HAL open science

Enhancing Experimentation in Wireless Networks

Diego Roberto Dujovne

To cite this version:

Diego Roberto Dujovne. Enhancing Experimentation in Wireless Networks. Networking and Internet Architecture [cs.NI]. Université de Nice Sophia Antipolis, 2009. English. NNT : . tel-00408682

HAL Id: tel-00408682

https://theses.hal.science/tel-00408682v1

Submitted on 31 Jul 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITE DE NICE-SOPHIA ANTIPOLIS

ECOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

T H E S E

pour obtenir le titre de

Docteur en Sciences

de l'Université de Nice-Sophia Antipolis

Mention: Informatique

présentée et soutenu par

Diego DUJOVNE

ENHANCING EXPERIMENTATION IN WIRELESS NETWORKS

Thèse dirigée par Walid DABBOUS et Thierry TURLETTI

soutenue le 7 Mai 2009

Jury :

Michel RIVEILL	PROFESSEUR	président
Marcelo DIAS DE AMORIM	DOCTEUR	rapporteur
André-Luc BEYLOT	PROFESSEUR	rapporteur
Mohamed NAIMI	PROFESSEUR	examinateur
Jorge FINOCHIETTO	PROFESSEUR	examinateur
Walid DABBOUS	PROFESSEUR	directeur de thése
Thierry TURLETTI	DOCTEUR	co-directeur de thése

"The only truth is reality."

Aristotle

UNIVERSITE DE NICE SOPHIA ANTIPOLIS

Abstract

ECOLE DOCTORALE STIC
Université de Nice Sophia Antipolis

Doctor in Science

by Diego Roberto Dujovne

Since the inception of the 802.11 standard in 1999, WLANs, which used to be exceptional, became a massive phenomena together with the evolution of portable devices. At the same pace, research on wireless networks, where both simulation and experimentation are used to validate protocols, has evolved rapidly. Nevertheless, the models used in this area were adapted from the wired paradigm, which has led to a significant gap between the simulated and the experimental results. Therefore, to validate wireless protocols or algorithms, wireless experimentation remains as an important resource. This thesis explores the improvements to experimentation on WLANs, from a methodological point of view. The main contributions of this thesis are: First, the new model for data abstraction to represent network events and aggregated data logs; second, the methodology to manage data in order to support a database model; and third the replacement of custom made processing scripts with data-oriented filter modules. These three key points converge to a proposal of a wireless experimentation methodology in order to attain reproductible experiments. We implement this methodology within a Wireless Experimentation Tool, and furthermore we show the improvement results through a wireless multicast experimentation use case.

Acknowledgements

I would like to thank PLANETE group at INRIA Sophia Antipolis where this thesis was completely developed. I would also like to thank to my directors, Walid Dabbous and Thierry Turletti who encouraged me with their support and the discussions on the ideas which became part of this thesis.

I thank to my friends, and specially the local chilean community, who helped me understand french life and shorten the adaptation process; but most of them all, for the times spent together during my studies.

Finally, I would like to mention the Saint Exupery scolarship program between Argentina's Education ministry and the French Embassy in Argentina, which provided part of the financial support at the start of my thesis.

Contents

Abstract ii

Acknowledgements iii

List of Figures vii

List of Tables ix

1 Introduction 1
- 1.1 Wireless Networks Ubiquity . . . 1
- 1.2 Wireless protocols evolution . . . 2
- 1.3 Models are often too simplistic . . . 3
- 1.4 Are simulations realistic enough? . . . 4
- 1.5 General comparison . . . 6
- 1.6 Experimental Platforms . . . 7
- 1.7 What follows . . . 9

2 Wireless Networking Evaluation 10
- 2.1 Introduction . . . 10
- 2.2 Physical channel model simplifications . . . 10
  - 2.2.1 Freespace model . . . 11
  - 2.2.2 Two-Ray Model . . . 12
  - 2.2.3 Rayleigh Model . . . 12
  - 2.2.4 Specific enhancements . . . 13
  - 2.2.5 Complexity increases . . . 15
  - 2.2.6 Other simulation pitfalls . . . 17
  - 2.2.7 Experimental platforms pitfalls . . . 18
- 2.3 MAC impairments . . . 19
  - 2.3.1 Busy state detection . . . 20
  - 2.3.2 Capture Effect . . . 21
  - 2.3.3 Hidden and exposed station . . . 21
  - 2.3.4 Collision vs. Noise/Interference . . . 22
- 2.4 Implementation bugs . . . 23

3 Wireless Experimentation Tools 25

3.1 Introduction . . . 25
3.2 Experimental Platforms classification . . . 25
3.3 Current platform initiatives . . . 30
3.4 General Purpose wireless measurement tools . . . 33
3.4.1 Requirements for wireless measurements . . . 34
3.4.2 Tools for Wireless Measurements . . . 35
3.4.2.1 Tools Based on Driver-Level Statistics . . . 35
3.4.2.2 Measurement Tools Based on Packet Traces . . . 37
3.5 Conclusion . . . 45

4 Wireless Experiment Control 47
4.1 Introduction . . . 47
4.2 Existing proposals for experiment control . . . 47
4.3 Experimentation . . . 49
4.3.1 Spatial considerations . . . 49
4.3.2 Temporal considerations . . . 50
4.3.3 Topology and Software considerations . . . 51
4.3.4 Experiment design . . . 51
4.3.5 Metrics . . . 53
4.3.6 Per-BSS Statistical Parameters . . . 55
4.4 Experimentation sequence . . . 56
4.5 What to measure? . . . 57
4.6 Conclusion . . . 60

5 Wextool: A tool for experiment control 61
5.1 What is experiment control? . . . 61
5.2 Design . . . 63
5.3 Wextool . . . 66
5.3.1 Technical requirements . . . 66
5.3.2 802.11 physical platform components . . . 68
5.3.3 Environment description method . . . 69
5.3.4 Experiment abstraction and scheduling . . . 69
5.3.5 Description of Modules in Wextool . . . 70
5.4 The General structure of the database . . . 72
5.5 Data management module specification . . . 74
5.6 Current format for measured data representation . . . 76
5.6.1 PCAP . . . 76
5.7 An Example Packet Log Database . . . 77
5.8 Flexibility and Scalability of the data structures . . . 78
5.9 A simple experiment . . . 79
5.9.1 Layout definition . . . 79
5.9.2 Experimental Parameter configuration . . . 80
5.9.3 Multiple runs, capture . . . 81
5.9.4 Processing and Analysis . . . 81
5.9.5 Storage . . . 81
5.10 Conclusion . . . 82

6 Leader-Based Multicast: The Case study **83**
6.1 Brief background on Multicast in 802.11 networks . . . 83
6.2 The challenge . . . 85
6.3 The metrics . . . 85
6.3.1 WisMon . . . 86
6.4 We created the first method . . . 87
6.5 Automate the experiment steps . . . 89
6.6 The first testbed . . . 89
6.6.1 Physical Setup . . . 91
6.6.2 Leader-based implementation issues . . . 92
6.7 Experiments and Results . . . 93
6.7.1 Why legacy multicast does not work . . . 93
6.7.2 Comparison between legacy multicast and the leader-based mechanisms . . . 96
6.7.3 Packet loss correlation between stations . . . 96
6.7.4 How to select the leader . . . 99
6.7.5 Performance analysis at the application level . . . 100
6.8 Using Wextool . . . 102
6.8.1 Nodes . . . 103
6.8.2 Software . . . 103
6.8.3 Experiment description . . . 104
6.9 Improvements . . . 104
6.10 The experiment results . . . 108
6.11 Conclusion . . . 109

7 Conclusion **111**

A Formal description of the multicast experiment **114**

B Publications **123**
B.1 Conferences, Journals . . . 123
B.2 Proposals at the IEEE 802.11aa TG . . . 123
B.3 Registered Applications . . . 124

Bibliography **125**

List of Figures

2.1 WLAN simplified packet structure 20
2.2 Hidden station condition 21
2.3 Exposed station condition 22

3.1 Platform classification by criteria 26
3.2 Location of Driver-Level statistics tools 37
3.3 Modular structure of Wismon 41

4.1 Relationship between Experiment, Test and Run 53

5.1 Experimental Methodology 63
5.2 Relationships between modules 70
5.3 An example of database structure 77

6.1 Screen capture of station list window from WisMon 87
6.2 Screen capture of the real-time graphics from WisMon 87
6.3 Instrumentation on the driver queue to measure packet drops 88
6.4 Distribution of the AP and wireless stations. There are two groups: near (STA1-STA2-STA4) and far (STA3-STA5) 91
6.5 Receiver side of a 802.11 WLAN board 93
6.6 Goodput multicast - unicast unfairness without background traffic 94
6.7 Interarrival time multicast - unicast unfairness without background traffic 94
6.8 Goodput multicast - unicast unfairness with background traffic 95
6.9 Interarrival time multicast - unicast unfairness with background traffic . . 95
6.10 Multicast Packet loss correlation: Uncorrelated packet losses (independent), Collisions (same packets lost by all the stations) and Queue Losses (AP packet drops) for each of the 4 experiments. 97
6.11 Per-Station Packet loss without background traffic 99
6.12 Multicast packet loss correlation for each group of stations 100
6.13 Cumulative Distribution Function of Packet Error Burst length 100
6.14 Video goodput for legacy multicast and leader-based mechanisms 101
6.15 Layout of the Experimental Lab 102
6.16 Processing time for parallel and serial database insertion 106
6.17 Processing time for serial database insertion by station 106
6.18 Processing time by function with number of inserted packets before optimizations 107
6.19 Processing time by function with number of inserted packets after optimizations 107
6.20 Packet Loss by source 108

6.21 Burstiness by experiment 109

List of Tables

2.1 β related to the environment characteristics 13
2.2 σ related to the environment characteristics 13

3.1 Input and Output formats for each of the tools 43
3.2 Wireless measurement tools 44

4.1 Per-Flow Statistical Parameters 54
4.2 Per-BSS Statistical Parameters 55
4.3 Common DLT header values 60

6.1 Mean receiving power value for each station 91
6.2 Table of Experiments 96

Dedicated to my parents, Adela and Adolfo, and my brothers, Alejandro and Gabriel who have always been my strongest links to life 14.000 kilometers from here, and to the memory of Jorge, who helped me follow the engineering path.

Chapter 1

Introduction

1.1 Wireless Networks Ubiquity

Today, connectivity is the engine which powers the information society. Without the current level of connectivity, today's content could have never arrived to so many people.

Connectivity is possible because of the existence of a network, which provides the paths and the means to let the information flow. The biggest network we have is the internet, which has evolved from an experimental communication platform, to the ubiquitous presence we know today. This could not have been possible also if network interface devices had not increased their processing power and become more accessible to users.

All these factors describe a reality where connectivity is taken for granted. Whenever a link is established, it is guaranteed that data will flow at the highest possible rate without any loss. Unofortunately, this is not true. So far bit errors on wired links have been considered small enough not to have a real influence on the link behavior. Thus, the only factors which influence the behavior of a network would be the delay and congestion.

The arrival of wireless networks changed these hypotheses completely. Wireless links have a much higher and dynamic BER (Bit Error Rate): it depends on variable channel conditions. Furthermore, users are not constrained to a cable connection, they become mobile users, and the channel characteristics between their station and the gateway are changing constantly. From a user's point of view, connectivity must be provided everywhere, which creates the problem of wireless resource allocation.

Wireless links changed completely the original static networking paradigm. From static backbone networks in the center and a dynamic mobile access networks at the borders to

the creation of wireless-only self-configuring networks, such as wireless mesh networks, for example.

The speed on innovation in the field is a consequence of an explosion of new wireless-enabled devices which can have multiple wireless interfaces (bluetooth, 802.11, wimax and 3G) to keep the user connected as long as possible. Wireless convergence techniques and compatibility issues are far from being solved, and, far worse, the wireless environment has become increasingly crowded. Consequently, the task to establish a wireless link and predict its performance on time is increasingly harder.

1.2 Wireless protocols evolution

On the effort to provide wireless data connectivity to the users, many protocols were proposed, but only those which were practical and applicable arrived to a wide deployment stage. Some examples of these protocols are UMTS [1], WIMAX [2], IEEE802.15.4 [3] and IEEE802.11 [138].

These protocols describe the MAC (L2) layer for the network; and they can be classified in centralized and distributed resource assignment. An example of fixed resource assignment is GSM [1], which gives users connectivity through a combination of TDMA and FDMA which assigns each user a specific time slot and frequency to transmit or receive. The user has not the right to use the network in any other slot or frequency. Since access control and management is the responsibility of one entity, this is a structured hierarchical approach which resembles the standard wired telephone system; and is mostly predictable in terms of available bandwidth and delay.

On the other hand, a distributed method gives control to the stations following a simple set of rules. There is no centralized control and all the stations decide by themselves when to transmit.

802.11 is an example of a distributed control MAC access level protocol. It is based on Ethernet's exponential backoff mechanism as a solution for multiple access. However, implementing Ethernet directly on a wireless local area network is not feasible. So, to overcome the wireless channel unfriendliness many changes were done to improve the reliability of the transmission. Such as the inclusion of an acknowledge packet after each unicast data packet is sent; the addition of a request to send/clear to send mechanism to avoid collisions when two stations do not listen each other's packets, and the management packets to cope with user roaming.

Although these additions are helpful, they were not enough to give the same reliability and fairness that wired distributed access networks have. If the MAC layer cannot solve a packet loss due to bit errors, for example, then a higher layer should take care of the reliability. To follow the example, if this lost packet was a TCP data packet, TCP will understand there was a congestion on the link, and reduce the packet sending rate by a half. The mix of an unreliable channel, unplanned wireless coverage and distributed control access results in a complex phenomena which propagates to higher network layers. This phenomena itself constitutes a research field, which connects the networking and communication worlds.

As a result, protocol and algorithm development at the MAC layer for wireless is done taking into account the characteristics of non permanent connections and the nature of the wireless channel. Furthermore, wireless protocol design is currently a very dynamic field, which arises from the numerous amendments to the 802.11 standard, where the IEEE standards board needed even to use a two-letter code for the new amendments (e.g. the 802.11aa proposal), to the creation of new wireless protocols, such as LTE. Additionally, each protocol or algorithm proposal has a corresponding validation stage. In the following sections we focus on this validation process.

1.3 Models are often too simplistic

From the communications point of view, the wireless channel's behavior can be approximated with models which combine a simplified version of electromagnetic propagation effects with the distance and speed between the transmitter and the receiver.

Many simplified channel models have been proposed, like Friis [4], Rayleigh [5] or Ricean [6] models, which combine free-space attenuation with signal fading, reflection and dispersion. If the devices have movement, then the Doppler [16] effect is also present. These statistical models try to mimic a channel's transmission power profile in time so as to predict which is the signal's power value at the reception point.

However, models usually do not take into account the background noise and the interference from other users. In fact, to be able to send data through a channel, the information is processed within the PHY layer, with a protocol dependant channel coding and modulation. For each type of modulation scheme, there is a minimum signal-to-noise ratio to respect in order to decode the data correctly. The higher the density of symbols transmitted, the higher the signal-to-noise ratio required to decode the packet. But, in reality, the receiver not only listens to the transmitter, but captures thermal noise (from the antenna and from the radio frequency amplifier front end at the reception side),

interference from other devices (like sparks from engines or radiation from a microwave oven) and also other transmitter's signals. Current simulation engines in development, like ns3, take into account the difference in signal-to-noise ratio as the deterministic factor for packet arrival decision, while leaving to an statistical channel model the rest of the decision.

Today, the interference generation on the wireless unlicensed bands is a fast growing problem. 802.11 access points are being deployed without planning and clients are generating traffic on the selected channels. Since the unlicensed spectrum is scarce, 802.11 channels were not assigned on an orthogonal basis, so that only a 3-channel distance will guarantee that there is no meaningful influence on the current channel. A recent proposal related to this problem was published by Maureira et al [7]to add wireless traffic extracted from reality to simulation. This proposal uses capture logs to create virtual sources from approximated locations and replay the captured traffic within the simulation.

The former elements represent part of the aspects which are often not taken into account when modeling a wireless channel. This implies that models do not represent a good approximation of "reality".

1.4 Are simulations realistic enough?

Simulators like ns2 [14] or Omnet++ [15] have tried to combine modeling and reproduction of the functioning of a real system in order to give more accuracy to the simulation process. Both have modules to include wireless stochastic models for 802.11 networks, which are based on the simplifications described in the former section. Thus, the channel module follows a stochastic model from the communications point of view, since simulation on a wireless environment with spatial or temporal propagation considerations is extremely expensive in computation time. As a consequence, experimentation remains as the only solution to the validation process which consider the changing wireless environment.

In fact, for fixed scenarios, the electromagnetic wave interaction with the environment is responsible of the signal level and the variability of the signal. In a simulation, the environment is free from obstacles, then the reflection, dispersion and fading factors are not tuned to any physical reality. There is no one standard or general environment to adjust to, but there are classes of environments which represent typical situations for users, like conference halls, offices, trains, buses, airports and coffee shops. These typical places can be classified into Indoor, Outdoor and Indoor/Outdoor environments,

using a LOS (Line Of Sight) or NLOS (Non Line Of Sight) criteria. For example, an Indoor environment is rich in signal reflections from different surfaces of objects Indoor offers an alternative path to signals to overcome an obstacle, while Outdoors have fewer reflections, giving a smaller chance to obtain an alternative path. Channel variations between LOS and NLOS are also generated from moving objects, which brings another important source of variability. Either the stations are fixed and people or objects (like vehicles) are moving around, or the stations themselves are in movement, with or without an environment in movement too. Every time an object blocks the path between a transmitter and a receiver, the LOS path disappears, and the only path is the NLOS (if it exists). Current simulators do not include a facility to represent moving obstacles in a deterministic way.

If the stations have mobility, the variability of the channel is even higher, since the station's relative speed to the other stations distort the signal in such a way that complex signal modulations like CCK [17](used on the 11Mbps 802.11 rate) are deeply affected, rendering information decoding impossible. Denser modulations like the current OFDM [18] implementation or the future application of MIMO [19] techniques on 802.11n increase the variability of the channel as seen by the MAC and upper layers, rendering the experimentation even more dependable on physical factors.

Furthermore, the channel also suffers from a variable distance to the source, and from the static and moving obstacles found on the path. This condition may be seen as a device moving through a sequence of different channels, each of them defined by the wireless conditions at the point on the path, at the instant when the device passes, superimposed with the attenuation due to the moving obstacles plus the doppler effect due to the relative speed between the transmitter and the receiver. All the former channel conditions and variations at the channel level are not considered in a deterministic way on the current models, creating a wide gap between what is simulated and what is obtained in an experimental platform.

On the other hand, interference on simulations is taken into account on the most basic level. If a distributed access method is used, during a packet transmission, SNR values are calculated by the simulator for each packet and compared only at the end of packet reception, regardless of the duration. As Takai et al [20] detailed within their comparison between simulators, that standard ns-2 802.11 standard's implementation uses SNRT (Signal to Noise Ratio Threshold) criteria. As a consequence, the decision of packet reception is based on the power comparison between the two concurrent packets, without taking into account the SNR for the current modulation scheme. Furthermore, there is no rate adaptation algorithm by default, and the transmission rate used is the same with a fixed SNR to enable detection.

Also, inter-channel interference on 802.11 is not considered: Packets sent on nearby non-orthogonal channels increase the noise level and, what is even worse, can also be decoded by the receiver and compete for the current channel usage. The interference coming from other sources like Bluetooth devices increase the noise level too. Bluetooth [8] uses FH (Frequency Hopping) spread spectrum transmission which sends data on a sequence of channels. For example, since the 2.4GHz band is shared between Bluetooth and 802.11 networks, a Bluetooth signal can arrive to a 2.4GHz channel and interfere with a transmission. New communication protocols like 802.15.4 used on sensor networks can also use the 2.4GHz spectrum band, adding even more interference. This interference can be regarded as mutual: A 802.11 packet generates interference to Bluetooth - and 802.15.4[3]-based devices too.

To summarize, interference models on simulators are not rich enough to take into account the always increasing number of devices trying to share the same wireless band. Furthermore, although simulation engines as NS-2 offer a controlled and reproducible environment, the simplifications done a the MAC and PHY levels to reduce the simulation processing cost creates a gap between simulation and experimental results. As a consequence, simulation, although a powerful resource to analyze the behavior of wired networks, is currently a weak tool to represent wireless network behavior.

In order to complete the validation process, the first step is to use an experimental platform to measure and compare the performance of the protocol.

1.5 General comparison

Kotz et Al. [9] have analyzed six assumptions which are common on simulation papers. They have done this study based on an extensive set of measurements from a large outdoor routing environment, to further demonstrate the weakness of these assumptions, consequently giving different results. **The first assumption** accounts for the physical environment modeled as a flat 2D surface. Hills and buildings, and even smaller heights (like waist-to-ground distance) change significatively the propagation model, and when nodes are installed within buildings, they are generally spawned through several floors. **The second assumption** is uniform isotropic radio propagation, which comes together with **the third assumption**: all radios have the same range. This accounts for the construction differences which make radios similar but not identical even if they are the same model. Furthermore, antennas are not perfectly omnidirectional, background noise varies from one place to the other, and finally, objects and people block or reflect wireless signals. They have shown these variations with the measurement of the reception probability at different angles, with different quadrants, and between all the participant

nodes. **The fourth assumption** states that if a node A can hear node B, B can hear A too, or, links are perfectly bidirectional. They proved the contrary by the examination of symmetric beacon reception between two nodes, meaning that if a beacon from node A is received on B, then immediately B sends a beacon to A, to preserve the most similar channel characteristics. They have shown that the behavior is asymmetrical for the same two nodes at further distances and for all the participating nodes. **The fifth assumption** is that if a node A can hear node B in any situation, it can hear B perfectly. This is not true, since the limit between reception and non-reception given by the range is not sharp, but a function of distance related to the reception probability: Some packets still can be listened, while some of them will not be listened although the receiver is in range. **The sixth assumption** says that signal strength is a function of distance. While this is true as an average value, the variations due to reflection, refraction, obstruction and scattering should not be overlooked. They show this fact by creating a graph of all the measured power values with distance, giving scattered values, instead of a simple power curve.

1.6 Experimental Platforms

Experimentation on a real network fulfills the validation requirements of wireless protocols and algorithms. Wireless experimentation is done in platforms, which can be classified into two types: *Controlled testbeds*, which offer controlled conditions for the experiments as Emulab[51], Orbit[52] and GRID5000[22]; and *Physical Tesbeds* aiming to do experimentation on top of production networks, which provide access through virtualization to real working conditions like Planetlab [53] and Onelab [54].

In physical testbeds, the traffic generated by an experiment shares real paths and restrictions with other users' traffic. Since networking conditions vary with time, experimental results obtained will also vary. To take into account these varying conditions, it is important to use a well-defined approach to first establish the experimental setup, executing the experiment, capturing, processing and storing the results. In this case, it would be easier for researchers to compare between experiments which were executed under the same setup from one instance of an experiment to the following. For example, if the load increases, it is interesting to be able to check whether the protocol or application under test will react (or not) to this change. In order to analyze this possible reaction, the traffic variation load must be registered too.

The required approach should concern the main objective of networking experimentation: the validation of an algorithm, protocol or application under realistic conditions. Unlike controlled testbeds, whose aim is to generate network conditions so that one is

able to reproduce the same experiment, real overlays provide no control on network conditions. Basically physical testbeds enable the reproduction of experiments with the same setup in a different time or place and to analyze in detail the effect of varying network conditions on the performance on the protocol under study.

Physical testbeds provide a combination of non controlled features including real network load and variable channel conditions with interference; while keeping control on the layout, setup and scheduling of the experiment.

To realize automatic experimentation or *workflow management* as it is called in Grid environments, the setup of the experiment must be well defined; this is a critical part of the experiment executed within a wireless network, since not only traffic load or delay can change with time, but also transmission conditions due to environmental factors. All the participating devices (including stations, access points, routers, switches and links) have a specific configuration which must be preserved in order to reproduce these conditions.

During the execution of the experiment, network variations must also be monitored and registered, in order to correlate them with the captured data. Another important item is the data capture process: on the one hand, this process describes all algorithms and statistical functions to obtain the results; on the other hand, raw data processing extracts parameters that can be used to compare the different instances of the same experiment between them. Finally, all these items must be classified and stored in an efficient way to enable easy access to the data. In a later stage, this stored data can be used to extract new results, and to build new experiments based on the same setup.

In order to execute experiments within a controlled schedule, the platforms must provide a tool to control the evolution of the experiment. Specific workflow management tools exist for grid applications, as it was shown by Jia Yu et al. [97]. On the other hand, several experimentation management tools exist in the context of the Emulab project [98], which are designed for controlled experimentation environments and they cover experiment definition using NS2 compatible scripting, control and data storage. On the physical testbeds side, there are few proposals, like Plush [100], and Weevil [99] both oriented to deployment, remote execution and monitoring of distributed applications on overlays. However, they do not include the processing and analysis stages, which are indispensable to display the experimental results [101].

1.7 What follows

As we have seen on the previous sections, validation of wireless protocol and algorithm proposals must include an experimentation stage. Furthermore, the resources to do such experimentation are available, but the specification, organization, data management, event scheduling, data processing and extraction of results are not standardized nor uniform.

The objective of this thesis is the improvement of wireless experimentation starting from the definition of an experimental methodology, and the creation of a set of rules which define the interfaces and structures of a software model to finally propose and develop an implementation which follows those rules and provide a use case for an application.

The main contributions of this work can be summarized as follows: The creation of a flexible and scalable data structure which represents and integrates both the event-based and statistical data captured during a wireless experiment; The design of a methodology to improve the reproducibility of the experiments through the description of the layout and experiment, and the traceability of the results through storage and data processing techniques; The development of the joint time-hash packet synchronization method, which uses a time window and a hash-based verification to insure the proper and coherent packet log merging between probes; The experimental study of the leader-based approach to measure the 802.11 multicast protocol in order to evaluate the bandwidth and reliability of streaming within a real 802.11 infrastructure network;

The plan of the thesis follows: On chapter 2, we describe the wireless network evaluation through simulation. We detail the limitations of the simulated MAC and PHY layers, and we point out the lack of accuracy of the results for several conditions, which motivate the use of wireless experimental platforms. On chapter 3, we describe the current available tools to instrument and measure on experimental platforms, their scope and their pitfalls to reveal which are the possible enhancements. On chapter 4, We propose a set of considerations to achieve these enhancements at the layout, setup and storage levels. On chapter 5 we transform the considerations into a methodology and describe the implementation called Wextool in detail. On chapter 6 we evaluate our implementation through a comparison between a standard measurement and Wextool through a use case of experimentation for a protocol enhancement of 802.11 Multicast. Finally, we conclude the thesis on chapter 7.

Chapter 2

Wireless Networking Evaluation

2.1 Introduction

In this chapter, we describe the current simulation's simplifications and limitations of their implemented models, based on the study of the standard public version of ns2. Then we go through some of the improvements on wireless modeling, which describe the underlying complexity of the electromagnetic propagation and we summarize some of the 802.11 MAC impairments, a direct consequence of the MAC implementation. We conclude the chapter with a reference to the 802.11 implementation bugs on ns2 and the lack of validation of the current wireless model.

2.2 Physical channel model simplifications

Wireless simulations are based on the belief that current MAC and PHY models are accurate enough to represent the behavior of a real network within the specified conditions. Currently used channel models are thoroughly studied by Rappaport on his Wireless Communications book [13]. To describe the channel models used in simulations, we'll take the ones used in the ns2 simulator as a reference, given its widespread use: Kurkowski et al [24] show within their study that between 2000 and 2004 in the MobiHoc conference, ns2 usage represents 44.4% of the published papers, followed by the group of self-developed simulators with 24.5%. In the ns2 documentation [14], there are several channel models available for the execution of simulations. Here we present a brief analysis of each of them, highlighting the wireless propagation simplifications.

2.2.1 Freespace model

The simplest model is the Freespace model, which considers only the power attenuation point of view of signal, as if both the signal source and destination are in the middle of vacuum, with no obstacles, noise nor interference whatsoever. It also supposes an ideal isotropic radiator which distributes the power equally in all directions. As a consequence, if an ideal sphere with radius d is built centered on the isotropic radiator, the power on the surface of the sphere will be uniform and will respond to:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L}$$

Where P_t is the power transmitted by the radiator, G_t is the radiator gain, G_r is the receiver gain (if both the radiator and the receiver are isotropic, the gain is 1), L represents other sources of attenuation, and λ is the wavelength. This model accounts for several simplifications:

- An ideal power source: real antennas are not ideally isotropic.
- Time is not taken into account, so the power is constant at the same position if the source is static. The same reasoning applies for speed, since there is no time taken into account, the channel is not influenced by the doppler effect.
- Although the formula allows for variable gain, it is aimed for point to point link calculations. A directional antenna needs a reference to one angle in two dimensions or two angles in three dimensions.
- There is no interference from other sources, nor a variable interference due to obstacles moving between radiator and receiver.
- This formula supposes Line of Sight (LOS) propagation, without any possibility of Non Line of Sight (NLOS) with a two ray model, for example.
- Coherence time is not taken into account, so it supposes that the phase value of the signal is predictable during the transmission. As a consequence, there is no diffraction, scattering or dispersion of the signal.
- Reflected signals interact with the main transmitted signal. This interaction can generate spatial and temporal power attenuation, leading even to a complete loss of signal.

