>
<td>11939</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.68</td>
<td>12868</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.65</td>
<td>12310</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fireman 3</th>
<th>Probability of connection</th>
<th>Time connected (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.61</td>
<td>11520</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.68</td>
<td>12868</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.45</td>
<td>8466</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fireman 4</th>
<th>Probability of connection</th>
<th>Time connected (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fireman 1</td>
<td>0.82</td>
<td>15483</td>
</tr>
<tr>
<td>Fireman 2</td>
<td>0.65</td>
<td>12310</td>
</tr>
<tr>
<td>Fireman 3</td>
<td>0.45</td>
<td>8466</td>
</tr>
<tr>
<td>Fireman 4</td>
<td>−</td>
<td>−</td>
</tr>
</tbody>
</table>

In the following figures is presented the distribution of time of disconnection between firemen and sink.