Although the Freespace is a primitive and rough model, it is used as a base level on simulations. As a general case, the increase in complexity of the channel model increases losses, with the inclusion of several environmental factors.

2.2.2 Two-Ray Model

The two-ray model includes the reflection of the signal on the ground. An increase in the distance increases the attenuation faster than the Freespace model:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2 h_t^2 h_r^2}{d^4 L}$$

Where d is the distance from the radiator, h_t is the height of the radiator referred to ground and h_r is the height of the receiver referred to ground.

For a small d, the Freespace model replaces the Two-ray model, in order to avoid power level oscillations.

2.2.3 Rayleigh Model

The Rayleigh model takes into account the fading through multipath effect, which is represented by the use of a random variable. The Freespace and the Two-ray models show only the mean value of the power, while an approximation which includes a random variable for power is considered to be a better approach:

$$\left[\frac{P_r(d)}{P_r(d_0)}\right]_{dB} = -10\beta log\left(\frac{d}{d_0}\right) + X_{dB}$$

On the left side of the equation, $P_r(d)$ is the power at a distance d referred to the power at a reference distance d_0. The relative power is measured in dB. On the right hand side of the equation, the relative power is proportional to a β factor, which depends on the environment, giving the factor between 1.6 to 6. Typical values, from ns2 documentation, are given on table 2.1.

The second term X_{dB} adds the random variable, which is Gaussian with σ_{dB} deviation. Again, depending on the environment, the typical values of σ, which range from 3 to 12, are described on table 2.2.

It is up to the user to select the representative combination of β and σ to establish a generic environment for the simulation's execution.

TABLE 2.1: β related to the environment characteristics

Environment		β
Outdoor	Free space	2
	Shadowed urban area	2.7 to 5
In building	Line of sight	1.6 to 1.8
	Obstructed	4 to 6

TABLE 2.2: σ related to the environment characteristics

Environment	σ
Outdoor	4 to 12
Office, hard partition	7
Office, soft partition	9.6
Factory, line of sight	3 to 6
Factory, obstructed	6.8

Although this model is thought to be more realistic than the freespace model, the only added feature is the statistical signal loss which depends on β and σ. Fading is not only a temporal but a spatial effect. Both effects should correlate the losses when two stations are on similar conditions, and not as individual states for each station. Of course, there is still no speed effect, nor interference from similar or foreign sources.

The most complex wireless model on ns2 to date is the Rayleigh model, (for the official distribution). On the other hand, ns3 [25] [26], implements a more flexible and detailed channel model, which was originally developed for ns2 and 802.11a. This models includes Noise, Interference and BER for the current packet modulation scheme within delivery probability, thus enabling the user to tune this factors independently. Nevertheless, this flexibility still requires the user to specify the statistical environmental factors which affect signal propagation.

2.2.4 Specific enhancements

There exists an implementation of the Ricean model by Punnoose et al. [16] for ns2, which is not included in the official release. This model takes into account small scale fading, including time correlation from the Doppler spectrum. The time correlation adds the possibility to generate more accurate burst errors. They propose to use a precomputed time-sequenced fading envelope. This sequence is repeated cyclically, to achieve longer simulations. Also, the sequence is generated in such a way that there are no discontinuities when repeated. This approach was chosen because a filter made

with the Doppler spectrum over the random numbers would increase the computational requirements. In this case, the generation of a sequence represents the real channel better, recognizing the existence of channel evolution between events.

Furthermore, a different sequence for each parameter configuration (maximum Doppler frequency, Ricean K factor, and time-averaged power) must be generated for each situation where any of the parameters change. The use of a cyclic sequence, although lighter for processing, repeats the error burst pattern during a simulation yielding to a non-realistic approach.

The former models belong to the currently available and mostly used models on wireless simulation.

Nevertheless, wireless propagation can be modeled in a more accurate way, using improved models, but resulting on an increased of calculation complexity. These improved models can be appreciated on the work of Durgin, Rappaport et Al. [12] on the comparison of PDF of the envelopes of channel models. On this publication, they created a general expression representing the voltages across the antenna, namely:

$$\tilde{V} = \sum_{i=1}^{L} V_i e^{j\Phi_i}$$

Where V_i is the amplitude for the multipath waves, and Φ_i the phase. For modeling purposes, the phase is considered as statistically independent random variables, uniformly distributed over the interval $[0, 2\pi[$.

Then, they classified each component into a group of factors, called Reduced Wave Grouping:

- A Specular Component, which represents a single arriving multipath wave (random phase Φ_i, constant envelope V_i).
- A Nonspecular Component, which groups two or more multipath waves as terms on the general equation
- A Diffuse Component, which is a Nonspecular Component with numerous individual waves, each of them carrying negligible power compared to the total average power of the Diffuse Component.

$$\tilde{V} = \sum_{i=1}^{N} V_i e^{j\Phi_i} + \sum_{i=1}^{M} V_i e^{j\Phi_i}$$

Here, there are N Specular Components and M Nonspecular Components.

The Diffuse component is a sum of many independent phase waves. They obey the central limit theorem, and they have a quadrature and in-phase components, both random Gaussian variables.

On their proposal, they call this general model TWDP, which can be tuned to arrive to a One-Wave, Two-wave, Rayleigh or a Ricean model. This generalization shows that wireless channel models are in constant evolution, and furthermore, they include more deterministic parameters than the former statistical models. Nevertheless, the processing power required to calculate the propagation path for each packet is still prohibitive.

A step further comes from the analysis made by Yu et Al. [27] which calculates the values for N and M as contributors to either the dominant specular waves or distributed diffuse components. In order to approximate the Rayleigh distribution, thus establishing that TWDP with $N > 6$ and Rayleigh distributions are similar.

2.2.5 Complexity increases

We can observe that whenever the model tries to approach to reality, it increases the complexity and becomes more environment-dependent (or layout-dependent) to deduce signal reflection and dispersion as major factors. Moreover, the space correlation is not taken into account for fading, dispersion or doppler distortion, since this is modeled as a random variable with the distance parameter only. For example, if a group of several stations suffer fading from an obstacle to the source, on a simulation they will be assigned the same probability of receiving a packet as if they were at the same distance than other non-blocked nodes.

Furthermore, techniques which propose the use of multiple antennas to transmit and receive (like the MIMO 802.11n draft amendment) bases their improvement on the existence of spatial diversity through multipath.

Wireless channel models are based on the calculation of the signal power at the receiver to compare this value to the noise+interference and decide if the packet will be marked as received or not. The models do not take into account the rate (and the modulation) to decide if the packet will be decoded or not. This issue is raised on [28] by Xiuchao, where he analyzes the ns2 models and adapts the configuration and the simulator to use the different reception thresholds and noise floors, according to the transmission rate, from the datasheet provided by the card brands. He also notices that the transmission rate of the packet preamble is different than the payload; as a consequence, carrier sense must be calculated for the corresponding rate modulation. Vyas et Al. [29] propose a study on

the 802.11a channel indoors with three configurations: room-room, room-corridor and corridor-corridor. They measure four parameters:

- Channel space and time variability. For three stations at the same distance, they measure the variability range over time for each of the receivers and the combined spatial variability, from the worst to the best receiver. From the results it can be observed that although the receivers should have a similar mean power value from the Freespace model, their values differ on 14 dB.
- Path Loss in Different Scenarios. For the three configurations described above, they can calculate the power attenuation distance, showing the that the variation in attenuation is steeper in the room to room situation. Neither of these conditions are are considered on the wireless channel models.
- Carrier Synchronization Although this item belongs to the hardware side, it depends on both the SNR and the ability of the hardware card to discriminate a signal from the noise and lock to that signal. On their paper they show that the synchronization probability (and consequently, the detection of the start of a packet) is a function of the SNR and it is not a sharp edge. .
- Data reception at different data rates. After the synchronization, the rest of the packet must be decoded, which depends on the selected data rate. They show that for low data rates between 6 Mbps and 24 Mbps, if the synchronization was successful, the rest of the packet was decoded. They notice that from simulations, the SNR difference needed between 6Mbps and 24Mbps is 10dB; in this case this was not true, since there was no advantage on using 6 Mbps than 24 Mbps to reduce losses. On the contrary, for higher rates, where modulations are more complex and error-prone, the power variability influence has an important role to determine the packet loss levels, with no direct relation to a fading process.

We must notice that these experiments were done for 802.11a, where the modulation schemes are more complex than 802.11b, with selected bands for carriers, embedded FEC and better immunity to multipath due to the cyclical extension technique on modulation. As a consequence, the packet decoding capacity of the receiver is expected to be better after synchronization than 802.11b. To sum up, the differences from the expected behavior from the simulations in complex (indoors, office) but typical environments cannot be represented accurately enough by the simulations' models available using only a Gaussian random variable.

2.2.6 Other simulation pitfalls

Another item which is not considered within the simulations is the time granularity. Simulators are event based, meaning that, on the one hand, the channel is continuous and the events are discrete: for example, when there is no traffic, the channel does not evolve. On the other hand, the power value assigned to a packet is supposed to be constant for the packet, without variations within the packet which may lead to bit errors, and consequently packet loss due to CRC check. As we have mentioned before, the paper by Punnoose et Al. [16] proposes a more realistic channel including evolution between events as a solution to numerically intensive calculation, but it has the disadvantage of cyclic behavior.

Another approach towards a more accurate channel is the detailed propagation model. In [31], section 3.2, Takai et al, describe the use of SIRCIM (Simulation of Indoor Radio Channel Impulse Response Models with Impulse Noise). It examines not only the power of the packet received and the noise, but also the payload, and defines if a packet is lost observing the percentage of bits correctly received whether or not the entire packet is correctly received. This model offers tuning through the use of LOS (Line of Sight) and NLOS (Non Line of Sight). Orthogonally to the NLOS/LOS configuration, there are other three parameters which characterize the environment: Open Plan, Hard partitioned and Soft partitioned. This model tries to go deeper on the hardware modeling at the signal level. Although this technique is more accurate, it is computationally intensive.

From a more general point of view, Shakkotai et al. [38] propose the use of Cross-layer design whenever a wireless network is present. In their publication, they synthesize two main differences between wireline and wireless networks. First, the wireless channel is variable over time and space, and has short-term memory due to multipath. This generates bursts of errors, which generates discarded packets at the receiver. In fact, the small scale fading changes the characteristics of the channel within a few milliseconds. Second, large-scale channel variation are spatio-temporal conditions which depend on the user's locations and interference levels. Which influence directly the channel access time from each user based on their location or relative speed, for similar data rate requirement.

A throughout discussion on the role of simulations and experimentation was published by Robinson et al [30]. They measure issues related to channel separation in multi-radio mesh networks, where each radio on the same node interferes with the other, which reduces to half the throughput from simulation if RTS-CTS handshaking is used. They also show how channel separation is not enough to avoid interferences, so extra shielding

between antennas (more than 35dB) should be used to increase the throughput. The throughput reduction within the same node due to inter-wireless interface interference is not considered on standard simulations. On the same publication, they also notice that simulation and experimentation must be calibrated in order to evaluate the simulation accuracy. The calibration is done under one-way traffic within a very simple layout and clean environment. This publication shows that experimentation under similar layouts and environments can be used for calibration and that experimentation gives more insight of other effects which are not taken into account within simulation and which may greatly influence the results.

We must acknowledge that the former cited publications are only a representative sample of the channel modeling efforts, and their limitations. None of the individual approaches is exhaustive because of two modeling limitations: If the model is too general, there is no room for particularities of the environment or the physical layer on the transceivers, while if the model is specific, it approximates one particular aspect of electromagnetic propagation on the environment, making the simulation expensive in computer cycles and consequently, time.

2.2.7 Experimental platforms pitfalls

Although we have recalled a long list of PHY layer simulation limitations, there are also a number of elements which limit wireless experiments. On the one side there are the commercially available wireless cards which range from minimal control possibilities to a relatively deep control of the MAC and PHY layer. On the other side there are fully controllable wireless devices which, although practical to test and measure PHY and MAC protocols and algorithms, are expensive and do not mimic with precision any commercial device. The tradeoff on wireless platforms is to use standard commercial wireless cards with the highest possible controllability. Nevertheless, commercial cards do not expose every embedded feature which in turn can lead to biased or mistaken results if those elements are not considered.

On this subject, and to compare the behavior of real wireless cards with the theoretical backoff behavior, Bianchi et al [94] have examined in detail the backoff period for different situations for a group of wireless cards using standard wireless drivers. They have shown that none of them performs exactly as the 802.11 wireless standard rules. Two main causes

Finally, Tinnirello et al [96] show how proprietary undocumented algorithms implemented in vendor-specific cards affect the measurements done with those elements.

They claim that outdoor based links are greatly influenced by the sensitivity modifications and antenna selection algorithms by Atheros. On their publication they show with measurements that results are significantly different when enabling or disabling both mechanisms, which may lead to biased experiments and furthermore, erroneous interpretation of the results.

Although experimentation is required to correctly validate a protocol or algorithm, it must be carefully done so as to take into consideration the possible bias on the method and the results of using commercially available devices to execute them. The experimenter must be aware of the limits of the expected results and of the possible (even undocumented) bias that may arise at the analysis phase.

The main dilemma at this point is the limit between controllability and realism of the experiment. If commercial cards are used with standard drivers, then the controllable items are only the publicly available and accessible elements, while the proprietary algorithms, although they may exist, should remain as the limit of a black box. On the other hand, if a completely customizable solution instead of a standard card is used, then the experiment is more controllable at the hardware level, but less representative of the perceived reality by the users. The ideal (and currently utopic) situation would be to have complete public access to the commercial implementation of the cards in order to understand and include bias and artificial errors in the experimental analysis, and thus bridge the current gap.

To sum up, experimentation is an indispensable step to validate an algorithm under realistic circumstances. An experiment can show performance variations, failures and unexpected behaviors which simulations cannot reflect due to their limited representation of reality. Nevertheless, an experiment is valid only when the setup is completely defined and when the environmental conditions and the traffic can be measured in such a way that it is possible to reconstruct the experiment.

2.3 MAC impairments

In this section we focus on the 802.11 MAC implementation, which is from the research point of view, the most widespread, available and standardized technology for wireless experimentation. Although there are other MAC implementations for other purposes, their use is currently less widespread and the measurement still need specific instrumentation.

The MAC layer defines the first step to insert information inside the communications channel in order to communicate within an equilibrium between overhead, fairness, delay and robustness. The implementation of the MAC layer should address most of the problems which are inherent to the communication channel, nevertheless, this is not always possible: A wireless channel using an 802.11 MAC, and working on unplanned (and possibly mobile) node configurations is the source of a group of impairments. These impairments are underrepresented within simulators, generally due to the high computation complexity and environment dependency. From the current literature, we can identify five main impairments: The channel busy state detection (also called clear channel assessment), the capture effect, the hidden station, the exposed station and the confusion between collision and packet loss due to noise or interference.

2.3.1 Busy state detection

To identify a clear channel situation, Giustiniano et al. [35] have worked on the identification of the sources of false channel busy signals: collisions and interference. Furthermore, they have developed a statistical method to identify each of them. They have worked on the development of estimators for frame loss proportion between collisions and interference. For this purpose, they have created experiments where they have used modified hardware drivers and firmware to capture the clear channel assessment function at the channel level. This instrumentation opened the possibility to create a cross-layer idle/busy estimator to show the probability that an error is due to a collision or to another cause. They also propose an estimator based on the observation of the difference between the PLCP errors and the CRC errors, with the hypothesis that the existence of a CRC error means that the PLCP was well-decoded. In this case, the payload section had bit errors, but did not suffer a collision. Nevertheless, there is a condition which is not considered: Bit errors not due to collisions also happen on the PLCP before any CRC is checked. Figure 2.1 shows the PLCP header and CRC field location within the WLAN packet.

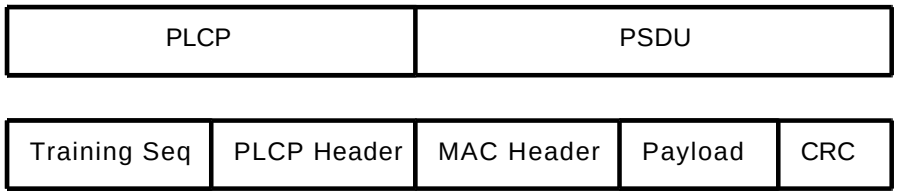

FIGURE 2.1: WLAN simplified packet structure

This event classification methodology requires the use of experimentation to validate the technique, since simulators cannot model this level of detail to generate wrong headers and to address a realistic recovery probability.

2.3.2 Capture Effect

Another impairments is the capture effect, which is the result of receiver lock into a more powerful signal during a collision. It has been shown by Kremo et al. [36] that the capture effect generates unfairness, but recovers one of the colliding packets, increasing the throughput of the highest SNR station.

Again, the capture effect cannot be simulated and experimentation is the only way available to take into account. Furthermore, the combination of packets of different rates and consequently different modulations, result in a complex effect which is not taken into account by the simple interference model from simulators.

2.3.3 Hidden and exposed station

Two other impairments are the hidden station and the exposed station. Both are covered extensively by Anastasi et al. [37] where they characterized the differences between wired and wireless networks, and then they focused on two features which illustrate these differences: The hidden station condition, as depicted on Figure 2.2, is the result of a pair of stations mutually out of range (A,C), trying to send a packet to third station which can hear both (B).

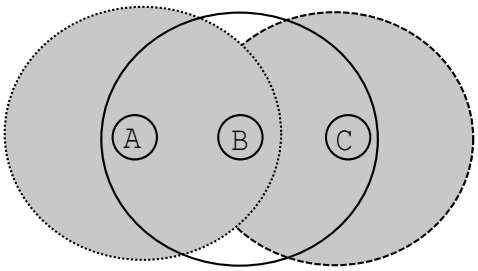

FIGURE 2.2: Hidden station condition

On the other side, the exposed station effect, as shown on Figure 2.3 happens when two pairs of stations (A,B) and (C,D) are arranged in such a way that A and C are in range of B, but D is not. Then, if during the transmission of a packet between B and A, C wants to send a packet to D, C listens that the medium is busy, and refrains from transmitting until the medium is free. The packet could have been sent without interfering A from receiving (since A does not hear C). The consequence is a throughput reduction. Anastasi et al investigate both effects and they conclude that the use RTS/CTS is not effective on 802.11b, since the overhead increase due to the RTS/CTS transmission at 2Mbps is higher than the time lost due to collisions. Nevertheless, their layout is different to the one used by to Kremo et al. [36], so they do not observe the capture effect.

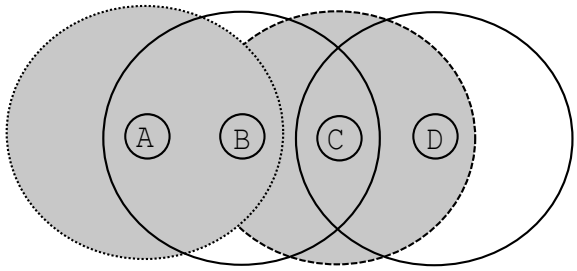

FIGURE 2.3: Exposed station condition

The main contribution from [37], is the identification of three ranges in agreement with the synchronization sequence rate, the PLCP rate and the data rate. These ranges, in decreasing size, determine the detection of the existence of a frame, the decoding of the header and the complete transmission of the frame.

Although the hidden station problem has been addressed by the 802.11 MAC, the CTS/RTS solution is not as effective as it was expected to be, and the difference arises from the different transmission rates and collision probabilities. On the other hand, the exposed station condition is highly dependable on the range, which is different between the PLCP header and the payload. Both effects constitute a simulation problem, since different rates are not supported, neither the corresponding different SNR ranges.

2.3.4 Collision vs. Noise/Interference

The last impairment is the inability of the MAC to identify a packet loss due to collision from a packet loss due to noise or interference. Rayanchu et Al. [39] have addressed this problem, trying to differentiate a collision from a weak signal. A collision signals a congestion problem, which is valuable information for TCP. Whereas a packet loss because of interference or noise under a non-congested environment is misleading for TCP, which reduces the rate to half without the need to do so. On the contrary, for rate adaptation algorithms the congestion is misleading, while the packet loss due to weak signal is valuable, since it flags the need of a rate change. They implement loss diagnosis, to analyze the source of packet loss, through bit-level error patterns after a packet loss.

The most remarkable item in [39] is the classification on the scatter plots for the identification of collisions, an effect which is not available at all on publicly available network simulators, so experimentation is required to improve the MAC layer with a new detection algorithm.

Finally, although the complete MAC mechanism can be simulated and analyzed, the MAC interaction with the physical layer is lacking several conditions which bias the

results and does not allow bit level or interference dependent experiments. Furthermore, variable transmission rates, their SNR sensibility, and the corresponding ranges are not present on the ns2 simulator.

2.4 Implementation bugs

Although ns2 has been widely deployed and used, bugs and implementation issues survived within the code for many years, possibly yielding to biases on the results, and furthermore, to wrong interpretations.

Schmidt-Eisenlohr et al. [40] have published a technical report with bug fixes on 802.11 for ns-2.28, where they rise several implementation problems for the MAC and PHY layers: they point out the abuse of the EIFS, the behavior face to a packet arrival during a transmission, the packet handling during the transmission, the expiration of the NAV and the capture effect implementation. Furthermore, Purushothaman et al. [44] note the weak implementation of the infrastructure mode, lacking beacon packets, and also the scanning, authentication, and association functions. Both the existence of bugs or incomplete implementation of the standard and its validation are two strong elements to encourage the use of experimentation to validate an algorithm or protocol. Although we expect the simulator evolution (specially ns2) in correctness and completeness, it is highly probable that more bugs still survive into the code. Thus keeping the doubt on the results from simulation, for the already published and future papers which are based on these models. Other comparative studies show the differences and biases generated by the simulator, and further propose solutions to this misbehavior. Ryu et al [42] point out the unreliability of results from the reception of packets for stations located between the carrier sense range and the transmission range. This generates an artificial unfairness effect, thus providing biased results. They propose bug fixes on the incorrect error frame assumption, and generate a correct BER model for ns2. Finally, Chen et al [43] describe and implement a new 802.11 structured and modular 802.11 model for ns2, to replace the original model, including cumulative SINR calculation, preamble and PLCP header processing and capture, and frame body capture. Even more, their model include the basic 802.11 CSMA/CA mechanism, which increases the credibility of simulations. Both solutions point to the need of a validated and more realistic modeling framework, a feature which will be taken into account on the brand new ns3 simulator.

To sum up, first, current simulation models include several simplifications reducing their accuracy compared to the experiment results. This is a consequence of the complexity of the wireless channel, whose accurate modelisation implies prohibitive computation costs. As a result, wireless experimentation is used for all those experiments which

depend either on the real channel variations or on the access to MAC and PHY level data, especially where simulations cannot help. But, there must exist a physical support specialized on the type of network and low layer access for each type of experiment, this is, an experimentation platform.

Nevertheless, the experimentation platform alone is not enough. In order to support experiments, instrumentation and measurement tools are required. Furthermore, measurements must be obtained systematically following a sequence of predefined steps so as to be able to reproduce and compare experiments under similar conditions. This last item is the main reason to create an experimental methodology so as to enhance wireless experiments.

Chapter 3

Wireless Experimentation Tools

3.1 Introduction

Since experimentation has an important role as a validation step for new protocols, in this chapter we survey tools available to enhance experimentation in a wireless environment. Network monitoring tools that capture wireless raw data and extract the results have been designed and implemented for this purpose. We focus on open source tools since they are the ones which allow users to check and understand the capture process. Ranging from the most basic tools used to capture the traffic and to establish aggregate measurements from network interfaces, to platform-specific tools which support the execution of experiments. We first classify the experimental platforms intended for wireless experiments and then we detail which are the tools, the steps of the experiment covered, and their classification. Finally, we sinthesize the missing links and the drawbacks of the current experimental sequence.

3.2 Experimental Platforms classification

Every experiment needs an appropiate platform for its execution, which represents or describes the intended scenario. In this section we classify the platforms with four different criteria: The PHY layer of the wireless device, the realism achieved by the platform, the communication standard and the mobility.

Most of the research in wireless networks is centered on improvements and new uses for the same bands and different evolving standards in MANETS, WLANS, WSNs and other topologies. Although we can account that 802.11 is not the only available MAC

to use the wireless environment, commercial availability, a non-licensed band and the constant evolution results in an increased enthusiasm from the research community.

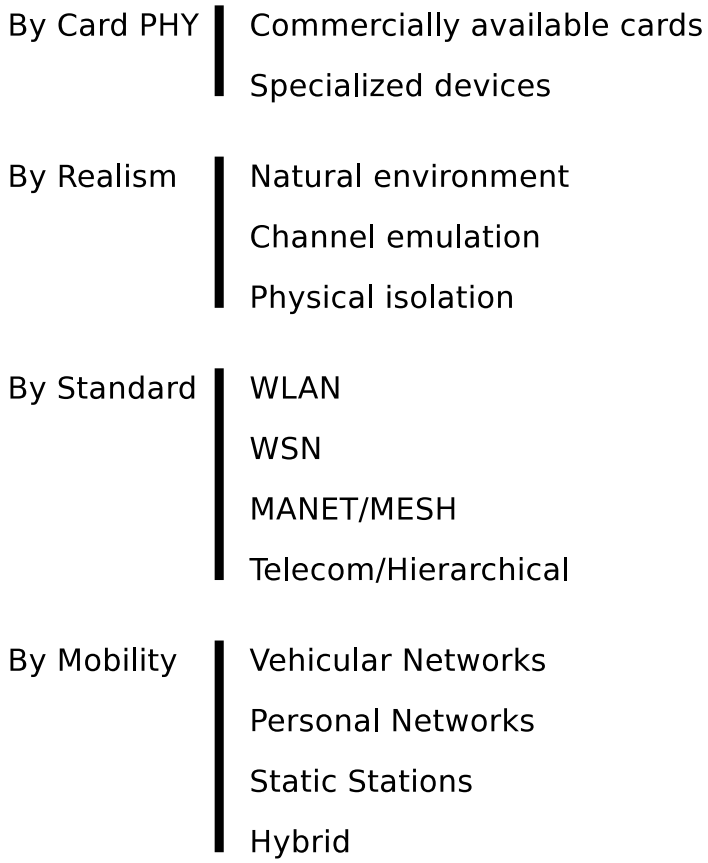

FIGURE 3.1: Platform classification by criteria

As we can observe on Figure 3.1, wireless platforms can be classified first by their objectives and inherent flexibility to exercise changes at the MAC and PHY level:

- One of the approaches for wireless experimentation platforms is to use commercially available cards. Although the MAC layer is standardized, its implementation (mainly due to time restrictions and commercial secrets) restricts to their experimentation availability. This is reflected on a mixed driver-level MAC administration with firmware-level time-critical functions, like ACK sending after a SIFS. Furthermore, wireless cards from different brands have their particularities:
  - There are non standardized features which are used to improve the performance of the card: like the rate adaptation, noise reduction algorithms and antenna selection algorithms.
  - There are vendor-specific non-standard implementations like Athero's turbo mode, which uses a wider bandwidth for data transmission,

  - And finally there are erroneus standard implementations, like backoff handling after a packet loss which changes the fairness between stations.

  The main advantage is the use of the real devices with their own biases. Since they reflect the behavior experimented by the user, while limiting the MAC and PHY experimentation possibilities to what is available (or made available via an NDA) to the researcher. Furthermore, a network using commercially available cards is easier and less expensive to build, while the support for the cards has a higher availability.

- The second approach for wireless platforms is the use of specialized devices, which are completely configurable and changeable at the MAC level, and which can be modified on many features at the PHY level. These devices, like the GNURADIO platform and the WARP project, give more freedom to the user as to prototype and test specific algorithms, while the configuration and programming is up to the researcher. The main advantage, as we have said, is the complete configurability of the device, while the monetary and development costs rise significatively. Another disadvantage is the use of a non-commercially available device, even if the device configuration emulates a real one, since the hardware and drivers for this case differ from standard user oriented implementations.

Wireless platforms can also be classified by the level of realism and controlability of the channel. We define realism as the level of closeness to a completely real environment using commercially available devices with a real traffic profile and composition. The closer the platform to the former conditions, the higher the realism we obtain.

- A first approach is to use the natural environment, the stations transmit sharing the medium with other stations and interfering sources, using an indoors, outdoors or mixed layout. The richness of the physical variations and the interfering traffic provides the most of the realism on the given physical structures and moving objects. We can say that this configuration is similar to a "wild" environment, while the results and variability will approach to reality. Although ideal of this platform is as realistic as possible, it would also be desirable to control or, at least characterize the environmental conditions. One of the control possibilities is to do the measurements while there is almost no movement of people around, like during the night. Another control method would be to stop the most wireless generating devices outside the group of experimentally-involved stations. In other words, experiment repeatability is proportional to environmental condition control.

- A second approach is to use a channel emulator (FPGA or otherwise based), for example, as Judd et al. describe on [90]. Channel emulators change wireless conditions artificially to simulate different kinds of environments with high repeatability. Each of the cards with external antennas are connected to a board that exchanges channel-level data with each other, emulating blocking objects, distance attenuation, fading, reflections and other channel effects. In this case, repeatability is achievable, scalability is limited to the realtime processing power for the channel emulator, and realism is less than the first case.

- A third approach is to use physical isolation within a limited space, like an anechoic chamber. While this is the most expensive of the conditions, it provides a higher flexibility and finer control on the test environment. Nevertheless, this kind of platform is intended to separate effects and measure the influence of materials and structures with respect to signal propagation, while it permits also enough isolation to execute antenna radiation pattern measurements and controlled interference tests.

On the other hand, Wireless platforms can also be classified with respect to the type of communications standard used. Among them:

- A WLAN testbed can be configured to respond to the different operating modes of the 802.11abg specification: Ad-hoc, infrastructure or bridging modes. Furthermore, a 802.11s is aimed for mesh networks and 802.11i for security, 802.11e for quality of service and others.

- A WSN testbed can use different kinds of technologies, like 802.15.4 and even including 802.11 nodes, while the MAC protocols on these type of networks are still a matter of research and discussion..

- A MANET/MESH network uses 802.16 or 802.11s (as said above) standards, while there are also many non-standardized protocols under development.

- An Telecom hierarchical network uses GSM, GPRS, UMTS (and future LTS) and other standards which respond to the typical devices used on commercial deployments for the Telecom Carriers and providers.

Finally, Wireless platforms can be classified by their mobility, depending on speed and trajectory:

- First, platforms aimed for vehicular networks, for fixed (e.g. buses/trains) or mostly uncontrolled (e.g. cars) trajectories, and variable speeds (from land vehicles to airplanes.

- Second, platforms aimed for personal networks, which use PDAs and wireless-enabled portable devices, targeted for human walking speeds.
- Third, platforms of static stations, which aim to test the behavior of a fixed layout with a moving environment.
- Hybrid platforms which combine any of the former.

In this thesis, we focus on commercially available, natural environment, WLAN and static platforms:

First, the platforms based on commercially available cards, are available for use as measurement instruments and as traffic generators. At the same time commercially available cards are low cost devices compared to the specialized instrumentation for RF and protocol analysis; this fact provides a wider range and a higher number of users than special instrumentation. Nevertheless, the use of commercially available cards limits the precision of the measurements on physical parameters like time and reception power.

Second, the natural environment provides the variability of the channel which captures the propagation conditions for a specific time and layout distribution. Although channel emulation and physical isolation increase the controllability of the experiment, they simplify and override the original channel characteristics.

Third, WLAN is the most accesible, widespread and standardized specification which does not require a special license to execute the experiments. Other communication standards require specific instrumentation, as 802.15.4 probes and licenses, as 3GPP.

The static case represents users which have wireless access from wireless-enabled devices like desktops and laptops, but not moving or portable devices. The environment is a combination of LOS/NLOS situations due to the people movement around the stations.

In order to reproduce an experiment, there must exist a group of controllable variables which are tuned within certain values or ranges to create conditions similar as the original experiment. All the other variables, which are not controllable contribute to the inherent variability of the environment and the traffic characteristics. Therefore, the degree of controllability of an experiment ranges from a completely uncontrolled environment, setup and traffic, where there is only observation and description; to a completely controlled environment, setup and traffic, where the inherent variability is minimized.

Thus, we can classify experimentation platforms from the point of view of the influence on the medium, as uncontrolled and controlled experiments are executed.

Uncontrolled experiments are those where the results are based exclusively on observation, without any kind of forced influence on the physical medium or traffic, as Henderson et al. [47] describe on the evolution of a large scale WLAN deployment at Dartmouth College, based on their first measurements, by Kotz et al. [50]. Within this publication, they describe the most important roaming and traffic characteristics of the wireless network under study.

Controlled experiments are those where there is a protocol, algorithm or phenomena under examination, and a specific traffic patern is injected to analyse the response.

Furthermore, a controlled experiment has several degrees of control. The Physical layer is the most elusive for controlling, since to the extreme would require total isolation and controlled interference generation. Work on this area has been developed by Judd et al. [90], where the wireless channel is emulated based on the modification of the signal characteristics using a dedicated FPGA for digital signal processing . MAC layer access for controlling purposes is as available as the wireless card vendor and drivers allow, for example, for Atheros-based cards under Linux, certain parameters (as the transmission rate, the ACK timeout or the output power) can be controlled as a standard feature. The upper layers (Layer3+) are fully controllable at the software level, like any other wired experiment. In this case, the only limitations to experimentation would be the operating system and the hardware processing power.

3.3 Current platform initiatives

Although most of the available platforms are customized for specific objectives, there are several general purpose experimental platforms to fulfill requirements at a higher scale or with paticular hybrid characteristics: wired and wireless together.

The GNURadio platform: The GNUradio platform is a data capture and generation system at the signal level. It consists on a specialized device which samples the wireless channel fast enough to be able to process the captured data with software radio algorithms on a standard CPU. To generate an output, the samples are created on software and sent to the Digital-to-analog converter output of the device. With this architecture, the system becomes the most flexible, to the expense of dedicated processing time from the host CPU and the constrained MAC timings and signalling. The main difference with a standard wireless board comes from the lack of realtime processing, due to processor sharing with the standard operating system. This does not happen if the MAC controller is independent and executed within a dedicated processor architecture, as many wireless cards currently do. We can classify GNURadio as a specialized, WLAN,

static and natural-environment platform. It was used by Gollakota et al [46] to obtain bit level information in order to decode packets during collisions in hidden-station situations.

The ORBIT testbed:

The ORBIT testbed, hosted at WINLAB, at Rutgers University, is a wireless experimentation platform which combines a matrix of wireless-enabled nodes with an "experiment deployment" application and data capture environment. The hardware is composed by 400 nodes uniformly distributed on a grid inside a hall. Each node has a CPU, atheros-based or intel-based wireless interfaces, a hard disk and an ethernet interface to build a support infrastructure network. The 400-node grid size is 21m x 24m, which makes every card in range with each other. This configuration can help with experiments related to interference between nodes, while a multihop configuration or a hidden station experiment are not possible. Since all the nodes are visible to each other, the whole platform can run only one experiment at a time, as a consequence, a reservation system is required to execute the experiment.

On the software side, the experiment scheduler is called OMF (Orbit Management Framework). There is a server which administrates the nodes via multicast of commands. This is executed via a node handler, which sends OS images to the nodes, enables execution of commands, monitoring and data capture during the experiment. The experiment execution is rule-based, meaning that although this increases flexibility, it also increases the complexity and monitoring; during and after the experiment.

The next step is the processing stage, executed by the Orbit Measurement Library (OML). It filters, compresses and collects measurements to the database. Within a more detailed publication, Singh et al. [36] reveal several features of the OML:

- The experiment configuration comprises the measurement points and parameters, which are configurable through a web interface. The results are stored in a XML file which describes which parameters should be collected within an application.
- The data is collected until a pre-programmed trigger is fired, for a certain number of samples or after a period of time.
- A filter is a calculation done over the captured parameters.
- All the rules and configurations defined on the XML file generate code which is compiled and sent to the corresponding stations.
- The data captured after the filters is organized within a database which is created automatically after the XML configuration file.

To resume. ORBIT is a dedicated platform for aggregated statistics of events or periodic sampling of physical parameters, like RSSI. Although raw packet capturing is possible, the application is not intended to capture nor merge this kind of data.

ORBIT can be defined as a standard device, WLAN, static, and semi-isolated platform; Kremo et Al. [36] have shown throughput and latency measurements on the ORBIT testbed, to demonstrate the utilization of the platform.

The roofnet experimental platform: Roofnet is an experimental platform, created at MIT CSAIL by Bicket et al [48] for medium-range point-to-point links. This platform was created within the Massachussets Institute of Technology and the surrounding areas. It was used as a testing platform for a mesh architecture, with the objective of providing high performance internet access while having little deployment planning or operational management. The platform itself was specific: to observe link quality variability with rate adaptation and with routing decisions, under real traffic conditions generated by students.

On the cited work from Bicket et al. [48], the authors consider an architecture of unplanned node placement, omnidirectional antennas and multi-hop routing. In this case, the network was composed of 37 nodes dispersed over four square kilometres. They have evaulated the performance of the mesh network through the effect of node density on connectivity and throughput; the characteristics of the links selected by the routing protocol, the usefulness of omnidirectional antennas for robustness and throughput, and the potential performance of a single-hop network using the same nodes as Roofnet. They have shown that the average throughput between nodes was 627kbps and that the mesh network increases connectivity and throughput compared to a single-hop network.

Roofnet can be classified as a standard device, MANET/MESH, static and natural environment platform.

The Emulab [49] platform: Although Emulab was created as an emulation platform, this definition corresponds to the wired side. Whereas for the wireless side it is a real deployment platform with APs dispersed on the same building. Their wireless section can be classified as a standard device, WLAN, static and natural environment platform. Emulab is a platform based on network emulation by means of the interconnection of virtual nodes through virtual links. On top of Emulab, there is the Experimentation Workbench, which support realistic, large-scale replayable research. This platform in the wired version allows people to move backward and forward through the experiment, manage activities (planned and unplanned) software, data and analysis. On each virtual node, the whole operating system, the network stack and the application run as independent machines, connecting to each other as if there was a real link between them.

The experimental conditions are recorded in such a way that the whole experiment has a high level of control and repeatability. This is acknowledged as replayable research, meaning that the experiments can be re-executed within the same controlled conditions and configuration. Emulab includes several experimentation features:

- the notion of experiment definition and instance,
- related experiments where certain controlled environmental conditions are kept constant while the others are varied one at a time,
- multiparty experiments,
- provides data management tools
- provides configuration, definition and control setup tools
- provides the storage of the experiment data

All these features are implemented through an online portal where the experiment is defined through an ns2-style script, it is controlled by the user and the configuration and captured data is registered through records. Although emulation is done on the wired networks with virtual machines, wireless experiments are executed on their real wireless nodes, which are part of the Emulab infrastructure.

We can observe that current experimental platforms are strongly attached to their management and experimental software. Although this fact is positive, to compare different experiences under similar conditions, the main consequence is a very small flexibility from the wide range of configurations and environments which can be achieved; and furthermore, the impossibility to use the same experimental framework on a local platform. Nevertheless, the platform can be decoupled from the management and experiment software in such a way that the only customization is the specific instrumentation. On the next section, we account for open source general purpose tools which, independently from any platform can be used as capture and processing applications.

3.4 General Purpose wireless measurement tools

Once we have described the specific platforms with their associated tools, we analize the individual general purpose tools which can be applied to ad-hoc platforms. As a matter of fact, this is a modular measurement and processing approach, to create a complete measurement chain with the most appropiate pre-existing tool and bridging the gap with custom-built modules.

3.4.1 Requirements for wireless measurements

First, we provide a structure to the measurent and processing chain to define the functionality of each of the modules. Then we describe each of the analized tools and we fit their functionalities within the proposed modules.

The most important source of information to analyze wireless networks are statistics inferred from packets sent on the channel. They are obtained using packet sniffers, which are stations that passively listen to all the packets on the medium. Furthermore, statistics provided by the drivers of wireless cards can be used as a complementary source of information, and the range of available statistics depends on the evolution driver's state. The choice of open-source drivers as madwifi [80], enables the instrumentation of the driver and also the sharing of the source code with other laboratories in order to validate the experiments. In the following, we describe the different steps involved in doing wireless measurements.

- **Sniffing:** The packet sniffing task consists in retrieving all frames transmitted on the wireless medium. It can be divided in two stages: The first one is the hardware side, which depends exclusively on the ability of the wireless cards to detect, decode, buffer and transfer the packets to the PC bus (either through PCI, PCMCIA, USB, PCI express or whichever standard communications bus is used). The second one is the software side, where the driver sends a copy of any received or transmitted packets to a part of the kernel called the packet filter. By default, all the packets are then copied from the kernel space to the user space where the sniffer is actually running. To create a sniffer, the driver of the wireless card must be configured in *monitor* (also called *promiscuous*) mode. Each sniffer generates an event log (or a packet trace) composed of all packets sniffed[1] on the wireless channel.

- **Merging:** Producing an accurate packet trace requires great care. In the wireless domain, spatial diversity prevents any single sniffer from capturing the overall traffic. Thus, many spatially dispersed sniffers are required to reconstruct all the traffic. Using too few packet sniffers, placing them poorly, or using inadequate hardware can miss or reorder packets with incorrect timestamps. When multiple sniffers are used, the independent traces have to be combined and synchronized down to microsecond granularity to construct a synchronized single trace of all frame transmissions [103–106].

[1]For obvious privacy reasons, the payload of packets should be discarded before building up packet traces.

- **Processing:** Once an accurate packet trace is ready, the actual processing can start. First, if some packets present on the trace are not relevant for the analysis, they can be filtered out from the packet trace to speed up the following computations. Then the parameters to analyze can be extracted, possibly combined with other sources of statistics (like the ones provided by the wireless card drivers). Various computations can follow, such as average calculation and the result can be displayed to the user.

- **Monitoring:** If the processing task is done in realtime, i.e. when sniffing and processing operations are done simultaneously, this is called *monitoring*. Monitoring is useful in general to analyze in realtime the network behavior, for example to observe the evolution of an experiment. A monitoring device is limited to the capture and realtime processing capability.

3.4.2 Tools for Wireless Measurements

After the description of the measurement steps, this section proposes a classification of the main open source tools which fulfills them. Wireless measurement tools come in three different flavors: Adaptations or derivatives from wired measurement tools, like Wireshark [119] and Mognet [117]; Monitoring tools specific to the wireless environment like Kismet [116], Wifiscanner [118] and Wavemon [112]; and tools which target experimental wireless measurements, like Airtraf [113], Jigsaw [105] or WisMon [120]. A tool is required for each of the tasks we have defined on section 3.4. Furthermore, some tools can be used for different purposes; for example, Wireshark can be used for capturing and to do filtering as a processing task. At the end of the section we provide a classification where each of the tool is assigned to one or multiple tasks to execute during an experiment.

In the following, we distinguish between tools that retrieve wireless parameters through the driver of wireless cards from tools that capture and/or process logs of packets sniffed from the wireless channel. We present only the most essential features of each tool.

3.4.2.1 Tools Based on Driver-Level Statistics

As we mentioned in Section 3.4, the drivers of wireless cards can be used as a complementary source of statistical information. The madwifi driver [80] is currently the most advanced open source driver available for Atheros-based wireless cards. It uses the Wireless Extensions for Linux and has companion applications to simplify the configuration and the statistical data capture. It allows to print the internal PHY and

MAC events to the system log through the athdebug and 80211debug utilities respectively [107]. Also, the driver keeps internal statistics which can be accessed through the athstats [108] utility for the PHY related statistics, and using the 80211stats utility for the MAC statistics. Although the madwifi driver itself is open source, it depends on the proprietary Hardware Abstraction Layer (HAL), which is only available in binary form. However, it is currently evolving to a full open source driver called ath5k, which does not depend on HAL.

Furthermore, some devices store these statistics in management information bases (MIBs). In this case, the statistics can be retrieved using Simple Network Management Protocol (SNMP) tools [135]. However, note that SNMP statistics can sometimes be inaccurate and should consequently be used with caution [109].

Wireless Tools for Linux: The Linux Wireless Extensions (WE) and the Wireless Tools (WT) [110] form a generic API allowing a Linux driver to expose to the user space configuration and WLAN statistics. This set of tools creates a uniform interface at the user space to configure, control, query and debug the wireless interfaces. A typical usage example of wireless tools is the access to the aggregate data statistics using iwconfig, the setting of specific driver-level parameters with iwpriv and the listing of the results of access points in range. They use a very basic textual interface and are included by default on most Linux kernels. However, different or invalid implementations of these utilities may lead to erroneous results, so these statistics should be compared with known statistics to guarantee that the results are coherent. The Wireless tools for Linux are distributed as open source code under the GNU Public Licence (GPL).

WaveMon: WaveMon [112] is an example of a wireless monitoring tool that uses the Wireless Extensions for Linux. It is a *ncurses*-based lightweight wireless monitor tool that can be run with or without a GUI. It supports devices with low processing power and low resolution display and allows to watch in realtime the signal and noise levels, packet statistics and wireless card configuration. Two different views are available: A snapshot of the current state of the wireless link statistics, and a historical graph for the same parameters. During an experiment, Wavemon can provide basic insight about the link quality from the signal power side. It can not be used for batch mode experiments because packet logging is not available; all the processing is done in realtime. Current version is 0.4.0b and it is distributed as source code under the GNU public license for the Linux operating system. A typical application of Wavemon is experiment monitoring.

WRAPI: WRAPI [111] is not a tool but a hardware-independent library that allows applications to access MAC-layer information. It requires an application on top of it to call the functions, and execute data capture. So it is not functional by itself, but it is very useful to perform measurements on a Windows platform, and, to our knowledge,

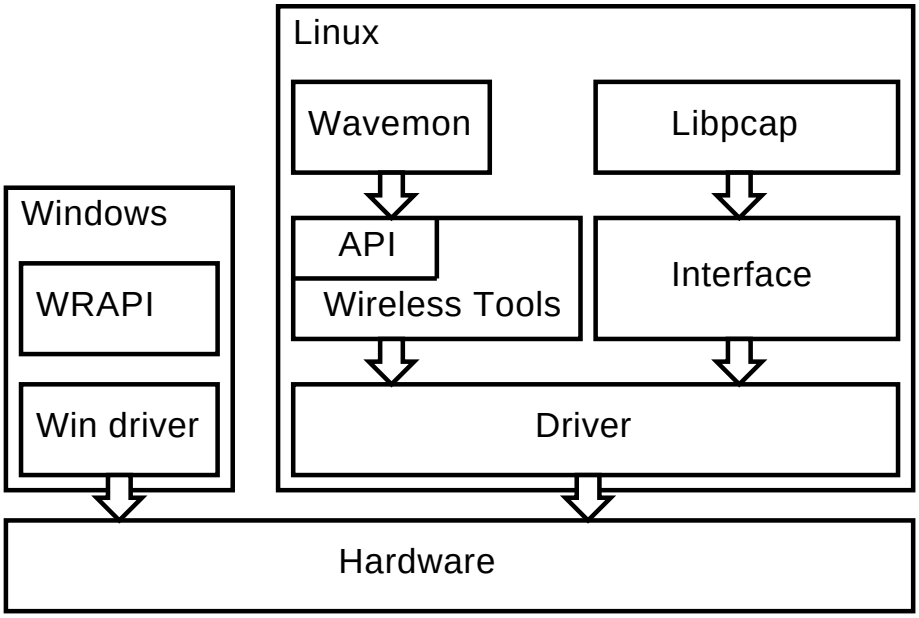

FIGURE 3.2: Location of Driver-Level statistics tools

it is the only open source package that works on Windows XP. WRAPI uses the NDIS user mode I/O protocol to communicate from the user-mode side to the driver. Using this communication protocol, it is possible to query information and set parameters. A limited number of parameters is available, including wireless configuration, packet level statistics, current MAC address, signal strength and AP information, but per-packet information is not available. It is worth mentioning that the statistical information provided by the driver for part of parameters is preprocessed internally, e.g. to calculate running average. As preprocessing operations affect the measurement results, the fact that these operations are not documented and the corresponding source code is not publicly available is problematic to perform rigorous analysis. This library was developed for Windows XP exclusively, and Version 2.0 of the source code is available on the WRAPI website [111] without any license information.

On Fig. 3.2 we can observe the location of the former tools within the Operating System.

3.4.2.2 Measurement Tools Based on Packet Traces

In this section we discuss relevant open source tools that can be used to perform the measurement tasks described in Section 3.4, i.e., sniffing, merging, processing and monitoring. To recapitulate, Table 3.2 gives a snapshot of the main functionalities of each tool.

Airtraf: Airtraf [113] is a multi-layer wireless analyzer that gathers accumulated wireless statistics from each station and for each TCP flow. It can be used to monitor

wireless statistics during an experiment, such as packet count, byte count, bandwidth usage and signal strength. It can also analyze the state of APs and the associations between stations and APs. Airtraf is sniffing-based and depends on underlying libraries to capture packets. There is no graphical interface available as open source for the polling server. Aitraf allows to create a "near realtime" picture of the wireless parameters, access points, stations present in the BSS and on TCP flows transmitted between them. This information is presented either as a cumulative (for packet or byte counts), mean value (e.g. power value) or instantaneous value (e.g. current bandwidth usage). Unfortunately, there is no historic log nor graphical user interface display to analyze the behavior of parameters. Furthermore, there is no event log to follow the transactions from the management packets. Version 1.1 of Airtraf for Linux is available under the GNU public license, but a commercial branch of this tool is available from Elixar Inc.

EasySnuffle: EasySnuffle [114] is a measurement tool composed of a collection of modules to insert on the kernel, device drivers and user space. These modules act as probes at MAC, IP and UDP layers and have been designed to analyze performance of multimedia transmissions over wireless networks. The basic setup includes a specific wireless device driver for the Prism2 chipset, a custom modified kernel and a user application. The main advantage of this tool is the possibility to instrument the system at different levels of the communication stack. However, the last version of the tool was released in 2002 and the modules lack flexibility to evolve to a newer version of the kernel. The source code is available for the Linux operating system at the Snuffle website [136]. Although this tool is rather outdated, we have included it as an example of instrumentation through all the measurement chain. Using this type of instrumentation, it is easier to do a cross layer analysis since all the data is captured on the same time basis.

Jigsaw: Jigsaw [105] is a multi-sniffer tool for infrastructure wireless systems that combines the packet traces to generate a comprehensive view of events taking place in the network. It is used to fulfill the sniffing and merging tasks. To synchronize the time across traces gathered by multiple sniffers, Jigsaw identifies frames that are overheard by multiple (but not all) monitors. It tackles the problem of clock drifts (the change in skew over time) using an exponentially weighted moving average of past skew measurements to predict future skew on a per-instance basis. Jigsaw can be used offline to gather and synchronize all the collected traces from an experiment. The output is composed of a single file including all the packets collected during the experiment. This tool allows to reconstruct a complete description of all link- and transport-layer conversations. Some inference techniques are used to deduce the presence, time placement, and even contents of missing data. Version 2.4 of this tool is available under the GPL licence from the Jigsaw website [115] for the Linux operating system.

Kismet: Kismet [116] is a passive monitoring tool to discover wireless networks. It creates a list of available access points in the selected channels and a list of attached stations using information contained on collected packets. For each item, it provides detailed information such as the addresses, traffic and station activity. The lightweight text-based interface allows the user to build a monitoring system without loading a graphical server. The main advantage of this tool lies in its ability to recover and gather in a smart way per-station traffic and fill station information structures with the results. This tool can also provide GPS information about scanned APs using the *gpsd* open source tool. Kismet's flexibility comes from the client-server architecture, built as a server processing engine and a client, interconnected with a proprietary protocol. The client (and possibly multiple clients) can be run on a different machine than the server. This tool does not provide any statistical analysis tools, although it can be used as a source of disassembled packets for data capture. Since Kismet includes a packet dissector, an application can be built on top of it to retrieve only the relevant parameters, which can be selected in realtime through the client-server communication protocol. Kismet runs on Linux with a large number of cards and can also run on Windows but only on cards that support AirPcap from CACE Technologies. It is licensed under the GPL and current version 2007-10-R1 is available at the Kismet website [116].

Mognet: Mognet [117] is a lightweight sniffer and analyzer tool inspired by Ethereal, but has been designed for personal digital assistants (PDAs) that support Java. It captures packets in realtime and disassembles the 802.11 headers using either a graphical or a text-based interface. Mognet can also generate output in libpcap format. The main objective of this tool is to display and disassemble packets with no further analysis, but with the possibility to examine packets collected in realtime. It does not include processing nor analysis functionalities because the processing power is still limited in PDAs. Since this tool can generate trace logs, it can be integrated within experiments where merging and postprocessing are done in batch mode. Although Mognet has not evolved since 2003, the current 1.16 version still works on any wireless cards that support the monitor mode. Mognet is available under the GNU public license and works on any PDA that includes a Java interpreter and a C compiler.

PCAP: The pcap library [79] is not a tool but since it is the core of many tools such as tcpdump[79] and Kismet, it is worth mentioning it in this section. Libcap is an open source library that provides a high level interface to network packet capture systems. The goal of this library is to create a platform-independent API to eliminate the need of system-dependant packet capture in each application, as every OS vendor may implements its own capture mechanisms. Different types of DLTs are specified in this library, including a variety of media like fiber optics, PPP serial links, ethernet and wireless networks.

Wifiscanner: Wifiscanner [118] is an analyzer/detector of IEEE 802.11b STAs and APs which can listen alternatively on all the 14 channels. It can be used to monitor parameters during an experiment and also for sniffing. Wifiscanner also includes an integrated IDS (Intrusion Detection System) to detect anomalies like MAC usurpation. A basic text-based interface is provided for passive sniffing operation. This tool provides the user a realtime packet disassembly of the 802.11 header. It also keeps cumulative values of the observed packet types at the MAC level. A list of current stations can be displayed with the number of transmitted packets. All network traffic can be saved in the pcap format for post analysis. Current version 1.02 is available under the GNU public license for Linux operating system [118].

Wireshark: Wireshark [119] (formerly Ethereal) is the de facto open source network protocol analyzer. It integrates a general-purpose packet sniffer and packet analyzer tool for almost any type of network. It includes a very powerful packet dissector and classifier tool and currently supports more than 700 different protocols. Specific filters can be built for each field of the captured packets. However, Wireshark lacks post-processing and analysis tools, providing only basic statistics and graphs. The main packet list uses coloring filters, which helps identify which type of packets are present. It also includes decryption support for many protocols, including IPsec, WEP and WPA/WPA2. Packet timestamps come from the pcap (called Winpcap for Windows) packet capture library, which is independent from Wireshark. Version 1.0.2 of Wireshark is available under the GNU GPL and runs on Windows, Linux, MAC OS X, Solaris, FreeBSD and NetBSD operating systems [119].

Wismon: In brief, Wismon [120] is a packet analyzer tool that has many functionalities for a wireless experimental usage. It provides physical parameters in realtime for evaluation during experiments and allows to record logs for further processing.

Wismon was developed within the framework of this thesis as the first approach to monitor the experiments on near-realtime. The first release of Wismon included several features which allowed the user to observe the behavior of the traffic and the physical parameters from the received packet information, and not from the calculated value offered by the wireless interface. This different approach transfers the raw values to the application to obtain more precise measurements, and gives the highest precision possible from the commercially available cards.

Wismon uses multiple probes to observe the wireless medium from different locations and merge the captured data into a single list of events. This event list is further analyzed as classified in order to extract a series of parameters which become time and space correlated, such as observed reception power: the same packet can be recognized from all the sources, and the reception power displayed as a function of space and

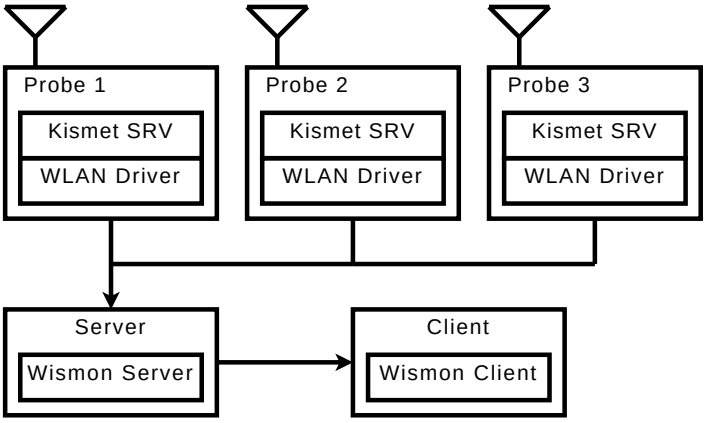

FIGURE 3.3: Modular structure of Wismon

time. Wismon is specially useful for monitoring wireless production networks to track interference, unfairness and packet loss problems, and also to monitor the evolution of wireless network experiments.

Two of the main features which differentiate Wismon from the other monitors are: First, the ability to synchronize packets in realtime, at the source of the packets, from a single reference. As packets arrive to the card at the probes, the driver synchronizes the reception timestamp with the one coming from the reference, for example an AP in range. Second, Wismon merges the packet list in realtime, inserting the packets within a round-robin linked list, and at the same time processes the list to obtain the required parameters. Each of the measured parameters themselves is calculated and stored in a round-robin list for a time window (e.g. 30 seconds) and then simultaneously stored to a log file for later reference.

As we can see on fig. 3.3, the modular structure of Wismon enables the expansion and evolution of the tool, while keeping each function independent and replaceable. The structure comprises a chain of modules: A realtime synchronization section at the driver level *[WLAN driver]*, achieved through a modified Atheros wireless driver; a capture section to provide an extract of the captured packets *[Kismet SRV]*, from a modified version of Kismet open source sniffer; a merging and postprocessing module *[Wismon Server]* and a user interface *[Wismon Client]*, developed for Wismon.

One of the main drawbacks of a realtime approach is the lack of scalability of the system in terms of the number of probes and the processed traffic. The hardware requirement to run Wextool is to provide a wired network of wireless probes to capture the data; but as the number of probes grows, the concentrated traffic on the Wismon processing module grows too. The station where the processing engine runs is limited in processor power and communication capabilities. Nevertheless, if the realtime capability is not

indispensable, multiple networks of wireless probes can be used, and the saved results merged and processed offline.

Finally several changes have been proposed to improve Wismon: First, to synchronize at the user level instead of at the wireless driver; Second, to replace Kismet as an intermediate layer with a specific preprocessing engine; Third, to add a web interface; Fourth, to use rrdtool [139] as a support for the processed parameter data, and finally an improved user interface in order to discover and represent the wireless nodes.

Version 0.1.R3 of this tool is available under the Cecill[2] license for the Linux operating system from the Wismon website [120].

Wit: Wit [106] is a MAC analyzer tool for IEEE 802.11 networks that includes a distributed passive sniffing mechanism. It collects traces obtained from multiple and independent passive sniffers and stores them in a common database. Wit can be used to perform the merging task of wireless experiments. Three processing steps are done to construct an enhanced trace of packets. First, a robust merging procedure combines the necessarily incomplete views from multiple, independent sniffers into a single, more complete trace of wireless activity. Next, an inference engine based on formal language methods is used to reconstruct packets that were not captured by any sniffer and to determine whether each packet was received by its destination. Finally, it derives network performance measures from this enhanced trace. Wit is available online [121] as perl scripts that process data traces.

CRAWDAD repository of tools: The CRAWDAD [122] (Community Resource for Archiving Wireless Data At Dartmouth) website contains a repository of basic scripts and tools to process packet logs and SNMP statistics. Various scripts can be used to extract fields and flags from individual packets, to create a list of stations from the packet logs, to anonymize the packet traffic (in order to be published later in the public domain), and to estimate the location of devices. Most of these tools and scripts tools are available under the GNU public license (or some variation of it) for the Linux OS.

Snapshot of Tools Based on Packet Traces: Table 3.1 shows the input and output formats, and Table 3.2 gives a snapshot of the main characteristics for each of the tools described above.

[2] http://www.cecill.info/

Input Method/Format			Tool	Output format				
Pkt Capture	Pcap	W. Tools API		TUI	GUI	Text	Pcap	DB
			Airtraf					
			Easysnuffle					
			Jigsaw					
			Kismet					
			Mognet					
			Wavemon					
			Wifiscanner					
			Wireshark					
			Wismon					
			Wit					

Table 3.1: Input and Output formats for each of the tools

Table 3.2: Wireless measurement tools

Tool	Sniffing	Merging	Processing	Monitoring	Comments	Platform
Airtraf	No	No	No	Throughput, Text-based interface	New version became commercial, including the web interface	Linux
EasySnuffle	Single probe	No	Statistics extraction	No	Driver and kernel instrumentation	Linux
Jigsaw	Multiple probes	real-time merging	Filtering, Traffic reconstruction	No	To sync, uses exponentially weighted of past skew measurements	Linux
Kismet	Multiple probes	No	No	Throughput, Text-based interface	Very popular and advanced wardriving tool	Linux and Windows (restricted)
Mognet	Single probe	No	Filtering	No	Java based, targeted for portable devices	Multiplatform (tool mostly written in Java)
Wavemon	No	No	No	Statistics calculation, Text-based interface	No logging	Linux
Wifiscanner	No	No	No	Network structure graph generation	Wardriving oriented	Linux
Wireshark	Single probe	No	Filtering and flow analyzer	No	Very extensive protocol decoding	Multiplatform
WisMon	Multiple probes	real-time merging	No	Data classification per source, recent history buffer	Packet logging and offline analysis	Linux
Wit	No	offline merging	Filtering, Traffic reconstruction	No	Has a formal language to describe conversations between hosts	Multiplatform (tool mostly written in Perl)

We can observe from table 3.2 that no tool provides all the required functionalities to develop a complete experiment, while from table 3.1 many combinations of tools are not possible due to a format incompatibility. Both disadvantages reflect the need to integrate and standardize wireless measurement techniques, through the construction of the missing parts and the integration of the existing tools if possible. Nevertheless, the resulting system should not repeat the same pitfalls: It must be modular and it must have a single interface format for the data.

A very recent example of an integrated toolkit, although aimed for MANETs, and not WLANs, is the EXC [92] toolkit. EXC is worth mentioning as a further step towards an experimental scheduling tool. It is capable of scheduling experiments in a semiautomatic way, with multiple experiment instances and provides a platform-independent solution.

This toolkit is complemented with a trace analysis package called EDAT, which uses a standard pcap file as input. The file processing is designed in a GUI as a pipeline of operators over the incoming data flow, which further generates a script describing the filtering sequence.

EXC and EDAT are tools which provide an integrated experimentation toolkit, with scripting support, to define an MANET experiment and to process the resulting data through a scripting engine. This implementation of a measurement methodology has the advantage of the independence of the platform for scheduling purposes, while it still relies heavily on a local server-based application to administrate all the involved nodes, and furthermore the processing is done directly on pcap files.

3.5 Conclusion

Unlike simulation, where most of the parameters can be controlled or changed with ease, wireless experimentation depends both on the instrumentation, measurement and processing of the captured data and the control of certain parameters.

For a simulation, the assumptions are the most noticeable limit, since the simplification of reality through a model must represent accurately enough the behavior we wish to test. Specifically, the level of detail increases the complexity of the simulation, thus requiring more computing power and more processing time to achieve. The balance between these two elements is the responsibility of the researcher who will solve the compromise between accuracy and processing time.

On the other hand, the experiment results depend on the measurement accuracy and instrumentation capabilities, since it is not always possible to have specialized instruments

available and to access all the functionalities of a wireless card. Then, indirect measurements and calculations will be required to obtain the results, through the process of the captured data.

For both simulation and experimentation the execution of multiple runs provides statistical data. In the first case, variability is generated by the use of a different random seed, while in the second case, multiple experimental runs are executed to arrive at an average value. In brief, in a simulation, order and reproductibility are natural, then random must be generated; in experimentation random is natural, and control to achieve reproductibility of the experiment is the main task.

Simulation has the main advantage that any interference generation is controlled and that the medium is as clean as the user requires, while during experimentation, the environment is the first uncontrollable element, and interference must be recorded to understand the deviations from the expected behavior.

Wireless experimentation available tools can be divided into three cathegories: the platform specific tools, which provide the control and measurement over a predefined platform; the general purpose tools, which invidually provide part of the requirements to execute experiments, and the most recent integrated experimentation toolkit, which provide control over a user defined platform and data processing on the obtained results.

This classification also shows an evolution which take into account two aspects: First, the recognition by the community that wireless experimentation requires different platforms and environments, so that there is no one platform solution; and Second, the creation and evolution of integrated tools which start to consider the experiment as a whole and not as individual tasks. Nevertheless, tools currently available do not fulfill integration, common interface, modularity and processing requirements; as a consequence, wireless experimental tools need futher enhancements and moreover, they need an underlying methodology and structure.

Chapter 4

Wireless Experiment Control

4.1 Introduction

In this chapter we move forward to define the experiment control steps and we identify the underlying requirements for wireless experiments. First in sections 4.2 and 4.3, we define which are the influential factors which impact wireless measurements, so as to define the environment and layout where the experiment is executed. Second, in section 4.3.4 we look over the design of the experiment, which includes the purpose of the wireless experiment, the selected metrics and the related raw measured data. Third, in section 4.4, we define the experimentation sequence after the design step, so as to enable the reproduction of the experiments under similar conditions. Finally, on section 4.5 we define the required protocols and aggregate data values to achieve the measurement.

4.2 Existing proposals for experiment control

While wireless measurements are generally associated with published packet logs captured on a specific site without any control, a wireless experiment is based on the effort to control the maximum of parameters available in order to identify their influence over the results.

Any passive capture of traffic and statistical information during a certain period represents what has been seen in reality, and provides information to analyze the behavior of traffic and users for that specific situation.

On the other hand, to provide an experimental validation of a protocol or algorithm, the experiment must take into account the two meanings of the control word: First,

to control the parameters and configuration of the devices in order to generate the correct test conditions, and on the other hand, to execute a control measurement of the environment variations without the influence of the protocol itself, in order to extract the experimental conditions, in order to ask the question: "What would happen if the protocol/algorithm was not there?"

In the following we present, first, two representative efforts to specify and execute wireless measurements from a scientific point of view, and second, the GENI (Global Environment for Network Innovation) initiative, which proposes an heterogeneous and flexible approach to test new internet architectures.

Portoles-Comeras et Al. [32] show the influence of single or aggregated throughput of multiple radios devices for experimental measurements. Their results show that there is no difference between the use of a single or a double radio device up to a workload limit. After this limit is crossed, the performance drops for a double radio device. They have also correlated this limit to the choice of wireless hardware components used during the tests. They emphasize the need of proper calibration of tools and devices before the tests using active and passive measurements, in order to obtain valid results.

A detailed guide to evaluate performance of wireless devices through a series of parameters is the "Draft for the recommended practice for the evaluation of 802.11 performance", defined as 802.11.2 D1.0, version of April 2007. It comprises a collection of experiments and detailed measurement techniques under controlled conditions for each parameter, but most of the procedures require specialized instrumentation which is beyond the equipments available on a standard wireless network platform. Furthermore, this guide is aimed at the comparison of wireless devices through a common set of standardized tests. This draft, although it was never approved as a final document, is a valuable guide to execute experiments associated with specific radio frequency tools. The innovative element from this draft would have been the creation standard tests at component and application level while fixing channel and MAC conditions. Nevertheless, the specification and experimental methodology are still not normalized nor standardized, and the result is the variable level of details which can be obtained from the experimental validations published to date.

An important effort towards a general platform and experimental specification is GENI. It is defined as "a set of components including optical substrates, forwarders, storage, process clusters, sensor fields, and wireless regions combined with a software management framework.". This program is composed of two parts: the *research program*, which supports network research and experimentation, and the *research facility*, for the exploration and evaluation of scaled architectures under realistic conditions. The GENI [33] architecture is a heterogeneous construction of wired and wireless networks. The

wireless side of GENI will be provided by a PEN (Programmable Edge Node) connected to several PWN (Programmable Wireless Nodes), not attached to any protocol or modulation, so as to enable the experimentation with innovative wireless proposals. Nevertheless, although GENI has evolved from the original 2006 project with resource allocation with proposed virtualization, slicing, and programmability features, there are still no provisions for the experimentation measurement techniques. Furthermore, there are no specifications for the associated experimental methodology, neither for a prospective common storage and processing standardization to extract results.

We observe that the work of Comeras et al is oriented to hardware issues and correlations with experimental results, while the 802.11.2 recommended practice would have been a practical guide to be used as a base for wireless experimentation on 802.11 WLANs. Nevertheless, neither of the former proposals propose a structured path to integrate the measurement methods, nor an implementation of a tool arises from them.

We believe that these publications are only the starting point to establish a common standard to enhance wireless experimental practices, taking into account a combined physical and networking approach.

4.3 Experimentation

Wireless LAN experiments become variable and unpredictable if Physical considerations are not taken into account. The electromagnetic propagation characteristics deeply modify the behavior of a received signal from one physical distribution of devices to another. Rough classifications as Indoor and Outdoor represent specific characteristics of the environment where the experiments are executed, but the infinite possible configurations which can influence the wireless transmissions cannot be synthesized by either of these words.

Since our aim is to improve the control at the experimental phase, we provide a set of considerations to improve the control abilities at the experimental stage. Nevertheless, the physical and logical platform limits the parameter controllability (like noise and interference) and the precision of the measurement (like sampling granularity of parameters).

4.3.1 Spatial considerations

If we think about a photo of the state of the experimental platform before the start of the experiment, we can observe a group of items to take into account:

- The first consideration is the position, in three dimensions. The freespace attenuation predicts the value of the signal at a certain distance from the source. This is an indicative value which must be registered to describe the environment. All the positions must be recorded: the location of stations, obstacles and any other reflective/dispersing surface. Furthermore, reflected signals travel a longer path than direct signals, then they can distort the original information when they arrive together at the receiving antenna. A dispersed signal contributes to the increase of noise and consequently, the decrease of SNR at the receiver. Position also captures the effects of the different radiation patterns from the antennas, which greatly affect the perceived power level at the receiver.

- The second consideration is orientation. The radiation pattern is not ideally isotropic, so the orientation of the devices change the received power value.

- The third consideration is the obstacles which are present between the devices, which can attenuate or reflect the radio signals. A detailed description of their dimensions, position, orientation is required. If the construction materials are available, this is desirable to differentiate the attenuation factor between obstacles and walls.

4.3.2 Temporal considerations

Now, if we imagine that during the experiment there is movement, then this movement must be characterized too. The movement can be:

- Internal, meaning that the same stations are changing positions in a restricted area, and stations can arrive or leave during the experiment. The path and the speed which of the (now mobile) stations must be characterized with enough detail in order to increase the reproducibility of the experimental conditions.

- External, meaning that the obstacles (like people or objects) move during the experiment with respect to the fixed stations. Although this situation is more complex to characterize, the transit of people generates a NLOS situations which increase the variability of the wireless channel. This variability can be measured through the use of the Ricean K factor as a way to characterize and compare different situations [102]. Office hours imply generally movement of people around, while weekend or night hours represent a scarce presence of moving objects in office environments. But these timetables vary when other environments are considered, like moving people in airports, homes, shopping centers, streets, public transports, bars and restaurants have different patterns from the office. Outdoor moving

elements like vehicles or special situations such as tunnels and subways must be characterized since they also represent sources of channel variability.

- Mixed: Internal and external combined, which would represent the most general and less controllable case.

4.3.3 Topology and Software considerations

Wireless network has an additional element to modify the traditional topology concept, which is range. Although the use of an interface for a certain flow can be controlled, there will not be any possible connection if the receiver is not in range of the transmitter or if the SNR is not enough to allow any data transfer. Because of the same fact, wireless links are temporal and variable due to movement and channel variations.

The requirement to define topology is limited then to the definition of the nodes, their interfaces and their physical configuration for transmission: transmission power, single or multiple antennas, rate adaptation algorithm, channel, sensitivity specification and use of vendor-specific transmission improvement algorithms.

The software which is running each of the nodes defines the behavior at the different levels:

- The wireless card driver imposes the relationship between the hardware and firmware from the card to the rest of the MAC layer, which can be implemented inside the driver or at the OS level.
- The Operating System is a main factor which limits the efficiency of the link while it manages the rest of the networking stack.
- and finally, the applications generate traffic and interact with the other lower layers; their configuration parameters define the source of the testing conditions of the experiment.

4.3.4 Experiment design

After we have reviewed the experimental considerations, the experiment itself must be designed given the controlled conditions and the protocol or algorithm under test. Here, the implementation of the protocol/algorithm or a close representation of its behavior, becomes part of the wireless experiment: The development of the communication modules and applications influence the performance of the results, which not only gives

feedback to improve the protocol/algorithm itself, but also arises the fact that a different implementation of the same algorithm can result in a strong gap on the expected results.

The first step to design an experiment is to establish the *functional objective* of the experiment: The validation of a protocol or an algorithm. To achieve this purpose, the behavior and performance of the protocol or algorithm is evaluated from a known starting condition through a series of evolving traffic and environmental changes. Furthermore, similar protocols or algorithms can be evaluated under the same experimental conditions in order to compare their performance.

The *operational objective* of an experiment is to provide measurements which represent the most accurate state and evolution for the desired parameters. To achieve this objective, during the experiment, both the events and the cumulative data must be captured. The events correspond to packet transmission and reception, plus any of the MAC layer state changes which may be available from the wireless driver. The aggregated data shows a snapshot of several types of parameters as a cumulative value, e.g. the number of packets dropped at the multicast queue, the number of packets with physical errors, the number of retransmitted packets.

The common practice to validate through controlled experimentation can be described through a series of steps:

- The creation and deployment of an experimental platform,
- The setup to execute the experiment
- The experiment itself, with the variation of the parameters
- The storage of the captured data
- The processing of the data, depending on the expected parameters to measure
- The construction of the graphs and tables to display the results

The design of the experiment responds to the experiment measurement objective and expected results. For example if an experiment is aimed to demonstrate unfairness between TCP flows for different bit rates, then the throughput on time for each flow is the measurement objective. To achieve the experiment, a group of different tests is designed. To continue with our example, it would be one test for each concurrent flow bit rate. For each of the tests, and to assure a statistical validation, a group of runs is executed, so as to study the variability of the uncontrollable parameters between runs for the same test. This relationship can be observed on fig. 4.1.

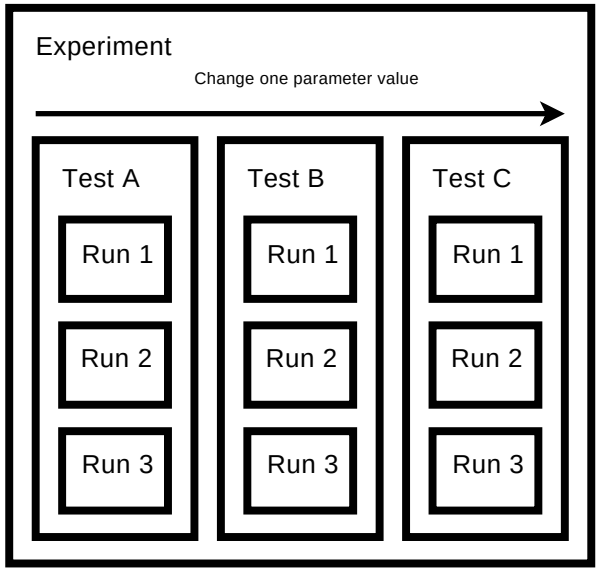

FIGURE 4.1: Relationship between Experiment, Test and Run

While it is fundamental to have a sequence of steps to follow during an experiment, the former sequence must be part of a well-known method to execute experiments and extract results. The common practice on this stage has several pitfalls, like the one-time and never again configuration, (hardware is reused for another purposes, the stations change place or they are simply removed), the lack of storage for the captured data, which may rise to many gigabytes, a one-time and poorly documented filter, written without parsing optimization, and a difficult or impossible synchronization between event logs and aggregated data. All the former factors contribute to the nearly impossible reproduction of similar conditions for experimentation, since they are incomplete, unavailable or simply lost.

4.3.5 Metrics

The design of an experiment not only specifies the objective, but how to extract a metric which represents the improvement of the protocol or algorithm under test. This means not only the definition of the metric itself, (e.g. the quantity of bits per second by source) but the formula or process to extract the values from the measurement, plus the representation method, which by means of interpolation or different scales enhances the appreciation of the results.

Here we briefly describe two types of statistical parameters which can be extracted after the processing of an experiment.

Per-flow statistical parameters are essential to analyze the performance of applications on top of IEEE 802.11 wireless networks, or to study the fairness between the different stations. They are general industry-accepted statistical parameters obtained by processing packet traces sniffed on the wireless medium, i.e., by analyzing packet presence, packet headers and arrival time.

Table 4.1 presents a non-exhaustive list of per-flow statistical parameters with a use case example for each of them.

Table 4.1: Per-Flow Statistical Parameters

Parameter	Description
Goodput	Measures the packet arrival rate during a fixed period of time at the application level. It can be used to evaluate the quality perceived by the application.
Data loss rate	Measures the number of data frames lost during a period of time, it is the inverse of the data delivery ratio. The loss can be due to transmission errors (e.g. noise, interference), buffer overflow or collisions. It can be used to estimate the link quality, the quality perceived by the application or as feedback to adaptive wireless-aware protocols.
Data loss burstiness	Measures the number of packets which are lost consecutively. This parameter can be used to measure the quality of a link or to tune the error correction algorithms [123].
Delay	Measures the latency at the application layer for a frame from departure at the transmitter to the arrival at the receiver. The delay is a consequence of queuing, packet retransmissions and packet transmission on the medium. It can be used to estimate the channel load or to evaluate performance of real time applications.
Jitter	Estimate of the statistical variance of the data packet inter-arrival time. High variability may harm the protocol stability and performance; It can be the result of packet loss bursts, after successive retransmissions. Same use cases as for the delay.
Airtime	Stands for the effective transmission time on the medium. This parameter is used as a fairness metric between the flows that share the same wireless channel.

Continued on next page

Table 4.1 – continued from previous page

Parameter	Description
Retransmission probability	It is function of the packet loss rate observed for both data and acknowledgement transmission. It can be used to estimate the link quality.

Per-Station Statistical Parameters: Each station may have several active flows at the same time. It is sometimes useful to analyze the performance of all flows that belong to the same station. Per-station statistical parameters correspond to the aggregation of per-flow statistical parameters detailed in the previous subsection, plus statistical parameters including:

- active flag: true if the station has at least one flow running,
- association state: associated or disassociated,
- association duration,
- MAC address of the current associated AP,
- Current Received Signal Strength Indication (RSSI) [124, 125],
- Current uplink physical transmission rate.

4.3.6 Per-BSS Statistical Parameters

Per-BSS statistical parameters include the aggregation of per-station wireless parameters, and other statistics such as the parameters described in Table 4.2. These wireless parameters are used to monitor and to analyze the characteristics of the whole channel.

Table 4.2: Per-BSS Statistical Parameters

Parameter	Description
Channel capacity	Theoretical maximum traffic rate at the physical layer. For example, it is equal to 11Mbps for IEEE 802.11b and to 54Mbps for IEEE 802.11a.
Overall data throughput	Aggregated throughput of all data packets transmitted on the medium. It can be used to estimate the channel load.

Continued on next page

Table 4.2 – continued from previous page

Parameter	Description
Overall signaling throughput	Aggregated throughput of management and control frames (such as beacon, RTS/CTS/ACK) of all packets transmitted on the medium. It can be used to compute the channel overhead.
Packet loss spatial correlation	Identifies the packet loss related to the position relative to the AP and the other attached stations. This information is available from the correlation between packet logs (source and receiver probes) and from the wireless driver statistics logs (packets lost at the sending queues are not considered). It can be used to analyze and improve the performance of multicast transmission protocols [126, 127].
Load level	Number of packets present in the wireless medium per time unit. It indicates the level of medium usage.
Available bandwidth	Corresponds to the rate at which a new flow can send traffic without affecting competing flows. Different algorithms have been proposed to estimate the available resources, see [128, 129, 131].

4.4 Experimentation sequence

Once the design and the metrics have been established, we examine the time dimension of the experiment, which involves the traffic and the tasks to execute on each of the participating stations. To achieve this, the design of the steps to follow during the experiment is essential. We suppose that the space dimension is already defined, which involves the description of the setup, nodes and links.

Since the experiments require the synchronization of the events, there must be a provision to start the tasks in the correct sequence according to the current experiment design.

Furthermore, the mode of the stations must be defined according to the 802.11 specification, this is, Access Point, Station, Ad-Hoc and Monitor, so as to preconfigure them and assign the correct task; for example one Access Point node and two Station nodes with four Monitor mode nodes at different distances, then, the Monitor nodes will observe the traffic from different points of view to capture most of the data exchange.

The tasks that exchange data between them must be configured to send the packets through the available wireless interfaces on the nodes, with certain instrumentation to capture the defined parameters.

The former facts show the need to build a structure with several abstraction levels to describe the experiment and their interaction. The topology description provides the nodes themselves, their modes and their interfaces, while a detailed schedule provides the traffic behavior which will be created during the experiment.

Finally, the description of the setup and the sequence shall be enough to recreate the same experiment later, under similar conditions in order to enable the reproduction of the experiment.

4.5 What to measure?

Wireless experiments may expect results up to any layer within the protocol stack, as any other kind of communications experiment, while the most relevant interest lies on PHY and MAC layers themselves and their cross-layer interactions with higher level protocols such as TCP/IP, ICMP, FTP and HTTP. In this section we focus on the wireless protocol elements which represent the difference to the wired counterparts, while the higher layer protocols remain common.

The choice of target protocols to study is tightly related to the tasks running during the experiment and with the analysis which follows the capture stage. Depending on the experiment length and amount of traffic, the capture and storage capacities may be compromised due to the high amount of generated data. A smart configuration of the capture devices and storage system in order to transmit and store only the relevant part of the captured data leads to a more efficient processing stage and smaller logs.

Here we account for a brief description of what can be obtained from the PHY layer and the MAC 802.11 layers while the execution of an experiment.

802.11 MAC Layer: The 802.11 MAC layer aims to provide access control functions to the wireless medium such as access coordination, addressing or frame check sequence generation. Two different classes of wireless configuration have been defined for 802.11: The infrastructure network, where many stations (STAs) can communicate with the wired backbone through an access point (AP), and the ad hoc network, where any device can communicate directly with other devices, without any connectivity to the wired backbone. In infrastructure mode, an AP works as an authentication and network association device, and can act as a bridge with other networks. A group of STAs

coordinated by 802.11 MAC functions is called a basic service set (BSS) in infrastructure mode and independent BSS (IBSS) in ad hoc mode, respectively. The IEEE 802.11 MAC sub-layer defines two medium access coordination functions, the basic Distributed Coordination Function (DCF) and an optional mode called Point Coordination Function (PCF), which is unused in practice.

DCF is an asynchronous transmission mode based on Carrier Sense Multiple Access with Collision Avoidance scheme (CSMA/CA). Collision detection can not be implemented because, due to the nature of the channel, a station is not able to transmit and listen at the same time [132]. Actually, two different carrier sensing mechanisms are used: PHY carrier sensing at the air interface and virtual carrier sensing at the MAC layer. PHY carrier sensing detects the presence of other STAs by analyzing all packets received from other STAs. Virtual carrier sensing is optionally used by a station to inform all other stations in the same BSS (or IBSS) how long the channel will be reserved for its frame transmission. The sender can set a duration field in the MAC header of data frames, or in the Request-To-Send (RTS) and Clear-To-Send (CTS) control frames. Then, other stations will update their local timers of network allocation vectors (NAVs) to take into account this duration. The RTS/CTS mode is often used to reduce collisions in presence of hidden nodes [130].

IEEE 802.11e-2005 (or 802.11e) is an amendment to the IEEE 802.11 standard that defines a set of quality of service (QoS) enhancements for WLAN applications through modifications to the MAC layer, and has been incorporated into the IEEE 802.11-2007 specification. In order to provide queue based QoS support, a new MAC layer coordination function has been proposed, called hybrid coordination function (HCF). Within the HCF, two interoperable methods of channel access are defined: HCF Controlled Channel Access (HCCA) and Enhanced Distributed Channel Access (EDCA). HCCA is based on polling, while EDCA is based on a slotted and highly parametric CSMA/CA protocol. The 802.11e amendment is important for delay-sensitive applications, such as voice over wireless IP and multimedia streaming.

Structure of 802.11 Frames:

As we have mentioned earlier in section 2.3, all of the 802.11 frames share the same basic PHY level structure: a preamble to train the receiver followed by a Start of Frame (SOF) delimiter; a Physical Layer Convergence Procedure (PLCP) header and the payload called the MAC Protocol Data Unit (MPDU). The PLCP carries the signal field containing the payload data rate, a service field describing modulation characteristics, the length of the payload in microseconds and the Cyclic Redundancy Check (CRC) of the PLCP header. The preamble and the PLCP header are transmitted at 1Mbps

regardless of the current data transmission speed. Three types of MPDU exist: Management, Control or Data.

Wireless parameters are essential to monitor the network, to detect possible anomalies or to analyze deeply the behavior of network protocols. A very large number of wireless parameters exist at different levels of the protocol stack, and are available either from the wireless card drivers or through packet traces. In this section, we propose a classification of most common wireless parameters that distinguishes between per-packet, per-flow, per-station and per-BSS wireless parameters.

Per-packet wireless parameters are those included in PHY and MAC headers of each packet and are available through packet sniffing. The following two subsections detail the most important PHY and MAC information present in IEEE 802.11 frames sniffed from the wireless medium.

PHY data information: Every 802.11 frame starts with a training sequence, a Start Of Frame (SOF) marker and a PLCP header. When a sniffer captures a frame, the three of them are removed on the hardware interface before the frame arrives to the driver, and an artificial header, called DLT (Data Link Type), is added instead. Different formats of DLT headers exist and they are defined in the pcap [79] library. DLT headers include PHY-level information captured by the network interface card. Note that if some PHY-level parameters are not supported, the user still has the possibility to add the missing features himself. However, such changes require OS kernel development skills and assume that sufficient chipset documentation is available.

The different types of DLTs specified in the pcap library include a variety of media like fiber optics, PPP serial links, ethernet and wireless networks. For 802.11 networks, the most popular ones are AVS[1] [137], PRISM[2] and radiotap [133]. These three DLT types share some common parameters shown in Table 4.3. We focus on the radiotap header type because it provides more features than the other header types. The radiotap header type consists of a standard preamble followed by an extensible bitmap indicating the presence of optional capture fields packed into the header as compactly as possible. This typically includes information such as rate, channel number, signal at per-packet basis RSSI (Receive Signal Strength Indicator measured for the PLCP header by the circuitry on the wireless network interface card [125]) and timing information (a 64-bit field in microseconds indicating when the first bit of the MPDU arrives at the MAC).

[1] AVS: (AbsoluteValue Systems, Inc).

[2] Originally designed by Intersil Inc., PRISM Wireless LAN business is now part of Conexant Systems Inc.

Table 4.3: Common DLT header values

Parameter	Description
Timestamp	Packet arrival time. At the radiotap header, there is the MAC timestamp, which corresponds to the arrival of the first bit of the packet. libpcap [79] adds another timestamp extracted from the system clock, after the packet has arrived to the kernel level.
RSSI	Received Signal Strength Indication. Measures the average receiving power of the packet. The RSSI value and bounds varies between implementations [125].
Channel	Channel number where the packet was transmitted.
Noise level	Characterizes the background noise level measured before packet reception on the channel at the receiver.
Data Rate	Physical bit rate of the packet payload.
Preamble	PLCP preamble length (short or long).

4.6 Conclusion

In this chapter we have described in detail which are the requirements to fulfill in order to describe and execute wireless experimentation to improve the ability to reproduce experiments. Although wireless experimentation platforms can be created with very different criteria, the underlying physical phenomena of electromagnetic propagation and the interaction with the MAC layer remain to be the key factors in the limit between controllability, reproducibility and environmental influence.

Chapter 5

Wextool: A tool for experiment control

In this chapter we present the details of an experiment control plan that we propose. First, we describe the new model for data abstraction to represent network events and aggregated data logs; second, we continue with the methodology to manage data in order to support a database model; and finally we go through the replacement of custom made processing scripts with data-oriented filter modules.

5.1 What is experiment control?

As we have seen earlier on chapter 2, there is a lack of realism and accuracy of simulations, and also integrated and non platform specific tools which provide an experimental environment are missing. We have therefore identified a group of requirements to improve wireless experimentation from the methodology, traceability and reproducibility points of view.

In order to find an integral solution to the requirements, we propose in this chapter a wireless experimentation methodology. A methodology, calls for the body of methods, rules and postulates employed by a discipline: a particular procedure or set of procedures. In our case, we define a *wireless experimentation methodology* as the group of well defined experiment control steps, standard data formats and procedures to achieve a wireless experiment.

We are looking for an Experimental Methodology, to determine the procedures to:

- Identify the wireless parameters which influence the measurements: from them, there are those which can be:
  - Fixed: This group includes the software-controllable parameters, like transmission rate and power. Depending on the control level of the measurement devices, there may be many configurations possible, this is why every variable must be documented in order to keep the same setup to achieve the reproducibility criteria.
  - Directly measurable: This group includes the hardware-controllable parameters, like location and obstacles. Measuring the physical layout has not yet been standardized, so there is still no minimal list of parameters to register. Nevertheless, as we have detailed on chapter 4, there are several factors which affect the propagation, but which have different influence depending on the frequency and the modulation scheme.
  - Statistically measurable: This group include those parameters which cannot be directly controlled, but can be calculated statistically or bounded. Three examples of thos group are the received power variability as seen from each of the wireless probes, the bit error rate experienced for each of the links, and the observed co-channel interference.
- Describe the topology and tasks from a spatial and temporal points of view:
  - Spatial, as it is given by the description of the participating stations, their interfaces and interconnections.
  - Temporal, to describe the operations to execute during the experiment and their evolution.
- Specify the parameters to capture: What is significant to the measurement?
- Determine the data processing methods to merge the data sources and synchronize the events
- Specify the postprocessing calculation methods to extract the desired results
- Specify the storage standard to use in order to keep all the data on the same form.

The methodology should aim to enable researchers to follow a concrete structure, defining a common set of experimental steps, which involves the definition, execution, processing, analysis and storage of the statistics obtained during an experiment.

5.2 Design

In this section we make design choices to select the parameters for our proposed methodology, to define the the steps to follow.

It is composed of a sequence of the six following stages, as illustrated on fig 5.1.

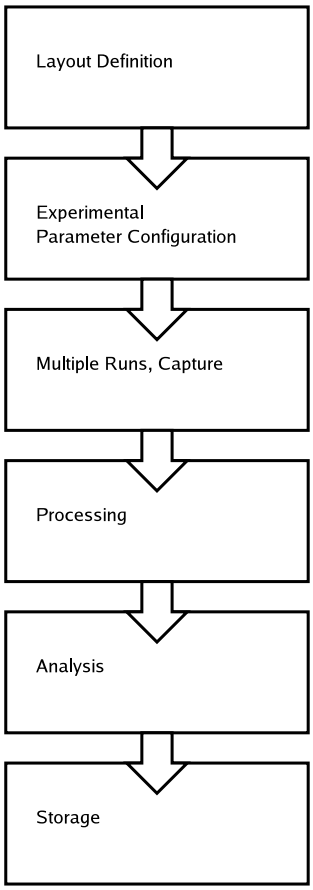

FIGURE 5.1: Experimental Methodology

Each of the stages has a predetermined objective and an output to be stored on the final "experiment record". The description of the stages follow:

Layout Definition: This step includes the description of the environment, as we have described at the requirements section. In order to achieve reproducibility, the user must be able to rebuild the same experimentation layout in another place and at another time by loading and executing the same group of steps.

These items must be described in detail in order to characterize the environment:

- Elements: Identify with unique names each of the known participating elements, also including those elements with a probability to influence the measurement in a significant way. Here we can separate two cases:

  - The Isolated case, where there are no external influences and the whole experiment is known to be within an electromagnetic isolation environment,
  - and the Non-isolated case, where experimentation is executed under the significant influence of other external sources.

- Create a floorplan with the locations, heights and positions of the devices: Electromagnetic propagation is a complex phenomena, which can be perturbed by many factors:
  - Obstacles, which can be full or partial, depending on the construction material and surface. The final result can be a dispersed, reflected or refracted signal.
  - Nodes, which transmit and receive data. Each node has at least one wireless interface with an antenna or group of antennas. For the physical distribution, the most important item is the antenna or group of antennas for each interface. The radiation pattern is fixed for a specific range of frequencies, and it is reversible. Nevertheless, this reversibility does not imply a symmetric channel. Due to environment characteristics, losses and non-linearities on the signal path generate this difference.
  - Surrounding structures and elements, which are not in the LOS of the signal, but they contribute to the scattering and reflection effects, which furthermore generate a complex coverage pattern.

- Create a datasheet for each of the participating elements, noting the hardware used, specifically the wireless card model and version, and the OS, and the wireless driver and applications version. Furthermore, any modifications to the hardware (like changing the antenna) or to the software (like changing the source code of the driver) must be registered too.

Parameter Configuration: The experimentation dynamics must be defined in detail: what data to capture and during which period; when to generate traffic and what kind of traffic pattern, how many times the experiment must be repeated, and at what timescale. The timescale is specially important since networking conditions can change according to time. Here we assume that there is no initial configuration, related to the experiment: if a station should be in monitor mode, the configuration script must be executed to do so.

Multiple Runs and Capture: This step consists of the experiment itself. The *runs* are executed as many times as defined in the former step, following the predefined sequence. The number of *runs* is defined by the researcher, which establishes a balance between the duration of the experiment due to inherent time variation of the channel given by the coherence time and the need to obtain a confidence level of the statistical results.

Depending on the time span, the wireless experiment will need time synchronization for the application start and stop times. For experiments focused on the PHY and MAC behavior, these events are critical, while on experiments centered on roaming events or long-term traffic trends, precise synchronization is not needed at all.

Another type of synchronization must be generated for the measurements. Depending on the type of data capture log, there will be synchronization data available with different time sources and precisions. In order to be able to correlate the data after the experiment, the event collection and aggregate data must be timestamped.

Realtime monitoring can be used if there is a wired backbone (e.g. ethernet) network where the nodes can send status messages. During the experiment, it is valuable to check the evolution of key parameters like traffic load and packet drops, so as to discover anomalies or divergences before the processing step. Furthermore, the task log and the standard and error output of the applications must be recorded for debugging and documentation purposes.

Traces Processing: This step performs offline processing of the captured data to insert it into the common storage facility. The first activity after the data capture task is to join all the information within a coherent structure. This structure must be able to contain all the different data sources with their relationships, the topology and the experimentation conditions. Here all the collected traces and logs are merged while establishing relationships based on time and pairs source-destination. Furthermore, the data structure must support classification of the captured data, in order to simplify and improve the analysis step.

Analysis: This step enables the researcher to extract statistics from the stored data while taking advantage of the structures and relationships created within the storage facility. The process is described by a script which has access to the data, performs searches, filtering, event count and mathematical operations to provide the expected results. Since the expected output of this step is a matrix of elements to display the results, it is important to keep trace of the method to extract data, to fulfill the reproducibility of the method to compare the results. In this way, if the whole process is documented, the reutilisation of the same process is possible, since it is published with the experiment data. For example, two experiments can be executed using the same configuration but in different environments, which enables the researcher to contrast the results between them.

Packing and Storage: Here the captured data is classified, organized and stored in an easily recoverable way. As we have said before, the stored data must have temporal and physical coherence. This means that every item added to the storage must have a

synchronization method with the available data, or a relationship with the other present items. For example, if we extract the list of stations from a packet log table, this derived table is related to the experiment through the observed traffic. Furthermore, if the number of packets at the queue level are available, the samples of this number on time must have a temporal synchronization with the packet log table. The experimental conditions, as we have said before, together with the experiment configuration, the execution sequence and the extracted results capture the environmental, functional and processing steps. All of them are stored with every experiment after the execution, so as to use the processed data as a source for comparison and for the obtention of new results.

5.3 Wextool

After defining each of the required steps of a wireless experiment, we now focus on the 802.11 protocol as the roadmap to our implementation. Nevertheless, the experimental methodology principles we have described before can be used to test protocols for other type of wireless platforms. The main objective is to keep a simple interface for the research community,to keep concentration exclusively on the experimental phase. We have designed and developed an experiment control tool for wireless networks called Wextool.

5.3.1 Technical requirements

The Wextool experimentation tool satisfies the following technical requirements:

Web-based User interface: A Web-based interface provides a standardized and remote access to control and execute experiments, and also encourages the use of the local platform by other members of the research community. This improves the widespread use of the tool, and the promotes the use of diversity on wireless experimentation: A protocol or algorithm which is evaluated on more than one environment provides more information about its behavior, and furthermore, obtains a better validation scheme.

Experimental instrumentation: The instrumentation can be defined to the group of elements used for measuring and capturing data within the platform. Different hardware probes are distributed spatially and software probes within the driver and network stack.

Specifically, the hardware instrumentation of an experimental platform varies from the most simple availability of several wireless enabled stations, which use the same wireless cards as measurement instruments, to spectrum analyzers and software-based radios.

This wide span of instruments can provide simple physical data extracted from the driver level statistics and headers to channel profiles to modulation and bit-level information with high precision. As a consequence, the environmental data must enable, within the structure, to add new complementary items to precise the conditions when specialized instrumentation is used. Furthermore, the captured data will have a different structure which must be allowed to fit within the database. For example, if there is spectrum analyzer in the platform, then the measurement of the spectrum done during an experiment shows the profile of the channel, the associated power value and enables the detection of anomalies at the channel level.

Scheduling: The experiments must be executed while following a sequence of steps to start and stop all the tasks on the nodes. The scheduling then must use either a centralized control of events or a distributed control via an agent on each of the participating stations.

Time Synchronization: It is necessary to synchronize stations to schedule events on the network. Packet traces and network statistics logs must have timestamps to correlate events for the merging step. Time synchronization between experimental nodes can be achieved through NTP or using beacon received on each wireless node. Each of the events (e.g., the reception of a packet or the generation of a report from an interface) occurs at a certain time that will be used later to build the database.

Standardized Data Structure: Raw data cannot be used to load the database unless the parameters are extracted and adapted to a common format. This common format has two main objectives: first, to represent data in a structured format to access each of the fields; and second, to adjust to the search and processing capabilities of the database engine. For example, some data formats are standardized, like tcpdump files, BGP tables, flow export data from routers, and SNMP reports. On the other hand, network interfaces can produce non-standard data formats. Each of these sources requires a parser and interpreter to load the information on the database. We detail the data structure design in the following section.

Realtime Monitoring: This feature is required so that a *run* may be aborted when necessary (e.g. after hardware failures or unexpected instabilities). To enable realtime monitoring, only a few key parameters along with their bounds must be supervised in order not to add a too much measurement overhead on the support network.

Modularity: The toolkit should be extendable so the user can add new modules without damaging nor interfering the activity of the other modules. The whole system may benefit from integration of standard tools. For example, the integration of wireshark can simplify the packet capture and interpretation steps. Wrappers are a solution for

those external tools whose interface does not fit on the capture requirement formats, for example *athstats* from the Madwifi [80] driver.

Open-Source Availability: It is important that the toolkit be developed under a collaborative philosophy in order to benefit from possible contributions and improvements (e.g. new networking environments and layouts) from the user community. Free access to the source code makes the data processing transparent and allows the user to modify the behavior of the tool. We believe that the networking community can highly benefit from a tool with the above features. Its widespread use enables the exchange of compatible data between researchers to cross-validate the experimental results. As a consequence, open source tools for experimentation happen to be the most attractive to the academic and research community.

5.3.2 802.11 physical platform components

Here we define the elements composing the hardware platform where experiments are run. The basic communication element is called a node. On 802.11, a node can behave as a AP, a Managed station, an Ad-Hoc station, or a Probe.

A node is a wireless-enabled hardware device, such as an embedded system, a PDA, or a Notebook PC.

- *AP*: Is a node which works as the master in infrastructure mode and manages the access of the associated stations in range. It also provides access to other networks as a bridge.
- *Managed station*: Is a node which works as managed in infrastructure mode. A station connects via a wireless link to an AP to exchange information with other stations and networks.
- *Ad-Hoc station*: Is a node which connects via a wireless link to other Ad-Hoc stations without the presence of an AP.
- *Probes*: a probe is a node in monitor mode that listens to the traffic on a specific point in the network, and stores this data for further analysis.

Each of the nodes execute an operating system, a network stack, the corresponding driver, an agent in charge of executing the programmed tasks on the node, and the application under test.

Probes can run on the same node than applications or they can be created as standalone nodes with capture and storage software running, e.g. a probe can consist of a notebook,

an ethernet interface and a wireless card. In the case probes and applications share the same nodes, processing done by the probes should be kept at the minimum in order not to overload the system. As a consequence, only monitoring data will be transferred on realtime and on a different medium in order not to interfere with experiments. At the end of the experiment, all the raw and processed data including the layout and configuration information are merged, synchronized and stored in order to be analyzed.

5.3.3 Environment description method

This element is based on a complete description of the devices within the lab with their location, orientation and tilt. This is done with a form to insert the data, classifying each node with it's precise role during the experiment.

The identification of the sources depends on the instrumentation availability of the laboratory. The most simple case is to execute Active scanning as it is specified on the IEEE 802.11 standard, to seek for the presence of external Access Points with the RSSI value from each of them. Another case is to do a passive measurement and extract the identified sources, as we have defined with the Kismet tool on chapter 3. Nevertheless there may be other devices which share the wireless band with the current stations and do not use the same protocol, like Bluetooth and 802.15.4 devices. To discover other type of devices, specialized instrumentation must be used. For example, a software-based spectrum analyzer with data decoder as AirMagnet can measure the wireless spectrum power distribution and detect the wireless devices in range.

5.3.4 Experiment abstraction and scheduling

Within our experimentation control tool we define the control stage, based on the generation of individual scheduling files for each of the participating nodes. On each of the nodes, the agent waits for the scheduling file which is read, executed at the corresponding start time and finally archived.

To generate the distributed scheduling, we need to define the experiment. To do this, we have created a hierarchy through a XML file which reflects the nodes, the tasks, and the schedule as follows.

First, we define the **node**, with the node name and identifier, the role and the its minimal wireless interface data. Second, we define the **actions**, which include non-interactive terminal based commands with the parameters and the redirection of the input and the output data. Third we define an **act**, which is an action to be executed in a specified time and duration within several nodes. Fourth, we define a **scenario**,

composed by a group of nodes which participate on the experiment. And finally we define an **experience**,which is a scenario to be executed a number of **runs**.

With these simple definitions, we can create the schedule for each of the participating nodes, to be sent through the support network to the standard place where the agent looks for the schedule files. Since the generation and transmission of files implies a delay, we have proposed a deferred start time for the whole experiment. An example of an XML experiment description file is shown on Appendix A.

5.3.5 Description of Modules in Wextool

We create a modular architecture where each module has one specific task to fulfill. These modules are coordinated by the scheduler that executes each module as specified.

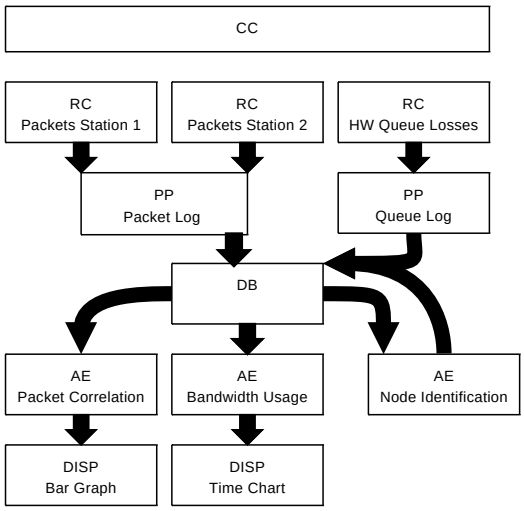

FIGURE 5.2: Relationships between modules

Fig 5.2 illustrates the relationships between the following modules:

Remote Capture modules (RC): One RC module runs on each of the probes and logs the events during the predetermined experimentation period. For example, if the RC module is in charge of packet capturing, a raw packet data file is created.

Preprocessing modules (PP): These modules run on the central processing station and read event logs from RC modules for data interpretation and storage on the database. For example, if the event log is a packet log, the PP module interprets the file and generates a description of the packets, using Packed Description Meta Language (PDML) [56]. Then, a classification engine parses this PDML file to load the database. There is one type of PP for each type of capture source. The PP connects as a client to the database server to store the preprocessed data.

Database module (DB): This module receives and stores the data from the PP modules using a standard structured query language (SQL) [58]. It is composed by a database server that interacts with the PP and the AE modules. The stored data can be used afterwards in different ways, and in particular to:

- verify the validity of experimental results;
- repeat the same setup in a new experiment and obtain new packet logs from the stations; and
- analyze the processed data using different criteria.

Analysis Engine modules (AE): These modules connect to the database and perform queries using SQL. The query results are analyzed and post-processed. The output is sent to the display (DISP) modules or goes back to the database to create node, flow, interface and/or source tables.

Central Coordination module (CC): This module is the dispatcher that generates the sequence of tasks (called experimentation sequence) to be executed on each of the stations. The experimentation sequence is created by the user to control the applications on the experimental nodes.

Display module (DISP): This module is able to create graphics, tables or any other representations from the outputs of the AE modules.

For the first version of Wextool, we have created a PP module to insert, merge and synchronize the captured data from each of the sources in a pre-existent database. There are several optimizations we have included to increase the efficiency of this task:

- We use the Beacon timestamp based on [104], but we use a time window together with a hash value calculated from the whole packet. Nevertheless, other packet synchronization techniques can be added for different synchronization situations.
- We expand the packet capture file with the use of a description standard called PDML (Packet Details Markup Language) [57] to extract the packet data.
- We use a state machine to read the PDML file which has three primitives: Packet, Protocol and Field. This results in processing the captured data as a continuous flow, to manage very long files, typical from packet logs. Flow processing of data enables also the use of pipes to connect between processing modules so the storage resource usage is minimized.

- Since the PDML file generation is not configurable, all the protocol headers are expanded. In order to reduce the number of database queries, we insert only the available fields on the database.
- Since this PP is a module, it can run on the same stations used for experimentation, so the processing task can be run in a distributed manner, whenever a backbone monitoring network is available.

The use of an intermediate PDML file is the consequence of the lack of standardization of the interfaces of the wireshark library. This could be a more efficient manner to integrate the whole process from the interpretation of a standard packet dump file (as explained on chapter 4) to insert the packets into the database.

5.4 The General structure of the database

This section introduces the data structures which are used to represent and classify the captured information. Data structures provide the skeleton where the information can be included in a normalized way, while permitting enough flexibility to expand and create filters to access and index the information. First we briefly introduce the concepts of Data structures and Relational databases, and then we specify the table building blocks and the rules to insert data.

From the analysis we have done on subsection 5.6.1, we found that the current data format is not efficient, neither in space requirements nor for search and filtering tasks. To enhance the data storage and to improve processing steps, we use a database engine where the data structures are based on the table abstraction.

A table is composed by columns, which define fields and rows which share a common field structure. A Database is the container where tables are stored. One of the main advantages of using a database is the flexibility of creating tables and establishing relationships between them. A relationship is a reference between a row of one table to a row of another table through an index number. Another advantage from using a database engine is the indexing capability. An index enables a fast access to the rows of a table through a unique index number for each row.

Finally, all the accesses (e.g. insert, search, filter, delete, count) to the data in a database is done through a query language, which in our case is SQL (Structured Query Language). The SQL language syntax becomes the base for the preprocessing stage of data insertion on the tables and the post-processing stage for the extraction and calculation of results.

Both the database structure created from the table abstraction and the query language provide a simple yet powerful combination for Wextool implementation. On the following we define our scalable table "bricks" which can be combined to describe a packet by the contained protocols without loosing the packet identity.

We propose to organize all the captured data during the experiment using tables on a relational database. In the following, we give five examples of different types of tables:

1. **Protocol table**: Contains the headers of the packets matched with that protocol, e.g.: ETHERNET and RADIOTAP tables. These tables are chained through indexes to rebuild the packets.
2. **Node table**: Contains the node identification and node-related statistics obtained from the discovery or from user-made layout information for each node on the network, e.g.: INTRANET_NODES and WIRELESS_NODES tables.
3. **Interface table**: Contains the parameters classified by interface from the node table, e.g.: WIRELESS_INTERFACES and ETHERNET_INTERFACES tables.
4. **Flow table**: This type of table lists the flows extracted from the protocol tables and their characteristics, e.g.: TCP_FLOWS and RTP_FLOWS tables.
5. **Source table**: this type of table identifies all data sources and their characteristics, e.g.: LOCAL_SOURCES and REMOTE_SOURCES tables.

Nevertheless more tables can be added as long as they belong to the same experiment, to include other type of information, as aggregate statistics from a wireless interface.

The proposed methodology is a tradeoff between preprocessing, classification and analysis tasks. We simplify postprocessing by preprocessing the packet logs and statistics to fit into a DB structure.

The SQL language provides a human readable, unambiguous and powerful method to control searching and filtering criteria on the database. For example, from the packets table, an AE module can extract the node list, and a different AE module can create the flow table.

An important advantage of this approach relies on the use of a single language to perform the queries on the database, which simplifies the analysis methodology. This solves one the most difficult problems with measurement studies: the ability to fully understand how the data was interpreted, and the ability to compare it with new experimental results.

5.5 Data management module specification

In this section we establish the specifications for the data management tool based on the proposed methodology. We have emphasized that Wextool is conceived as modular and that can be expanded by the creation of new insertion and analysis modules. In order to develop new modules, the following specification must be met.

- One database equals to one run of an experiment, or an experiment with a single run.
- All the information from a packet capture during a run is stored in a group of tables called PL (Packet Log).
- The Protocol tables gather the instances of headers of the same protocol. If the same packet was seen twice by different probes, then there will be only one protocol entrance for this packet.
- Each field from a table represents one field from a protocol. There may be less fields than the available ones from the protocol.
- The packet keeps its identity by chaining the headers stored on each table through the use of indexes. As a consequence, each header points to the next header until the end of the packet is reached.
- To keep the structure coherent, there must exist all the intermediate header tables to store each header of the packet until the last protocol. This avoids the breaking of a header chain when there is a header missing.
- The table and field names are the names specified on the PDML standard from tshark. This condition guarantees that the protocol header fields will be found on the Database after reading it from the PDML file, without any kind of mapping.
- There is a special table (also called history table) which is used for synchronization and multiple captured instances of the same packet. This table contains the RADIOTAP headers too, and it must be present always, since it is the head of the chain. There is one entry in this table for each of the observed packets for each probe.
- To be able to synchronize packet traces, there are mandatory elements on the Radiotap table: the hardware timestamp, the calculated hash of the packet contents, and packet size. The first two items are used for time synchronization between packet capture logs from different probes.

The packet log insertion module:

- The PP must interpret the tags from the PDML file, which has packets, protocols and fields as elements.
- There are two ways to insert a packet from a packet log: As the first appearance of the packet or as a already seen packet.
- The first instance of a packet is inserted by protocol, populating all the tables in sequence, and chaining one header to the following with the index of the next header.
- A new packet which is already into the database is inserted only on the History table, to keep a referece.
- A packet is considered to be new when it fulfills two conditions: first it has a hardware timestamp which is different than the previous and following packets, and second, the hash of the contents is different.
- The packet hardware timestamp stored on the RADIOTAP table is synchronized with the beacon timestamp for a unique AP source. On the arrival of each beacon, the hardware timestamp and the beacon timestamp are subtracted. This difference is added to the following packets of the same packet log in order to keep a single timebase from the chosen AP.

The aggregate data insertion module:

- The aggregate data must have on each entry a *hardware timestamp* provided by the driver in order to sync the data with the captured packet log. If this data is missing, the system timestamp is of use whenever there is a way to correlate both.
- A single table per probe must exist inside the database
- Since the provided data varies with the sources, there will be one function or one different module per data source.

The data extraction module: The data extraction module will execute SQL filters on the database depending on the specified measurement.

The parameter feedback module:

- This module is based on the same rules as a data extraction module but the output is a synchronized table within the database for time-related data

- Other type of data can be the output, which is no time related but includes countable elements.

5.6 Current format for measured data representation

Before the description of an example packet log database, we would like to describe briefly the current format for the measured data, which is further used for analysis. A file format is the standard method for stored data exchange between applications. The format itself defines the ability for an application to access the data and the efficiency on the retrieval of the information.

5.6.1 PCAP

Instrumentation on wireless devices provide several file formats, the most common of them being the tcpdump file format [88]. This file format stores the packets with a header describing minimal capture conditions: the file format version, the max packet size, and the data link layer used. The rest of the file, consists of a sequence of packet header / packet payload in binary form.

The packet header contains the timestamp of the packet with microsecond accuracy, the recorded packet size and the real size of the packet when captured. These are the only traces from the measurement which are stored at capture time. Furthermore, this structure is functional for storage purposes, but it is not practical for other type of manipulations since there is no index nor any other method to obtain directly the payload of the packets without going through an interpretation of the contents for each packet. Furthermore, there is no provision to separate each packet into protocols headers, nor to execute statistical measurements. The main advantage of this approach is the completeness of the file, which provides the trace of all the events captured during the experiment by a probe. On the other hand, packet dump files are large even for short duration experiments. For example, a 5 minute experiment with two TCP flows and one RTP multicast flow generates about 400Mb of trace foe one probe.

A new version of the tcpdump file format was proposed as an IETF draft called "PCAP Next Generation Dump File Format" [89]. Although this format is in development, it is worth to remark some enhancements over the traditional tcpdump format. The PCAP next generation format adds flexibility with the use of data blocks across the file, each of them defined by a block header. The most relevant improvements on the contents include the a better description of the capture interface, a packet hash and a

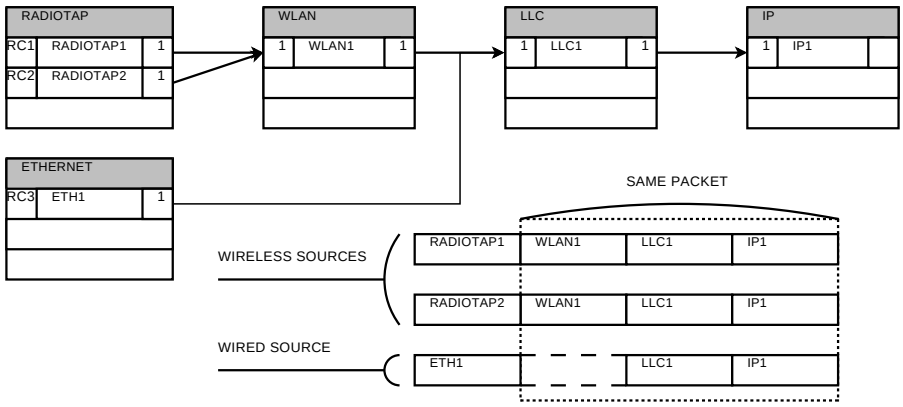

FIGURE 5.3: An example of database structure

detailed description of the link layer, the number of packets lost between packets, the name resolution block which stores the dns or arp retrieved information for the packets present on the file, and an interface statistics block which includes aggregate data for a specific timestamp during the capture. Although these enhancements add more relevant information to the capture file, this is still a file format representing a capture for a specific period.

Other captured data sources come from statistical data which have no standardized nor specific format, from different utilities; e.g. ifconfig, iwconfig and athstats (this last one from the madwifi wireless driver)

5.7 An Example Packet Log Database

In this section, we describe an example of packet log processing. We assume the presence of three packet capture RC modules that generate three different lists of captured packets. These lists are processed by the same PP module, which dissects each packet on each of the lists and identifies the fields to record them on the corresponding tables.

Fig 5.3 illustrates the original packet structure and the final destination of the headers on each of the protocol tables. We can note that the corresponding protocol tables are: RADIOTAP, containing the reception parameters for the packet; WLAN, containing the addresses and protocol specific data; Logical Link Control (LLC); and IP.

Now let us assume that another probe has captured the same packet. The merging function should verify if the packet is already present on the database, through the comparison of capture timestamps and verification using a hash on the packet payload. If the payload hash has the same value and if the difference with the former timestamp is within the clock drift window, one can deduce that it is the same packet. Thus, a new

entry is added on the RADIOTAP protocol table, and the index on this entry points to the former WLAN protocol entry.

Furthermore, let us assume that there is a new probe that captured the same packet: a server connected to a LAN in a remote location. One of the probes logs this event on its own packet capture file. At the end of the experiment, the log file from the remote RC module is processed by the packet log PP module. Since the packet already exists on the database, this event will be saved on the ETHERNET protocol table with a different timestamp. The index will identify the same IP payload. To obtain this behavior, the packet log PP module calculates a hash on the invariant part of the packet, i.e. in this case, the IP packet and its payload. We use this mechanism because the timestamp method is not valid here. Indeed, the packet has traversed the sniffed node and the delay has no relationship with the propagation delay on a Wireless LAN.

The resulting tables store the classified headers of the packets in a coherent, compact and comprehensible representation. Furthermore, the fact that the RADIOTAP table keeps all the occurrence of single events from all the probes even for the same packets provide a much richer information not present on any other available format for wireless packet dumps. For example, the decoding range of a packet by a probe determines the limit where the current packet arrives, and furthermore the source of packet loss in a broadcast or multicast protocol.

On the other side, the RADIOTAP and ETHERNET headers keep a record of the history of the packet, and from every instrumented element where the packet was observed. A complete and automatic trace of the packet can be drawn through a wireless and wired section, provided that packet uniqueness is assured as we described above.

Once the data is stored within the database, the AE modules are used to provide filtering, statistical processing and retrieval of results from the database. For example, an AE module can execute a filtering rule that selects all the WLAN and Ethernet headers referencing the same packet. The results from this query will list the number of the times the packet was seen on the network.

5.8 Flexibility and Scalability of the data structures

The use of a database to organize the captured data within a structured manner simplifies the representation of data, letting the user add the required protocols for the particular experimentation objectives. The storage then is scalable on two axes:

- The protocol side; it takes advantage of the simplicity of the model, which only requires two extra fields, pointing to the next protocol within the packet;
- And the packet trace source side, since the same packet can be captured by probes in different mediums (e.g. wireless and ethernet): The same packet will be linked by elements from two source tables: RADIOTAP and ETHERNET.

Furthermore, there is a filtering simplification. Such a data structure permits not only the inclusion of new data structures by the user in a transparent manner, but also generates filtering scripts which do not need high specialization on the use of databases, leaving a group of concrete steps to arrive to the results.

Nevertheless, there is a tradeoff behind this structure: It is recommended to normalize databases, this means, in a few words, simplify the data representation structures to the point where minimal redundancy is attained while the internal structures are still coherent, but with an important loss in the intuitiveness of the Protocol-Table association.

The cost of the use of this solution would be complex postprocessing scripts which need higher expertise from the user, and the creation of a general purpose database which, although formally correct, does not represent directly the packet structure while browsing it.

5.9 A simple experiment

In this section we propose a sample experiment so as to show how to fulfill all the steps in order to illustrate our methodology. A real and complete use case is left for the following chapter.

We propose as an example, the wireless throughput measurement on UDP traffic between one AP and one station. The objective of this experiment is to show the dependence of the throughput with the distance between the station and the AP.

5.9.1 Layout definition

To define the layout, we must first review which are the devices and the instruments available. We use two standard laptops with a wireless card which supports both roles: AP and station. The first laptop will act as an AP and the second as a station, connected to the AP. First, we use a laptop as a station and not a commercial AP since it is easier to manipulate to and to instrument. Second, since the power control at the stations

does not allow to reduce the transmission below 0dB, the distance cannot be achieved indoors, so the measurement will be done outdoors. Furthermore, the fact of an outdoor measurement reduces the number of reflections and obstacles. Third, we describe the environment with the most of details: location, orientation, height of the AP and the station. Fourth, all the measurements in distance will be done with the same station in different positions, in order to minimize the effect of the differences between cards and hardware in general. The disadvantage here is that we will not be able to have the same wireless conditions in time, and a longer measurement at each location will be required in order to validate the result statistically. Our metric will be the number of packets lost, per second. We design a map with the locations of both the AP and the stations, in order to register the description of the experiment.

5.9.2 Experimental Parameter configuration

The transmission bandwidth between the station and the AP depends on the environment, including noise and interference, on the rate and on the load of the AP and the station. First, we propose to measure the interference between both elements at each of the measuring points before each capture for the same duration. Although the measurement of the environment during the experiment is not possible, since we are adding traffic to the environment, the reference measurements before and after will represent the state of interference on the channel. Second, since wireless propagation is a time and space varying phenomenon, the duration of the experiment must be specified in order to capture the channel variability. From the wireless models available, we can calculate which is the minimum time where we expect the variation of the channel due to the propagation effects. Third, in order to control the experiment and minimize all the bias from the wireless cards, we must disable the noise reduction algorithms and we must fix the receiving antenna. Although both parameters can be used also as control variables to configure different experiments, we are going to bypass this possibility to focus only on the rate. Fourth, we define a group of transmission rates for the station, from the available 802.11b rates. To capture the transmitted packets, we add a wireless probe to the experiment, next to the transmitter.We count only the transmitted packets as good, and not the packets which were dropped at the kernel packet queue or at the hardware level queue. Fifth, we configure Wextool to generate an experiment for each of the rates, for each of the locations with 10 runs of the specified duration, one with the traffic, and one without for control purposes. In the experiment, the station sends UDP packets (with iperf, for example) and the packets are captured on the probe and on the AP. We fix the number of retries to 0, and we fix the packet size to 100. Once more, both items could be used as control variables, but we keep concentrated on the rate.

5.9.3 Multiple runs, capture

In this step, we create the scripts which will be run on each of the participating elements (AP, station, probe) and we schedule the configuration, the packet generation and the packet capture stages. For each of these tasks, we have to give a time gap to achieve the configuration and the association with the AP. The experiment is executed as planned, with 10 runs for each of the points and each of the rates. The location is changed manually between experiments. Either a fast check on the captured packets between measurements or a wireless traffic passive monitor will show if there were problems or faults during the experiment.

5.9.4 Processing and Analysis

The data is inserted using Wextool into the databases and we measure the correlation between the transmitted packets (from the probe) and the received packets (from the AP) is measured with the throughput module. Each database corresponds to a an experiment run, and we obtain the results either as a table to merge the information externally or we post-process the results with a new module to group them and generate graphs to observe confidence levels within each experiment.

5.9.5 Storage

Finally, we store the qualitative and quantitative experimental data:

- The layout description: a map with the position of the elements, their height, tilt and orientation.
- A description of nearby objects which may potentially generate reflections, scattering or other distortions to the received signals.
- The scripts to reproduce the experiment.
- The databases with the preprocessed data.
- The scripts to process and obtain the results.

After this last step, we have created a whole package which keeps not only the results but the data sources and the steps to reproduce our experiment.

5.10 Conclusion

We have presented in this chapter a the data model to represent in a simple, coherent and intuitive manner the packet capture data, together with synchronized aggregate data from the participating interfaces. Then, we have specified an experimentation methodology with an integrated structure to provide reproducibility and traceability of the results; Finally, we have described the implementation of the experimental methodology called Wextool, which shows the feasibility of the proposed mechanisms.

Furthermore, we believe that such an experimental methodology is crucial to characterize the traffic, execute experiments under common methods and promote the exchange experimentation data between research teams. We have put emphasis on the fact that the processing and analysis stages will be of great help to researchers by simplifying their task so that they concentrate on the experiment itself. The modular approach simplifies the expansion and improvement of the tool, and the database engine provides structured storage, flexible search and filtering functions. Finally, the combination of preprocessing modules, database storage and post-processing modules creates a balance between the raw data on one side and the experimental results.

Chapter 6

Leader-Based Multicast: The Case study

This chapter provides a real case study on the feasibility of 802.11 leader-based multicast is shown by several experiments.

Multicast in WLANs represents a challenge to other validation techniques such as simulation or emulation: The main requirement is to send a stream of packets to multiple receivers which observe each of them from a different point of view. Furthermore, WLAN multicast leader-based approach requires to have a feedback representative -called the leader- which will react on the WLAN multicast packet stream in order to increase the percentage of correct packets arrived to the receivers. The behavior of the WLAN multicast stream under different traffic conditions leads to the use of experimental techniques, since it is the only method which takes into account the time and spatial correlation of packet loss.

We take advantage of this case to compare two experimentation methodologies: First we use current common practices, as it can be found on the literature. Then, we execute the same experiment using the current implementation of Wextool. Finally, we analyze the performance issues for the insertion and synchronization tasks which represent the most resource consuming process.

6.1 Brief background on Multicast in 802.11 networks

In order to validate the experimentation methodology we chose a case study related to 802.11 Multicast video transmission techniques because it requires several factors: first, the analysis of packet logs from multiple probes; second, to aggregate data from

instrumentation at the driver level, third, multiple experiments and control conditions to reproduce and fourth, a cross-layer approach to understand the protocol behavior.

Furthermore, this application requires an experimental method to keep track of the values for the controlled variables, the statistical variability from the difference between experiment runs and the automation of experiment reproduction in order to avoid biases and errors due to human intervention.

Multicast video streams instead of unicast streaming results in a much more efficient use of the shared wireless medium. Whereas all these new applications are very likely to appear soon with upcoming WiMAX or DVB-H enabled devices, the IEEE 802.11 standard does not comply with multicast data requirements. In particular, the current MAC layer sends multicast packets in open-loop as broadcast packets, i.e., without any possible acknowledgments.

This open-loop transmission mechanism causes three main problems. First, without a feedback mechanism, it is not possible to adapt the contention window according to the network state, as it is done with regular point-to-point connections. Consequently, multicast flows achieve a higher priority[1] than concurrent unicast flows and the network may become severely congested and could collapse. Second, it is not possible to adapt the physical (PHY) transmission rate to the channel characteristics, so the packets are broadcast over the wireless medium at one of the rates included in the basic rates set. Third, there is no way to retransmit lost packets at the MAC layer, so the transmission is more lossy than for unicast flows, which degrades performance of the multicast application.

One of the alternatives to improve the standard multicast approach is the leader-based multicast mechanism [66][68]. In a nutshell, this solution proposes to select one of the receivers to send acknowledgment frames back to the sender. As with regular unicast transmissions, the multicast sender can use a PHY rate selection mechanism such as ARF [64]. Furthermore, the leader-based approach provides fairness with other concurrent unicast flows because the same algorithm is used to adapt the contention window.

The previous solution to improve performance on multicast over WLANs is only based on simulation models. Therefore, we believe that it is crucial to check if these assumptions are realistic with measurements made on current 802.11 devices. We measure the packet loss correlation between multicast receiving stations. We also compare characteristics of multicast flows and unicast flows to evaluate the gain obtained with a leader-based [66, 68] controlled multicast session.

[1]Note that we do not advocate here for a strict fairness between unicast and multicast transmissions, we only notice the absence of ways to prevent multicast flows to swamp all the network resources.

6.2 The challenge

Multicast measurement on WLANs is one of the most complex measurements which can be done on infrastructure mode, since it requires synchronized data from many sources to compare the data origin, and furthermore, it needs to include aggregate data logs to correlate measurements. At the mac level, the behavior changes when in low-power mode [2], and, in order to test different proposals, there is a need to change the behavior of the rate and CTS/RTS usage. At higher network layers, the RTP protocol must be analysed with a cross-layer approach to understand the origins of packet losses, looking at the PHY layer and correlating with the reception success from nearby probes. The current mechanism forces standard multicast to behave similarly to broadcast packets in normal mode, meaning, no ACK and base rate. The proposed multicast mechanism supposes the existence of a leader between the multicast receivers, who is going to send feedback through an ACK. With feedback, the point-to-point advantages can be implemented, as CTS/RTS, packet retransmission and rate adaptation. But to maximize the gain, the choice of the leader remains critical, so as to choose a leader which best represents the needs of the other multicast receivers and helps constructively when asking for packet retransmission or for rate adaptation.

6.3 The metrics

The focus of this study is the video stream performance of multicast streaming for infrastructure mode WLANs, where the critical parameters are the throughput, packet loss and delay. Throughput is measured as the quantity of packets arrived per second to each of the receivers, with the standard multicast with fixed step rate conditions and adaptive rate conditions when ACK was enabled. Packet loss is classified between packet drops at the AP's queue, packet loss due to correlated losses between all receivers, and uncorrelated losses. Since retransmissions are disabled for the experiment, the only separate source of delay would be the increment in backoff time (in the case of leader-based solution) and within a loaded environment. We observed in our experiments that the delay variability increased, but not the total delay. So the most representative item which cannot be taken from simulation is the correlated and uncorrelated packet losses, since they depend not only on the distance but on more complex channel effects as fading and multipath scattering, which are uncorrelated on the simulators models. Our experimentation method is based on the comparison between the standard multicast

[2] In Power-save mode the only difference is the marker on the beacon packet which signals when a station has packets waiting to be sent. In this case, the broadcast and multicast packets are sent after immediately the beacon.

mode on wireless, as established by the IEEE802.11 standard, and a feedback enabled multicast, which is replaced with an unicast flow between the Access Point and a station while the other stations execute a passive packet capture.

6.3.1 WisMon

The first series of experiments were done using the Wismon tool. WisMon[86], is a graphical software environment for data acquisition and automatic generation of per-station statistics logs. In this series, the experiment itself was processed offline since the processing requirements exceeded the available realtime capacity, and loaded the nodes altering the measurements.

We use a multiple coordinated probe scheme in order to capture all the traffic on a WLAN, because none of the probes is likely to receive all the transmitted packets [73]. Moreover, time is synchronized from a common reference available to all the stations[3], and the received packets are re-timestamped using this new time reference. Then, the captured data from all the probes are merged with a time-constraint criteria in order to remove duplicated packets, i.e., packets received by more than one sniffer. This list of packets is then filtered and processed through an on-line *packet classification engine*. In this manner, we can obtain near real-time histograms to analyze the behavior of WLANs. Several parameters can be obtained on a per-station basis such as received power level, inbound traffic, outbound traffic, number of retransmissions, PHY transmission mode and percentage of bandwidth used.

There are two functional modes for WisMon: offline and online. The offline mode can be used for example to study long-term patterns of some parameters, whereas the online mode is used to monitor parameters during the experiments. In the latter case, it is possible to detect anomalies in real time on a WLAN. Characteristics of current stations connected to the WLAN can be analyzed in real-time, allowing to focus the analysis of a particular station when necessary, see Figure 6.1.

A configuration file can be used to select and customize the sniffers to connect to, and to generate packet traces from the *packet classification engine*, as shown in Figure 6.2.

WisMon is built using a client-server architecture, in order to separate the heavy packet processing functionality from the lightweight GUI client. The whole system is built as open source, which results in a very convenient and customizable tool.

[3]The time reference is obtained using the Beacon frames which are periodically sent by the AP.

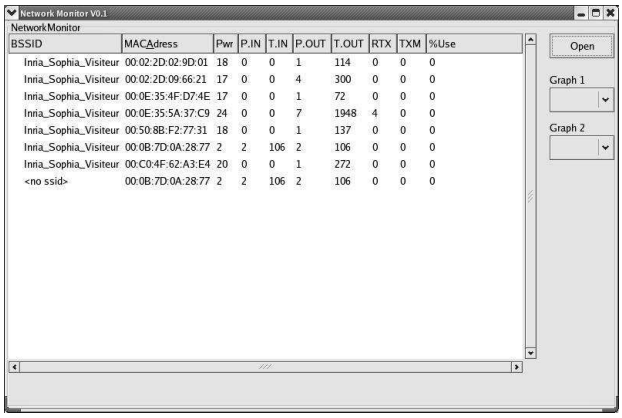

FIGURE 6.1: Screen capture of station list window from WisMon

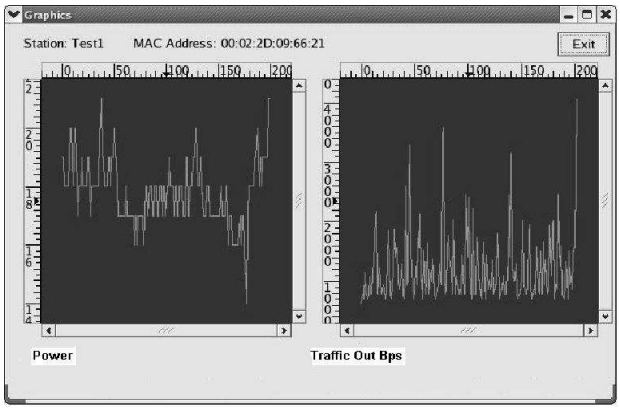

FIGURE 6.2: Screen capture of the real-time graphics from WisMon

6.4 We created the first method

Since the objective here was to analyze the packet loss correlation (temporal and spatial) under different rates, with retransmission and variable load conditions, then the layout consisted in a single source sending multicast packets to a dummy station and several probes which captured the packets from different points of view of the source: LOS, NLOS, corners, with and without walls in-between, different distances.

This layout was designed to generate different physical characteristics, mainly due to reflections and scattering. Video multicast packets were transmitted over the medium, while different amounts of traffic were injected with the objective of collision generation.

Five runs of five minutes capture each were executed, to obtain values to establish a confidence level. In total, more than 300 runs, comprising 60 experiments were executed manually to obtain the present results.

All the packet logs were stored within the individual station hard disk and retransmitted to a central processing machine. Statistics also were extracted from the source, where the

packets dropped at the AP multicast queue were extracted after further instrumentation of the wireless driver.

The video stream was transmitted as a RTP flow, so each packet had a sequence number (16-bit length, rollover prone) The packets were captured within the tcpdump file format, which had to be converted to a text-based packet description file to extract the sequence number of the packets. This generated a list of sequence numbers. Another script calculated the distance between the consecutive sequence numbers and generated two files: the packet loss burst file and the missing sequence number packet list.

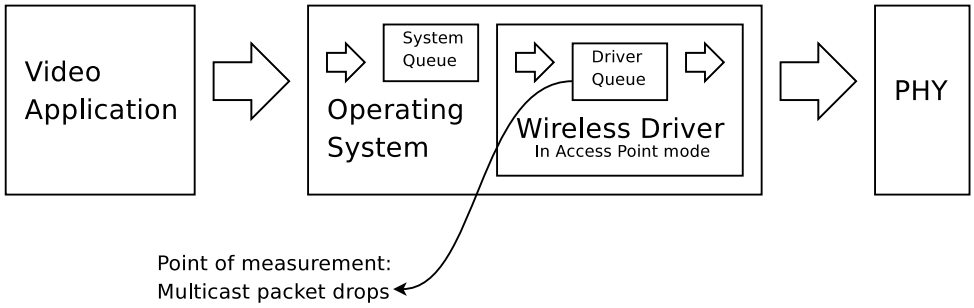

FIGURE 6.3: Instrumentation on the driver queue to measure packet drops

The missing sequence number packet list was compared against the other packet logs of the same run to check if the same packet was lost on another probe. If this was the case, a spatial correlation loss happened. But a packet which did not appear on the medium could have been lost at the queue level on the driver as we can observe on Figure 6.3. Since we did not know which packets were not transmitted due to a queue drop, but we had the number of packets dropped every second, then we subtracted the amount of dropped packets from the spatial loss correlation number. With this method, we differentiated the sources of packet loss. This procedure was done between all the possible combinations of one to five stations to see if correlated lost packets were lost individually, by pair of probes or by a bigger group of stations. As a consequence, a packet which was lost simultaneously by all the stations was a consequence of a collision. On the other side, a lost packet which was not correlated meant a loss due to interference.

This whole analysis was executed also for each of the runs from the same experiment. Different experiments were executed for a group of rates (1, 2, 5 and 11Mbps).

If any of the runs failed during an experiment, then the whole experiment was repeated, so that we keep temporal uniformity (same time of day, for example).

Furthermore, the experiments were also human error prone, due to the mechanization of the tasks, mistakes due to miss a capture start, lack of disk space, missing multicast routes after update and manual reconfiguration before the following experiment.

To sum up, offline complex processing tasks were executed using common practices: hand made installation and preconfiguration of the participating nodes, start and stop of the experiments was done also one by one; data transfer was sent over the support network without automation; filtering was achieved using custom scripts and the creation of graphs needed a post-processing effort through standard spreadsheets. These are resource-consuming practices need surveillance of the parameters during the experiment and, although Wismon simplified part of the monitoring tasks, the final results were not available until the whole post-processing and graph generation tasks were finished. This last fact produced even longer delays to arrive to the results.

6.5 Automate the experiment steps

Although the measurement sequence we propose takes into account most of the controllable factors, the experimentation time to obtain results required one week from the configuration to the obtention of the results. So our limitations were the lack of automation, the complex packet loss correlation detection mechanism, the lack of a standard data structure and the post-processing overload.

Furthermore, we believe this hand-made use-it-once procedure is a common practice, unlike simulators, where configuration, simulation itself and the calculation of results is mostly specified. Additionally, it can be seen in literature that although experimental conditions exist, they are described partially along with the traffic generation method, but the detailed actions taken during the experiment, the packet log treatment process and the post-processing tasks are absent.

From a more general point of view, the rise of wireless network usage has unveiled the widespread use of simulation which was inherited from wired networks, and the lack of rigorous experimental practices which are present in other sciences.

6.6 The first testbed

In our wireless testbed, both the stations and the AP are composed of standard laptops with off-the-shelf wireless boards. This solution allows us to instrument the AP driver in order to differentiate packet drops at the sending queue from packets lost due to collisions or bad channel conditions. Furthermore, our previous experiments made with three different commercial APs (Netgear WG602, Netgear ME102 and Linksys WAP55AG) showed that these APs periodically deauthenticate stations when the network is congested. This problem biases statistics because a re-authentication requires

up to 7 seconds (with no traffic exchanged), and results in long bursts of artificial packet loss. Furthermore, we have instrumented the AP in order to provide statistics of its sending queue. In this manner, we are able to differentiate collisions from packet lost before transmission.

It is important to note that during the experiments all the stations are fixed and do not enter the sleep or power-down modes. When a station changes to power-down mode, the wireless board sends a message to the AP to start buffering the packets until it recovers full activity.

Each laptop uses a Linux-based system because it brings more flexibility[4]for WLAN instrumentation.

As a complement to WisMon, we have developed the following toolchain. Tethereal (the terminal version of Ethereal, now Wireshark) captures and builds logs for each station. Vlc[85] is used to send video packets in multicast. These packets are then filtered, and the text output is parsed using a filter which extracts for each packet, the Prism header information and the RTP[71] sequence number. Finally, a script creates a file containing the list of lost packets, which is then post-processed to obtain the packet loss correlation values. Further details on the experimental setup follow.

Hardware:

- 6 laptops (STA1, STA2, STA3, STA4, STA5, AP). We use Dell Latitude D800 (1Gb RAM, Pentium M 1.7GHz) and COMPAQ EvoN800c (256Mb RAM, Pentium M 1.7GHz) with IEEE 802.11b Proxim Orinoco Gold wireless cards (Atheros AR5212 chipset).

Software:

- Operating System: Linux kernel version 2.6.8.1 installed in all the laptops.
- WLAN Board Driver: Madwifi[80] driver.
- Streaming generator and client: vlc[85] (VideoLan Client) media player version 0.8.1, used as a video server. RTP is selected to send video either in multicast to a group of stations or in unicast to a specific vlc client.
- TCP and UDP background traffic generator: iperf version 2.3.5

Packet Capture software:

[4]Note that the promiscuous mode is not always available on commercial Windows drivers. Furthermore, when this mode is available, sometimes management frames are discarded by the driver [75].

- kismet-2004-10-R1[78] modified for time synchronization. These modifications are available as a patch in [86]
- WisMon[86] version 0.1-R4.

6.6.1 Physical Setup

In our wireless testbed, 5 probes (STA1-STA5) are distributed in the receiving range of the AP to collect the traffic from different places. In this way, we obtain a variety of signal-to-noise ratios (SNR) and receiving power levels. The mean receiving power values (measured from the collected data) for each station are shown in Table 6.1, whereas the position of each station is drawn in Figure 6.4.

TABLE 6.1: Mean receiving power value for each station

Station	Power Value	Group
STA1	-45dBm	near
STA2	-51dBm	near
STA3	-64dBm	far
STA4	-52dBm	near
STA5	-61dBm	far

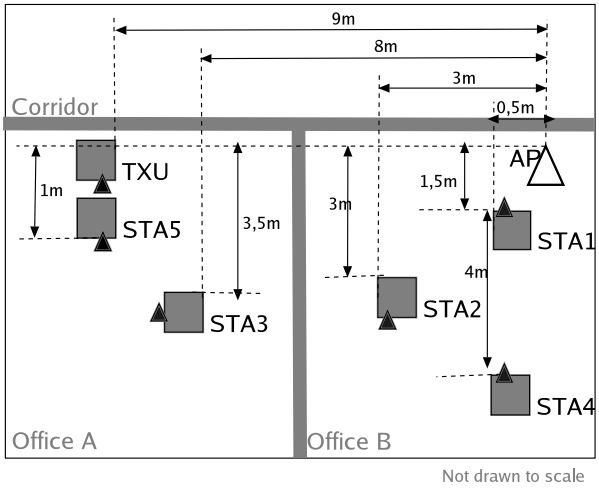

FIGURE 6.4: Distribution of the AP and wireless stations. There are two groups: near (STA1-STA2-STA4) and far (STA3-STA5)

The first three stations (STA1-3) are located in the same office as the AP (i.e., Office B). STA1 is the nearest station from the AP. It receives a very high signal from the AP. STA2 is located in a corner at Office B and in a place where the receiving signal is lower than the one of STA1. At this position, there is a high probability to find reflected

signals from nearby structures. STA4 is farther from the AP than STA2 and is placed at a very good reception spot.

STA3 and STA5 are located in contiguous office (i.e., in Office B). STA5 is placed at the worst place, corresponding to the corner of Office B. STA5 is located at a spot with better reception than STA4.

From Table 6.1, we can note that although STA3 and STA5 are located in a contiguous room and farther from the AP than STA4, they obtain a higher received power level than STA4, which is in the same office as the AP. One possible cause of this behavior is the proximity of physical structures and objects that may generate signal reflections.

To generate different levels of channel load, we use iperf to generate concurrent UDP or TCP traffic from each station.

Legacy multicast transmissions use the default transmission mode of the madwifi [80] driver which is equal to 1Mbps, whereas ARF [64] is used to select the PHY mode (from 1 to 11Mbps) for both unicast and leader-based transmissions.

6.6.2 Leader-based implementation issues

There are several ways to implement the leader-based approach. The most direct way is to modify the legacy multicast mechanism as follows. One of the receivers in the multicast group is elected to send acknowledgment (ACK) frames. The Duration/ID field of the MAC header of each multicast data frame has to be modified. In particular, the virtual carrier-sense mechanism provided by the MAC should take into account the extra delay for multicast acknowledgments. It is important to make the latter modification in order to prevent possible collisions between multicast ACK and other frames.

However, this solution cannot be used on our current WLAN devices. Indeed, generating ACK frames requires very precise timing synchronisation and is implemented within the hardware of the wireless devices to comply with this requirement. A simplified schematic of the receiver side of a 802.11 wireless card is shown on Figure 6.5

The first block synthesizes the RF and decoding stages. When a packet arrives, it is directed to the packet buffer. Packets are then selected depending on the MAC address destination field on the multicast filter and the unicast filter. The multicast filter selects the packets corresponding to the multicast groups the receiver has subscribed to. The unicast filter selects the packets addressed to the card.

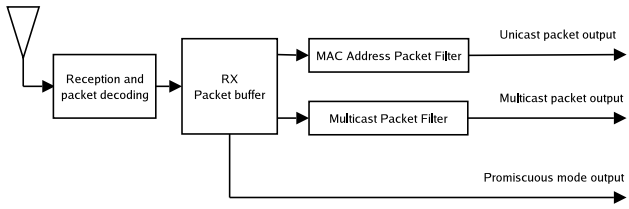

FIGURE 6.5: Receiver side of a 802.11 WLAN board

Fortunately, it is possible to implement the leader-based approach with current available hardware and make changes only to the software driver. For example, it is possible to fake multicast transmissions using the promiscuous mode on all receiving stations while the AP sends unicast frames to the leader-station. In this manner, all packets sent to the leader are also received by other stations. At the application level, packets must be processed to remove all the headers manually, since this method overrides the TCP/IP stack.

To minimize processing overhead, receivers can apply a filter at the kernel level to only let packets, sent by the AP to the leader (at the port corresponding to the multimedia session), reach the upper layers.

The promiscuous mode must not be passive, to allow other sources of traffic to access the medium. For example, it is possible to configure the madwifi driver to create two virtual devices: a standard driver to connect to the wireless network and a promiscuous mode driver to obtain the packets directly from the buffer.

The proposed solutions rely on the chipset capabilities and configurability, which vary depending on brand and model.

6.7 Experiments and Results

In this section, we first show how unfair the IEEE 802.11 legacy multicast is with other concurrent unicast flows in current WLANs. Then, we make various experiments in order to evaluate the leader-based approach and compare its performance with the standard multicast transmission mechanism.

6.7.1 Why legacy multicast does not work

As discussed in the introduction, the legacy IEEE 802.11 multicast is an open-loop transmission mechanism, so, it is not possible to retransmit lost packets or select the best PHY rate mode according to the channel conditions. But the most severe problem

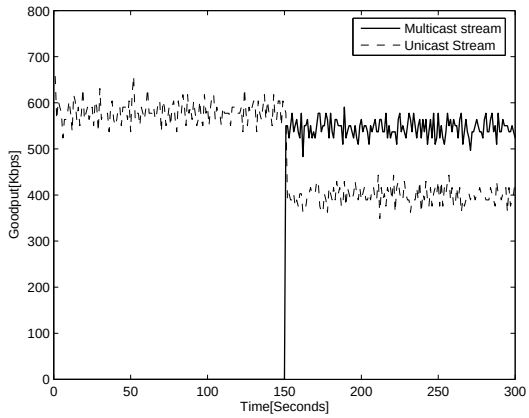

FIGURE 6.6: Goodput multicast - unicast unfairness without background traffic

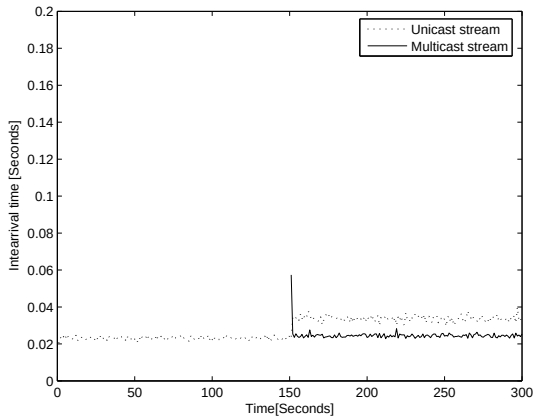

FIGURE 6.7: Interarrival time multicast - unicast unfairness without background traffic

is that contrary to unicast flows, legacy multicast flows cannot adapt their probability to access the channel according to the network load. This leads to severe unfairness between multicast and unicast flows and can even cause network collapse.

To illustrate this problem, we have compared the characteristics of two identical UDP/CBR flows transmitted simultaneously in unicast and in multicast modes from STA5 to the AP. We use *iperf* to generate both flows with CBR=500kbps and the default packet size equal to 1678 bytes.

Figures 6.6 and 6.7 show respectively the goodput and the interarrival time between two packets sent in the two different modes with unloaded traffic conditions, i.e., without background traffic.

At time $t = 0s$, the unicast flow is started at STA5 and achieves an average goodput

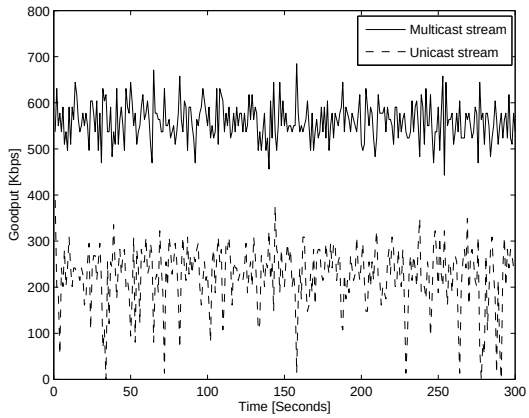

FIGURE 6.8: Goodput multicast - unicast unfairness with background traffic

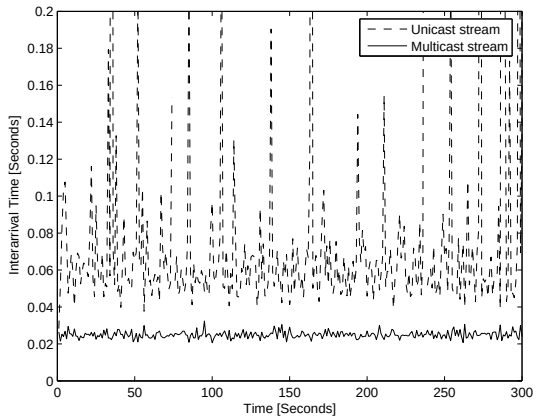

FIGURE 6.9: Interarrival time multicast - unicast unfairness with background traffic

of 600kbps [5]. The mean interarrival time between two packets is about $25ms$. At time $t = 150s$, the multicast flow is started on STA5. We observe that the multicast flow obtains the same performance than those observed by the unicast flow in the first $150s$. However, the performance of the unicast flow suddenly drops of about 30%, i.e. 30% less goodput and 30% more delay between two successive arrivals of packets.

Figure 6.8 and Figure 6.9 show the same experiment but in presence of high background traffic and when both unicast and multicast flows start at time $t = 0s$. To generate highly loaded network conditions, four saturated UDP unicast sources are added on each remaining station (STA1 to STA4) to the AP.

As expected, we observe a higher goodput variability for both unicast and multicast flows. But this time, the difference of performance obtained between these two flows is larger than before. The average goodput for the multicast flow is about the same than

[5]Note also that the CBR traffic of 500kbps does not include the RTP/UDP/IP/MAC/PHY headers overhead.

in the previous experiment (i.e. without background traffic), whereas for the unicast flow, the mean goodput drops for about 70% of its original value. Indeed, the quality of the unicast stream severely decreases whereas it remains roughly the same for the multicast stream.

6.7.2 Comparison between legacy multicast and the leader-based mechanisms

In this section, we present a comparison between the 802.11 legacy multicast and the leader-based approach. We consider a video streaming application in which a vlc video source (located in the same LAN than the AP) sends a VBR video to a group of 5 receivers (STA1-STA5). In order to analyze all types of packet loss, we have run experiments for 2 different network conditions: without background traffic and in presence of congestion. Therefore, four experiments, as detailed in Table 6.2, have been run to generate all the figures shown.

TABLE 6.2: Table of Experiments

Exp.	Back. traffic	PHY mode	Tx method
1	No	1 to 11Mbps	Multicast
2	Yes	1 Mbps	Multicast
3	No	1 to 11Mbps	Leader-based
4	Yes	1 Mbps	Leader-based

We use a standard DVD movie to generate realistic video traffic. The video stream is encoded by vlc with mpeg2v [87] and it is configured to send a VBR video flow with RTP/UDP encapsulation. Notice that although vlc has been configured to send a mean rate of 512kbps[6], the actual average sending rate is roughly the double. The background traffic is generated with iperf [77] as follows. For the first experiment, 10 UDP unicast flows directed to the AP are started per station, with a requested bandwidth of 10Mbps. This ensures that there is always a packet waiting to be sent in the driver queue. For the second experiment, 10 TCP unicast flows directed to the AP are started per station, which results in a lower but more realistic load than the former experiment.

6.7.3 Packet loss correlation between stations

In the leader-based approach, one of the receivers (called the leader) has to generate acknowledgment frames to the multicast source. The performance of such an approach

[6]Note also that the mean video rate of 512kbps does not include the RTP/UDP/IP/MAC/PHY headers overhead.

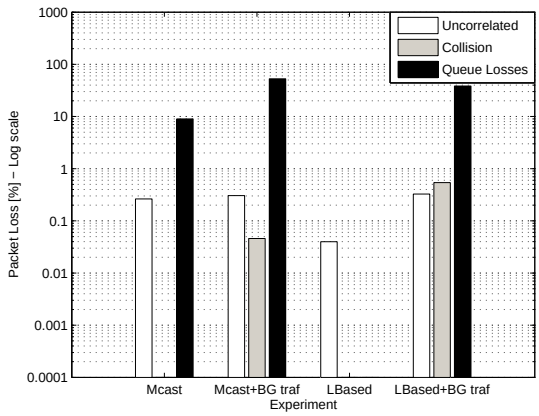

FIGURE 6.10: Multicast Packet loss correlation: Uncorrelated packet losses (independent), Collisions (same packets lost by all the stations) and Queue Losses (AP packet drops) for each of the 4 experiments.

highly depends on the algorithm used to select the leader and on the packet loss correlation between the stations. So, it is crucial to study the packet loss correlation between all the receivers.

When more than one station transmits at the same time, there is a high probability to observe a packet loss in all the stations (i.e., high packet loss correlation). We associate this event to a collision. When a collision is present, the packet is definitely lost when the legacy multicast mechanism is used. However, the leader-based solution should obtain better performance, because all the stations will benefit from the packet retransmission. The other uncorrelated losses can be assigned to background noise or interference, possibly the result of unadapted PHY sending rate for some of the receivers.

Figure 6.10 shows the packet loss profile for both the legacy multicast and the leader-based approach, with and without background traffic. This figure is composed of 4 sets of statistics corresponding to the 4 experiments detailed in Table 6.2. For each experiment, we plot 3 columns corresponding to the different types of packet loss: uncorrelated losses due to interference or noise, collisions, and packets lost at the AP sending queue. Five tests were done under the same conditions for each experiment. The first column comprises the mean value of uncorrelated packet losses between stations, the second column represents the mean number of collisions, and the third column shows the mean number of queue packet drops.

The first columns (in white) for these 4 experiments correspond to the uncorrelated packet loss statistics, including correlated packet losses for 2, 3 and 4 stations. Let us first consider the case when the network is unloaded. We can note that without background traffic, most of the effectively transmitted packets reach the destination: the uncorrelated packet loss is 0.03% for the leader-based approach and reaches about

0.3% for the standard multicast mechanism. This shows that most of the uncorrelated losses in the leader-based approach case are recovered by retransmissions. The leader corresponds to the worst receiver, and consequently experiences most of the losses. This can be observed on Figure 6.11

The second columns (in grey) for each group, of Figure 6.10 show the number of correlated losses between all the stations which are associated to collisions. There are almost no collision for all the experiments made without background traffic. However, in presence of high level of background traffic, the collisions are about 10 times more important in the leader-based approach than for the legacy multicast. The difference is due to the fact that the legacy multicast gets a higher priority than the background traffic, as it is discusses in Section 6.7.1.

The third columns (in black) of Figure 6.10 reflects the number of packets lost at the AP sending queue. In the unloaded case, the legacy multicast flow experiences a high level of packet loss (about 9%), while no packets are dropped with the leader-based solution. This difference is due to the different PHY rates used to send packets in both schemes. With legacy multicast, the PHY transmission mode, fixed at 1Mbps, is not able to support the VBR video stream. On the other hand, the leader-based solution allows to adapt the PHY sending rate to the channel conditions of the leader, which is usually the receiver that experiences the worst channel conditions. When a higher transmission rate is used, the AP queue length decreases. Because it is faster to transmit packets, no more packet loss is observed at the AP. We can observe that the ratio of packets lost at the AP sending queue is about 10% less in the leader-based approach than for the legacy multicast scheme. On both cases the main factor which influences queue growth is the transmission mode. When legacy multicast is used, the AP queue grows because the PHY transmission mode is only 1Mbps. When the leader-based solution is used, 11Mbps transmission mode with ARF is used. This allows for a shorter transmission time which leads to queue size reduction.

Figure 6.11 reflects the individual packet loss for each station when the network is unloaded.

Per-station packet loss is analyzed because it is an important criteria to select the leader. The station with the largest packet loss is likely to be the best candidate, since it will ask for the highest number of retransmissions. We can also observe that packet loss criteria is a better criterion to select the leader than the RSSI mean value. This is reflected from the comparison between Table 6.1 and the individual packet loss from Figure 6.11. In our experiments, STA5 obtains the highestvalue of packet loss. Using the RSSI value criteria, the selected station would have been STA3, instead of STA5.

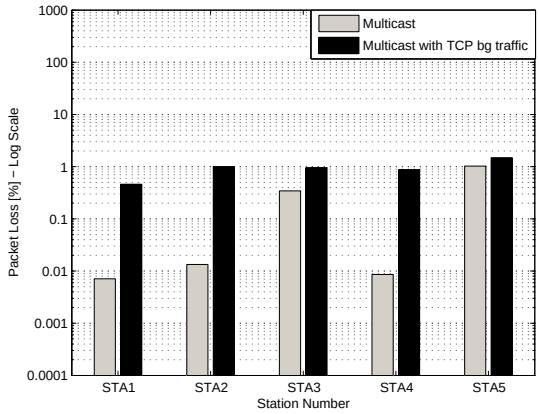

FIGURE 6.11: Per-Station Packet loss without background traffic

6.7.4 How to select the leader

The choice of the leader has a high impact on determine the performance of the leader-based mechanism. In our experiments, we have decided to select the receiver that experiences the worst channel conditions, i.e., STA5. But for a larger group of receivers, we can imagine to select the leader differently, in order not to penalize all the other receivers. In case the set of receivers is very heterogenous, it may also be possible to cluster the receivers in groups that experience similar packet loss as it is proposed in in the SARC [72] algorithm for multicast transmission over the Internet.

To evaluate such an approach, it is important to consider spatial correlation between receivers. The spatial packet loss correlation stands for the amount of packet loss experienced simultaneously by a group of receiving stations. If we observe high packet loss correlation, a leader-based solution for each multicast group could achieve good performance. Let us now also assume that an adaptive sending mechanism is implemented in order to prevent packet loss at the AP sending queue.

In our experiments, two groups of stations have been identified, as described on section 6.6.1 The criteria to cluster the stations are the distance and obstacles between the stations and the AP. Far group stations are located more than 6 meters from the AP and they are also behind a wall. Near group stations reside at the same room as the AP.

Figure 6.12 shows the correlated and uncorrelated packet loss for the two groups of stations using the legacy multicast transmission mechanism, with and without background traffic.

We observe 0.7% of uncorrelated packet losses and about 0.5% of correlated packet losses in the far group. The near group has 0.3% of uncorrelated packet loss and 0.1% of correlated packet losses. With such a low level of packet loss, we do not recommend to

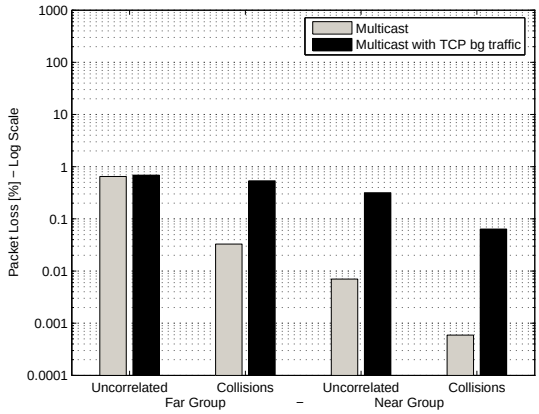

FIGURE 6.12: Multicast packet loss correlation for each group of stations

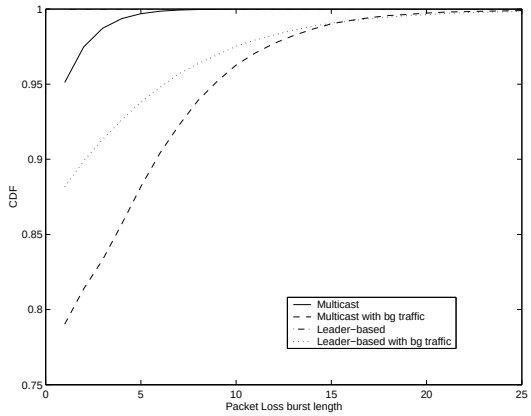

FIGURE 6.13: Cumulative Distribution Function of Packet Error Burst length

implement a multiple leader-based scheme because it does not worth the added complexity. Instead, an application-level FEC mechanism can obtain better performance. FEC mechanisms are efficient in presence of isolated losses or short bursts of packet loss. This is usually what we observe in our experimental results once we remove lost packets at the AP sending queue.

6.7.5 Performance analysis at the application level

In this section we compare the performance obtained at the application level for both mechanisms.

To study video streaming performance, the most important parameters to consider are packet loss and goodput[7].

[7]Latency and jitter parameters have more impact for interactive applications such as VoIP or video-conferencing.

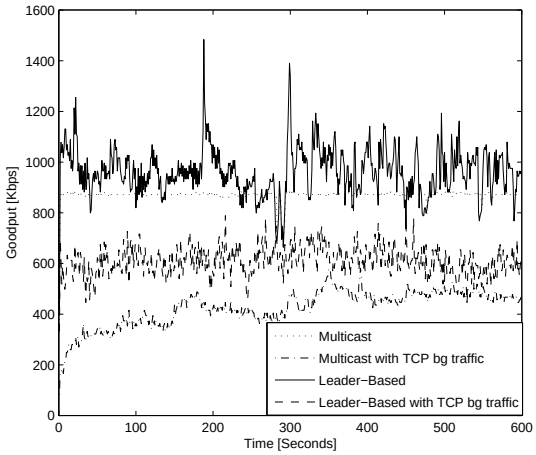

FIGURE 6.14: Video goodput for legacy multicast and leader-based mechanisms

Packet loss for the legacy multicast and the leader-based mechanisms can be observed in Figure 6.10. Now, we study the impact of packet loss on the video receivers. When the network is unloaded, the overall packet loss for the legacy multicast is more than 9% while it is very low (about 0.04% before possible retransmission) for the leader-based approach. In presence on high level congestion, packet loss is higher than 50% for the legacy multicast and it is about 40% for the leader-based approach.

Now, let us compare the corresponding goodput performance for both mechanisms in Figure 6.14.

Without the presence of background traffic, we observe that the leader-based goodput reflects the VBR characteristics of the video sent by vlc. On the other hand, the legacy multicast, using the default PHY transmission mode, is not able to send as much throughput and gets a constant goodput of about 800kbps[8].

In presence of high level of congestion, the goodput at the application level is about 40% less for the leader-based approach and half less for the legacy multicast solution – in the same proportion than what we have observed for the packet loss parameter.

It is important to notice that the bulk of packet loss appears at the AP sending queue. So, the MAC retransmission mechanism used for the leader-based approach will not help much in improving the video quality. With such high level congestion, it is preferable to stop straight transmitting video streams.

We can imagine several ways to prevent such a situation. The video sender should be reactive to packet loss observed (at the RTP level) and should implement a mechanism to adapt its transmission accordingly. The sending queue at the AP could be monitored and a signal should be sent before an overflow occurs, or it could transmit a load indicator

[8]In this case, the multicast source is saturated.

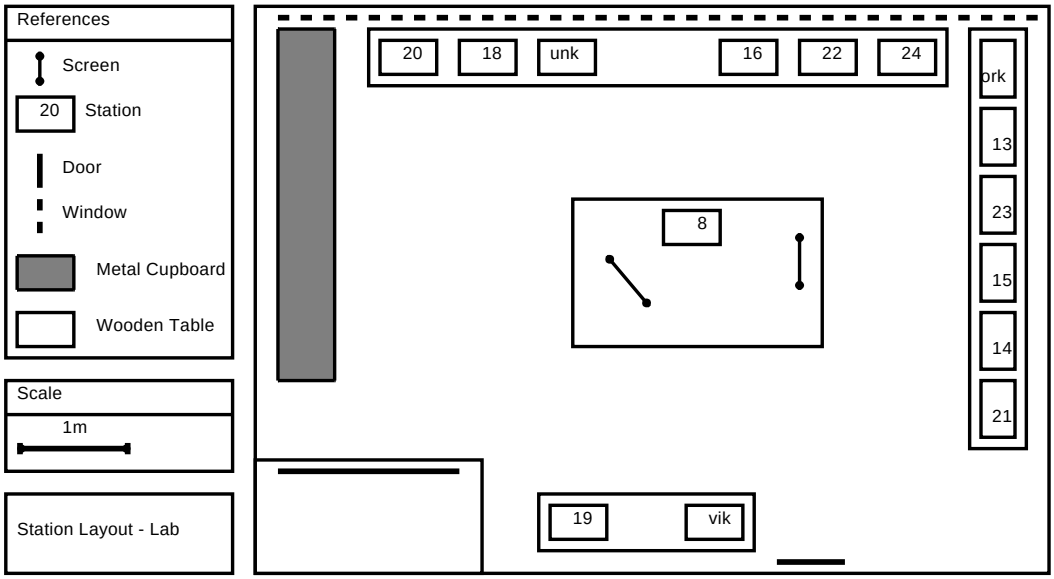

FIGURE 6.15: Layout of the Experimental Lab

in each beacon frame. Another way to estimate channel load is to monitor the average contention window which is a good indicator of channel load.

To summarize, we have described the problems presented by Multicast flows in 802.11 wireless environments, and then we proposed a solution using a leader-based approach. To validate the solution, we generated an experiment and selected representative metrics to further analyse them. The experiment required a sequence of steps and a particular instrumentation. In this first part, the experiments were done using common practices. On the next section, we execute the same experiment using Wextool and a new platform.

6.8 Using Wextool

Up to here, we have described a leader-based multicast experiment using standard common practices from the research community, using personalized scripts and general purpose measurement tools. From now on, we are going to execute the same experiment using a new platform, and the current implementation of the proposed methodology, called Wextool.

The first step, as we have described on chapter 5, is to describe the station layout. The stations are within a closed lab and all of them are in coverage range from each other, so there is no hidden station situation.

The laboratory machine is shown in figure 6.15

The Experimental laboratory has 14 nodes, but for our experiments we use a subgroup of them. The nodes are on top of wooden tables with metal structures beneath. All the stations are at 0.75m height from the floor and -since they are laptops- their screens are in vertical position. For the experiments, we use an external card, then, the screens' antenna is not used. There is a 1.80m metal open cupboard on one side, and there is a window on the back wall which covers from 0.75m height to the ceiling. There are two wooden made doors to the corridor on the front side, and there are two flat panel screens on the center table. To the reference the position at 0 degrees, on the top of the notebook, at 270 degrees there is the network card. The notebooks are aligned to the interior of the lab, on each of the desks, with the 180 degree sense and direction pointing to the internal side of the lab.

6.8.1 Nodes

All the nodes are identified by a number, except the one which is used as an access point, which is called orkney (marked as "ork" on the layout). The wireless nodes are:

- Dell Latitude 830 notebook, 2Gb memory, Intel CPU Core 2 Duo T7500 2.2GHz, Ethernet card Broadcom NetXtreme BCM5755M Gigabit
- D-Link DWL-G630 wireless card

6.8.2 Software

All the stations run:

- Fedora Core 8
- madwifi driver snapshot svn3626
- enabled services: ntp, nfs, ssh.
- wextool agent v0.1
- vlc 0.8.6e (on orkney)
- iperf 2.0.2 (may 2005) pthreads
- TShark 0.99.7

6.8.3 Experiment description

We have recreated the former experiments, whose formal specification is on Appendix A: The participant stations within the scenario are:

- orkney as the AP
- STA14,STA15,STA18 and STA20 as the probes
- STA21 and STA23 as stations when there is traffic generation
- STA19 as the receiver of unicast traffic

We have defined the following experiments:

- **Experiment 1:** Multicast video transmission without competing traffic. The multicast rate is standard at 1Mbps.
- **Experiment 2:** Multicast video transmission with competing traffic. The multicast rate is standard at 1Mbps.
- **Experiment 3:** Unicast link to video transmission without competing traffic. The unicast rate is defined by the ARF rate adaptation algorithm, with no retransmissions.
- **Experiment 4:** Unicast link to video transmission with competing traffic. The unicast rate is defined by the ARF rate adaptation algorithm, with no retransmissions.

The competing traffic is defined by two TCP flows from STA21 and STA23 to the AP. All the experiments have 5 runs each.

6.9 Improvements

The proposed experimental methodology has improved the execution of the leader based multicast on the following aspects:

- **Layout description:** The main items which participate on the experiment are described on a plan and with their location, position and orientation, to scale.

- **Structure and reproducibility:** with a functional specification as we described on section 5.5; all the required steps to execute an experiment are defined within an XML file which provides a description language and a structure, as it can be seen on Appendix A. This did not happen before, and the timings where not as exact nor reproducible.

- **Execution:** The execution was done automatically and scheduled during the specified periods of the day. We recorded the starting and ending timestamps of each programmed task to verify the synchronization of the stations. A throughout sandbox test was done before the launch in order to verify that the experiments would run as expected.

- **Data collection, merging and synchronization:** This process was executed on different forms, to analyze the performance of the experimental lab. Our server has the same hardware as the stations, and the connection of the support network is switched at fast Ethernet (100Mbps). We have tested processing speeds while distributing the Insertion and Synchronization process, (from the PDML description file to the SQL database). We first distributed the insertion process between the stations to observe the scalability capacity. The bottleneck stays on the insertion process at the SQL database: To insert a new packet:

  - We extract from the list of pre-existing packets all of which fit within a time window before and after the timestamp of the new packet.
  - We check if the packet was already captured by another probe with a hash test.
  - If this is the case, we only add an instance to the RADIOTAP table. If not, this is an previously unseen packet.

  We have not used indexing nor SQL server internal optimizations to calculate the processing time from concurrent sources as we show on figure 6.16.

  This process implies a seek action on the whole database which is growing after each insert. The consequence is a polynomial (degree 2) processing time for each of new item to insert. Concurrency multiplies this time by the number of stations inserting packets simultaneously. This calculation provides us the maximal processing time required for a number of packets from several stations in serial and parallel situations for our experimental laboratory layout and 55000 packets per capture file.

  Although these results display that it is possible to insert the packet logs on the database, it suffers from a strong scalability issue. We performed another packet insertion performance test where each station inserted the packets sequentially,

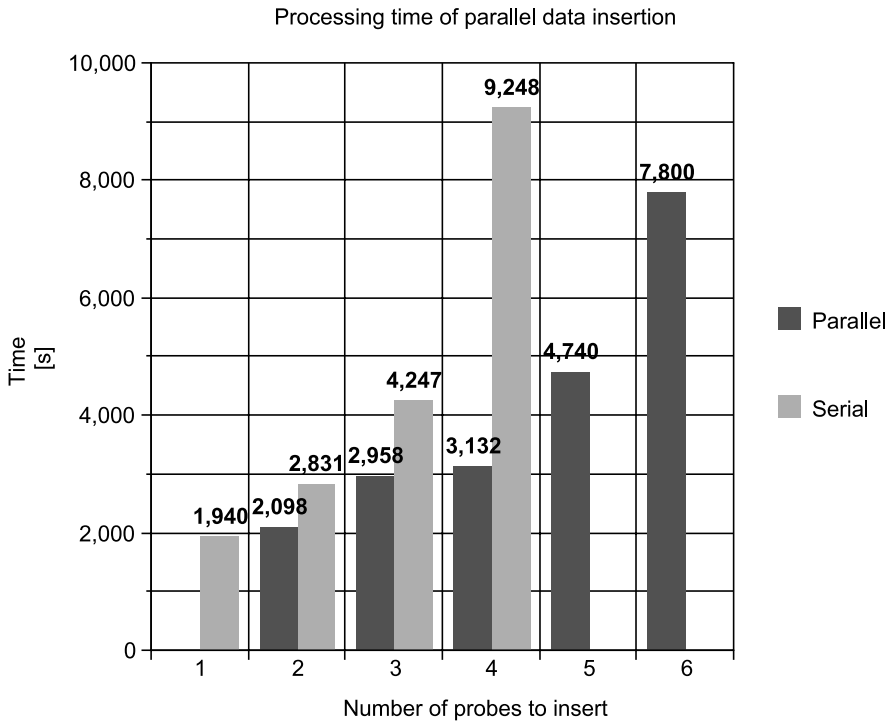

FIGURE 6.16: Processing time for parallel and serial database insertion

meaning at any time, there was only one station inserting packets. Figure 6.17 shows the results. We can observe the increase in processing time for each file insertion, which illustrate the scalability problem.

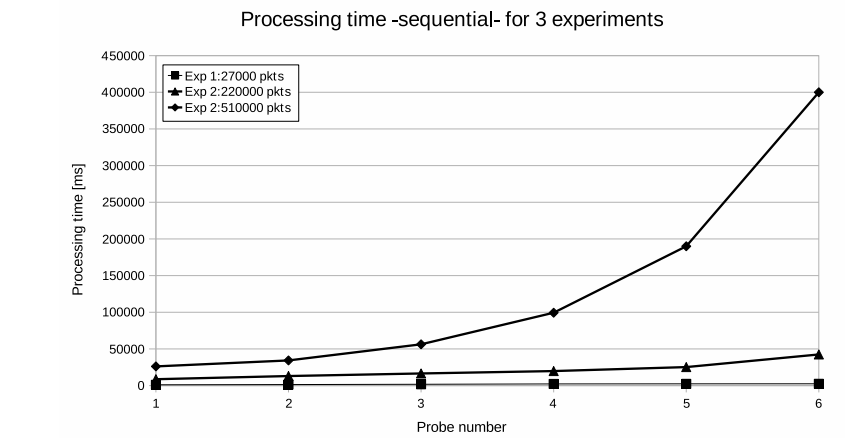

FIGURE 6.17: Processing time for serial database insertion by station

Two main optimizations have been applied to the packet insertion process: indexation of all the tables and the buffering of the packet search responses. On figures 6.18 and 6.19 we can observe the main tasks for packet insertion and their processing time for a cumulative number of processed packets. The only seek task is the only one which increases with the number of inserted packets. After applying

the optimizations, we have arrived to linear increase in processing time with a low angle, which mitigates the scalability problem.

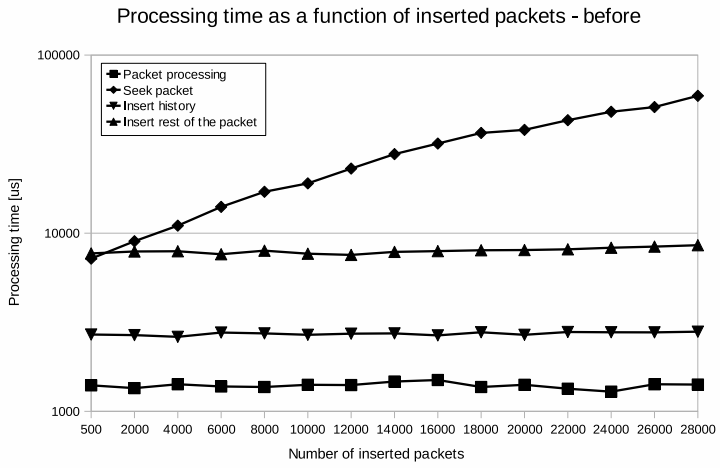

FIGURE 6.18: Processing time by function with number of inserted packets before optimizations

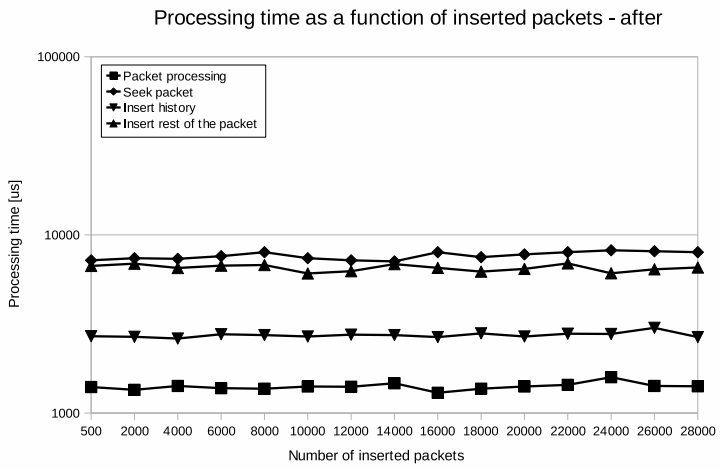

FIGURE 6.19: Processing time by function with number of inserted packets after optimizations

- **Data post-processing:** To calculate the packet loss correlation: We have created a module which does a reverse search on the tables:
  - We start from the RTP sequence number and obtain which packets from the IP table correspond to this table. We follow the same procedure for the LLC,WLAN and RADIOTAP tables.
  - We obtain, for each of the RTP packets, the list of stations which captured that packet.
  - We count the number of packets lost by one probe, so we find the uncorrelated packet loss, due to noise and interference.

– We count the missing packets from the RTP sequence number field, so we find the number of correlated packet loss (a packet not seen by any of the probes). We subtract the number of never-transmitted packets, which are counted as the packet drops at the AP's queue level, to ensure the accuracy of the result.

This method can be reproduced on any other database as long as it respects the same structure, so the same experiment executed on another platform with the same methodology, which follows the same rules, can take advantage of this processing module.

- **Storage:** Finally, the whole process with the layout, configuration and the database containing the data creates a coherent and reusable storage block, which we call "experiment record".

6.10 The experiment results

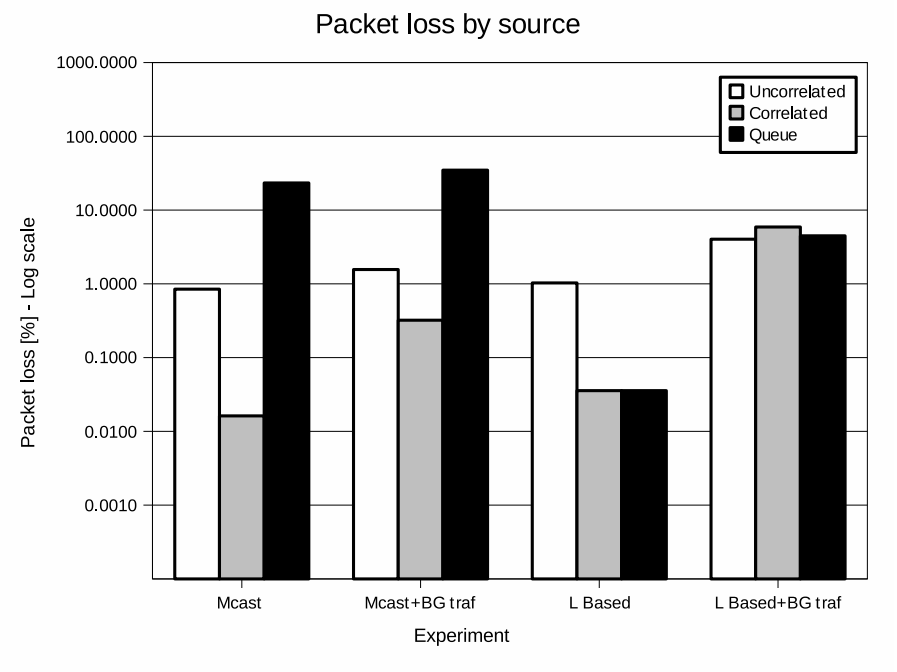

FIGURE 6.20: Packet Loss by source

With a different layout, the experimental results have changed. We can observe the same measurements executed on our new platform on figure 6.20. The first major difference comes from the increase in background noise, represented with the uncorrelated packet loss. This evidences the presence of other wireless networks which interfere with the experiment. Second, the correlated packet loss, which represent the collisions, have

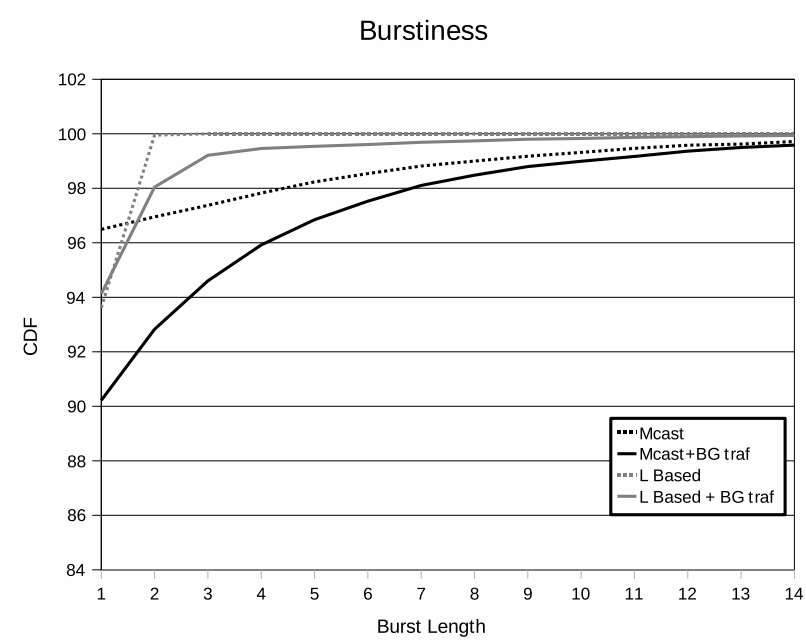

FIGURE 6.21: Burstiness by experiment

increased proportionally on both cases with the presence of background traffic. Nevertheless the queue losses have decreased significatively due to the improvement in the queue management on the wireless driver.

We have also performed a different analysis with the video traffic flows on the packet loss burstiness. As we can observe on figure 6.21, the leader based approach shows more packet loss bursts with shorter duration than the standard multicast approach. The loss of single packets is less harmful to video than the loss of long bursts of packets. Furthermore, although standard multicast has a better transmission reliability due to the low transmission rate, this low rate generates packet drops at the transmitter queue, which is reflected on long packet loss bursts.

6.11 Conclusion

In this chapter we have shown through a use case the difference between an experiment described and executed under the current typical scheme, with an experiment described and executed within our proposed methodology. We have also highlighted the advantages on layout description, configuration, execution, processing and storage, with emphasis on the reproducibility of the experiment and portability of the methods and results.

Furthermore, within the use case itself we analyze with measurements the impact of multicast transmissions on current IEEE 802.11 WLANs. Then, we use those measurements to help in designing the most efficient leader-based mechanism and we compare its performance with the legacy based multicast scheme. This shows the utility and the important decrease of experimentation time due to the automation of the experimental and processing stages. Although the use case permitted monitoring during the experiment, only the packet loss correlation results provided the final numbers. Since the experimentation and processing tasks were automated, failures, errors and mistakes during the sandbox phase could be corrected or avoided within a period of hours instead of a week.

We show with measurements that the legacy multicast transmission scheme is unusable due to its open-loop structure and can significantly degrades performance of other concurrent flows. Then, we evaluate the performance of the recently proposed leader-based mechanism and compare it with the standard multicast solution. Our results clearly show that the new approach outperforms the legacy multicast mechanism while preventing multicast flows to swap all the resources of the WLAN. Such a measurement study is crucially missing in the literature today. For example, the analysis of packet loss correlation is very important to consider while selecting the leader station.

Chapter 7

Conclusion

In this thesis we propose several enhancements to wireless experiments, which are structured within a new experimental methodology. The proposed methodology provides a framework where to describe, execute, measure, process and store the experimental data. We use two main criteria to design the experimental methodology: the ability to reproduce an experiment to the extent of the controllable platform parameters and the generality of application. We analize several problems related to the validation of protocols and algorithms in the wireless environment, which are a consequence of a combination of factors:

- the lack of accurate models due to the complexity of the possible environments,
- the variability of the physical channel due to the wireless propagation characteristics.
- the wireless channel models which respond to a point-to-point design and statistical behavior
- the absence of software tools which provide a framework to use a platform and manage the experimental data

Moreover, we analize the lack of an integrated experimentation tool, from the analysis of the currently available specialized platforms and measurement modules.

The methodology as proposed defines several elements which contribute to the structure of a wireless experiment: description, measurement, processing and storage.

First, we consider the environmental conditions: The wireless experimentation environment is a phenomena circumscribed to a limited scenario during a period of time. As

a solution, we propose a method to describe the experimentation physical layout (space dimension) and activity schedule (time dimension) during the experiment which is used also as a specification to execute the experiment. This double role of the description is aimed towards the possibility to reproduce the experiment, in order to trace the differences of the controllable parameters to the original experiment.

Second, measurements done within the scenario correspond to wireless events and to aggregate data, and both together describe to the best possible, the traffic evolution and the node behavior. We propose a method to schedule the activities during the experiment and the measurements which acknowledge wireless cards as measurement instruments: the probe point of view, the software instrumentation and the synchronization requirements.

Third, after all the data has been captured, the processing stage must take place, so as to build a unique sequence of events from the time dimension and a unique geographical point of view, from the space dimension of the experiment, while fusioning the event-based data and aggregate data. We propose a method to build a single database in which the captured data keeps the source identity as the spatial dimension while it is synchronized to a single timebase to keep the time dimension, and furthermore, provides a data structure which stores the events (for example, the captured packets) in tables related and classified by occurrence from the different probing points.

And fourth, once the processed data is made available, the environment process description, and the data itself must be packed and stored together. This element enables traceability from the results to the captured data and the experimental conditions where they were captured, while further enables the reproduction of the experiments with the same schedule but different experimental conditions.

Finally, we create an implementation of the proposed methodology called Wextool, which integrates each of the steps described earlier; and we describe a use case sensible to the critical PHY and MAC layers comparing the traditional approach to one which uses the proposed methodology. To conclude, the improvement we show in this thesis opens the doors to experimental validation using the scientific method in wireless networks, an points towards the bases to compare experiments under standardized and known conditions.

The future evolution of this thesis can be oriented into the following possible research paths:

- The improvement of the experimental condition measurement through uniform test cases which extract representative parameters from the wireless enviroment, in order to perform the platform calibration task.
- The exchange and integration with simulator engines to translate experimental conditions into simulation scripts.
- The evaluation of traffic sampling and tracking techniques as applied to the current database structure
- The creation of a standardized data repository of experimental data with anonimization capabilities, to allow remote studies on the available databases
- The standardization of the module creation to extract data from the stored databases.

As a final remark, the convergence of several disciplines from the rise of wireless networking research is a reality. The decoupling between networking and digital communication channel research is not more the case since there is a strong interaction between both communication research branches. Furthermore, electromagnetic propagation has a main role as the source of wireless channel variability which further propagates to upper layers. Consequently, experimental validation is the natural path to follow since the complexity of a deterministic physical model results in excessive computational requirements and statistical channel models hide into their generality the characteristics of variations only present during an experiment.

Appendix A

Formal description of the multicast experiment

The description file for the 1Mbps multicast with no competing treffic experiment follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<!– Multicast experiment, 1Mbps –>
 <Node>
   <id>1</id>
   <host>orkney</host>
   <user>root</user>
   <password>xx</password>
   <port>22</port>
   <role>station</role>
 </Node>
 <Node>
   <id>2</id>
   <host>radio14</host>
   <user>root</user>
   <password>xx</password>
   <port>22</port>
   <role>station</role>
</Node>
<Node>
   <id>3</id>
   <host>radio15</host>
   <user>root</user>
```

```
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>4</id>
    <host>radio16</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>5</id>
    <host>radio18</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>6</id>
    <host>radio19</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>7</id>
    <host>radio13</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>8</id>
    <host>radio20</host>
```

```
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>9</id>
    <host>radio21</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Node>
    <id>10</id>
    <host>radio23</host>
    <user>root</user>
    <password>xx</password>
    <port>22</port>
    <role>station</role>
</Node>
<Action>
    <id>1</id>
    <nodeid>1</nodeid>
    <application>/root/radio/video</application>
    <param> </param>
    <redirection_err>pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirectio_result>
</Action>
<Action>
    <id>2</id>
    <nodeid>3</nodeid>
    <application>/usr/sbin/tshark</application>
    <param>-i wlan1 -w /root/radio/test_mcast_100.dump</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>3</id>
```

```
    <nodeid>2</nodeid>
    <application>/usr/sbin/tshark</application>
    <param>-r /root/radio/test_mcast_100.dump -T pdml >/root/radio/test_mcast_100.pdml</paran
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>4</id>
    <nodeid>5</nodeid>
    <application>/usr/bin/SIMPLEXML</application>
    <param>/root/radio/test_mcast_100.pdml rtp100 root server wlab05</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>5</id>
    <nodeid>5</nodeid>
    <application>./root/radio/wireless_config_monitor</application>
    <param> </param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>6</id>
    <nodeid>5</nodeid>
    <application>./root/radio/wireless_config_station</application>
    <param>192.168.1.14</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>7</id>
    <nodeid>5</nodeid>
    <application>./root/radio/wireless_config_ap</application>
    <param>-i wlan0 -w test_100.dump</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
```

```
    <id>8</id>
    <nodeid>5</nodeid>
    <application>route</application>
    <param>add 224.0.0.1 dev ath0</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Action>
    <id>9</id>
    <nodeid>5</nodeid>
    <application>athlog</application>
    <param>mcast100.txt</param>
    <redirection_err> pas de redirection</redirection_err>
    <redirection_result>pas de redirection</redirection_result>
</Action>
<Act>
    <!– Preconfiguration of nodes in ap mode –>
    <id>1</id>
    <nbnode>1</nbnode>
    <nodeid>1</nodeid>
    <actionid>7</actionid>
    <start_time>10</start_time>
    <!– This act starts with 10 seconds offset –>
    <duration>10</duration>
    <!– This act lasts 10 seconds –>
    <!– Configuration should not take longer –>
</Act>
<Act>
    <!– Preconfiguration of nodes in monitor mode –>
    <id>2</id>
    <nbnode>9</nbnode>
    <nodeid>2</nodeid>
    <nodeid>3</nodeid>
    <nodeid>4</nodeid>
    <nodeid>5</nodeid>
    <nodeid>6</nodeid>
    <nodeid>7</nodeid>
    <nodeid>8</nodeid>
    <nodeid>9</nodeid>
```

```
    <nodeid>10</nodeid>
    <actionid>5</actionid>
    <start_time>10</start_time>
    <!– This act starts with 10 seconds offset –>
    <duration>10</duration>
    <!– This act lasts 10 seconds –>
    <!– Configuration should not take longer –>
</Act>
<Act>
    <!– Start capture –>
    <id>3</id>
    <nbnode>9</nbnode>
    <nodeid>2</nodeid>
    <nodeid>3</nodeid>
    <nodeid>4</nodeid>
    <nodeid>5</nodeid>
    <nodeid>6</nodeid>
    <nodeid>7</nodeid>
    <nodeid>8</nodeid>
    <nodeid>9</nodeid>
    <nodeid>10</nodeid>
    <actionid>2</actionid>
    <start_time>20</start_time>
    <!– This act starts with 10 seconds offset –>
    <duration>350</duration>
    <!– This act lasts 360 seconds –>
</Act>
<Act>
    <!– Add mcast route on Access Point –>
    <id>4</id>
    <nbnode>1</nbnode>
    <nodeid>1</nodeid>
    <actionid>8</actionid>
    <start_time>20</start_time>
    <!– This act starts with 15 seconds offset –>
    <duration>10</duration>
    <!– This act lasts 5 seconds –>
</Act>
<Act>
```

```
    <!– UDP Multicast generator on Access Point –>
    <id>5</id>
    <nbnode>1</nbnode>
    <nodeid>1</nodeid>
    <actionid>1</actionid>
    <start_time>30</start_time>
    <!– This act starts with 30 seconds offset –>
    <duration>330</duration>
    <!– This act lasts 330 seconds –>
</Act>
<Act>
    <!– Conversion from dump to pdml –>
    <id>6</id>
    <nbnode>9</nbnode>
    <nodeid>2</nodeid>
    <nodeid>3</nodeid>
    <nodeid>4</nodeid>
    <nodeid>5</nodeid>
    <nodeid>6</nodeid>
    <nodeid>7</nodeid>
    <nodeid>8</nodeid>
    <nodeid>9</nodeid>
    <nodeid>10</nodeid>
    <actionid>3</actionid>
    <start_time>350</start_time>
    <!– This act starts with 350 seconds offset –>
    <!– After the end of the experiment, to convert to PDML file –>
    <duration>300</duration>
    <!– This act lasts 5 minutes –>
</Act>
<Act>
    <!– Packet insertion on database –>
    <id>7</id>
    <nbnode>9</nbnode>
    <nodeid>2</nodeid>
    <nodeid>3</nodeid>
    <nodeid>4</nodeid>
    <nodeid>5</nodeid>
    <nodeid>6</nodeid>
```

```
    <nodeid>7</nodeid>
    <nodeid>8</nodeid>
    <nodeid>9</nodeid>
    <nodeid>10</nodeid>
    <actionid>4</actionid>
    <start_time>660</start_time>
    <!– This act starts with 660 seconds offset –>
    <!– After the end of the experiment, to insert packets –>
    <duration>300</duration>
    <!– This act lasts 5 minutes –>
</Act>
<Act>
    <!– Capture aggregate statistics from wireless card –>
    <id>8</id>
    <nbnode>1</nbnode>
    <nodeid>1</nodeid>
    <actionid>9</actionid>
    <start_time>30</start_time>
    <!– This act starts with 30 seconds offset –>
    <duration>330</duration>
    <!– This act lasts 330 seconds –>
</Act>
<Scenario>
    <id>1</id>
    <nbact>9</nbact>
    <actid>1</actid> <!– ap mode –>
    <actid>2</actid> <!– config monitor –>
    <actid>3</actid> <!– start capture –>
    <actid>4</actid> <!– add route –>
    <actid>5</actid> <!– generate traffic –>
    <actid>6</actid> <!– conversion to PDML –>
    <actid>8</actid> <!– capture aggregate statistics –>
    <actid>9</actid> <!– kill vlc –>
    <actid>10</actid> <!– kill athstats –>
</Scenario>
<Experience>
    <id>1</id>
    <runs>5</runs>
    <nbnode>5</nbnode>
```

```
    <nodeid>1</nodeid>
    <nodeid>2</nodeid>
    <nodeid>3</nodeid>
    <nodeid>5</nodeid>
    <nodeid>8</nodeid>
    <Scenarioid>1</Scenarioid>
</Experience>
```

Appendix B

Publications

B.1 Conferences, Journals

During the development of this thesis, the following works were published:

D. Dujovne and T. Turletti, "Multicast in 802.11 WLANs: an experimental study," Proc. of the 9th ACM international Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM), Torremolinos, Spain, October 2-6, 2006. pp. 130-138.

D. Dujovne, T. Turletti and Walid Dabbous, Experimental Methodology for Real Overlays ROADS Sigcomm Workshop, Warsaw, July 11-12 2007.

D. Dujovne, T. Turletti and F. Filali, A Taxonomy of IEEE 802.11 Wireless Parameters and Open Source Measurement Tools, IEEE Communications Society Journal on Surveys and Tutorials, to be published on 2nd Quarter 2009.

Juan-Carlos Maureira, Diego Dujovne and Olivier Dalle,"Generation of Realistic 802.11 Interferences in the Omnet++ INET Framework Based on Real Traffic Measurements", OMNet++ 2009 Workshop, Rome, 2009

B.2 Proposals at the IEEE 802.11aa TG

Y. Seok, T.Turletti, D.Dujovne, E. Qi, A. Stephens, A. Ashley, M. Wentink, P. Cuenca "Leader based Multicast Service Proposal, Proposal to improve multicast transmission on WiFi networks (v2)" , IEEE 802.11-07/2127r0, July 17th 2007.

Y. Seok, D.Dujovne, T.Turletti, E. Qi, A. Stephens, A. Ashley, M. Wentink, P. Cuenca "Leader based Multicast Service Proposal, Proposal to improve multicast transmission on WiFi networks (v2)", 802.11-07/0144r3, March 15th 2007.

Y. Seok, D.Dujovne, T.Turletti, P. Cuenca "Leader based Multicast Service Proposal, Proposal to improve multicast transmission on WiFi networks (v1)", IEEE 802.11-07/0120r6, January 16th 2007.

B.3 Registered Applications

Wismon, INRIA Sophia Antipolis, (2005)

Wextool, INRIA Sophia Antipolis, (2007)

Bibliography

[1] http://www.3gpp.org/specifications [Last visited: February 2009]

[2] IEEE Standards committee, “IEEE Standard for Local and Metropolitan Area Networks: Part 16: Air interface for fixed broadband wireless access systems“, June 2004.

[3] IEEE Standards committee, ”IEEE Standard for Information technology Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements: Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)“, September 2006.

[4] Lassabe, F.; Canalda, P.; Chatonnay, P.; Spies, F., ”A Friis-based calibrated model for WiFi terminals positioning,” World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a , vol., no., pp. 382-387, 13-16 June 2005

[5] Yahong Rosa Zheng; Chengshan Xiao, ”Simulation models with correct statistical properties for Rayleigh fading channels,” Communications, IEEE Transactions on , vol.51, no.6, pp. 920-928, June 2003

[6] Longley, A. G. ; Rice, P. L.,”PREDICTION OF TROPOSPHERIC RADIO TRANSMISSION LOSS OVER IRREGULAR TERRAIN. A COMPUTER METHOD-1968“,INSTITUTE FOR TELECOMMUNICATION SCIENCES BOULDER CO, JUL 1968

[7] Juan-Carlos Maureira, Diego Dujovne and Olivier Dalle,”Generation of Realistic 802.11 Interferences in the Omnet++ INET Framework Based on Real Traffic Measurements“, OMNet++ 2009 Workshop, Rome, 2009

[8] IEEE Standards commitee, ”IEEE Standard for Information technology Telecommunications and information exchange between systems Local and metropolitan area networks Specific requirements:Part 15.1: Wireless medium access control (MAC)

and physical layer (PHY) specifications for wireless personal area networks (WPANs)“, June 2005.

[9] D. Kotz, C. Newport, R. Gray, J. Liu, Y. Yuan, and C. Elliott, ”Experimental evaluation of wireless simulation assumptions,” in Int’l Workshop Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM 04). ACM Press, New York, Oct. 2004, p. to appear

[10] Shakkottai, S.; Rappaport, T.S.; Karlsson, P.C., ”Cross-layer design for wireless networks,” Communications Magazine, IEEE , vol.41, no.10, pp. 74-80, Oct 2003

[11] Heidemann, J.; Mills, K.; Kumar, S., ”Expanding confidence in network simulations,” Network, IEEE , vol.15, no.5, pp.58-63, Sep/Oct 2001

[12] G. D. Durgin, T. S. Rappaport, D. A. de Wolf, New Analytical Models and Probability Density Functions for Fading in Wireless Communications, IEEE Transactions on Communications, Vol. 50, No. 6, June 2002, pp. 1005 - 1015.

[13] Wireless Communications: Principles & Practice,Theodore S. Rappaport, Prentice Hall, 2002. ISBN 0-13-042232-0

[14] http://www.isi.edu/nsnam/ns/doc [Last visited: July 2008]

[15] http://www.omnetpp.org/ [Last visited: Feb 2009]

[16] Punnoose, R.J.; Nikitin, P.V.; Stancil, D.D., ”Efficient simulation of Ricean fading within a packet simulator,” Vehicular Technology Conference, 2000. IEEE VTS-Fall VTC 2000. 52nd , vol.2, no., pp.764-767 vol.2, 2000

[17] Pearson, Bob, ”Complementary Code Keying Made Simple“, Intersil Application Note AN9850.1, May 2000.

[18] Weinstein, S.; Ebert, P., ”Data Transmission by Frequency-Division Multiplexing Using the Discrete Fourier Transform,” Communication Technology, IEEE Transactions on , vol.19, no.5, pp.628-634, October 1971

[19] Gesbert, D.; Shafi, M.; Da-shan Shiu; Smith, P.J.; Naguib, A., ”From theory to practice: an overview of MIMO space-time coded wireless systems,” Selected Areas in Communications, IEEE Journal on , vol.21, no.3, pp. 281-302, Apr 2003

[20] Takai, M., Martin, J., and Bagrodia, R. 2001. ”Effects of wireless physical layer modeling in mobile ad hoc networks“ In Proceedings of the 2nd ACM international Symposium on Mobile Ad Hoc Networking; Computing (Long Beach, CA, USA, October 04 - 05, 2001). MobiHoc ’01. ACM, New York, NY, 87-94.

[21] Svilen Ivanov, Andr Herms, and Georg Lukas, "Experimental Validation of the ns-2 Wireless Model using Simulation, Emulation, and Real Network", Proceedings of the 4th Workshop on Mobile Ad-Hoc Networks (WMAN'07), pp. 433 - 444, Feb. 26. - March 2., 2007

[22] Cappello, F.; Caron, E.; Dayde, M.; Desprez, F.; Jegou, Y.; Primet, P.; Jeannot, E.; Lanteri, S.; Leduc, J.; Melab, N.; Mornet, G.; Namyst, R.; Quetier, B.; Richard, O., "Grid'5000: a large scale and highly reconfigurable grid experimental testbed," Grid Computing, 2005. The 6th IEEE/ACM International Workshop on , vol., no., pp. 8 pp.-, 13-14 Nov. 2005

[23] I. Stepanov, D. Herrscher, K. Rothermel, On the impact of radio propagation models on MANET simulation results, in: Proceedings of the 7th IFIP International Conference on Mobile and Wireless Communications Networks (MWCN 2005), Marrakech, Morocco, September 2005.

[24] S. Kurkowski, T. Camp, and M. Colagrosso, MANET Simulation Studies: The Current State and New Simulation Tools, Technical Report MCS-05-02, The Colorado School of Mines, February 2005

[25] Lacage, M. and Henderson, T. R. 2006. Yet another network simulator. In Proceeding From the 2006 Workshop on Ns-2: the IP Network Simulator (Pisa, Italy, October 10 - 10, 2006). WNS2 '06, vol. 202. ACM, New York, NY, 12.

[26] Henderson, T. R., Roy, S., Floyd, S., and Riley, G. F. 2006. ns-3 project goals. In Proceeding From the 2006 Workshop on Ns-2: the IP Network Simulator (Pisa, Italy, October 10 - 10, 2006). WNS2 '06, vol. 202. ACM, New York, NY, 13.

[27] Zhan Yu; Chin Choy Chai; Tjeng Thiang Tjhung, "Envelope Probability Density Functions for Fading Model in Wireless Communications," Vehicular Technology, IEEE Transactions on , vol.56, no.4, pp.1907-1912, July 2007

[28] W. Xiuchao. Simulate 802.11b channel within ns2. http://www.comp.nus.edu.sg/~wuxiucha/research/reactive/report/80211ChannelinNS2_new.pdf

[29] Vyas, Amit K.; Tobagi, Fouad A.; Narayanan, Rajesh, "WLC34-6: Characterization of an IEEE 802.11a Receiver using Measurements in an Indoor Environment," Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE , vol., no., pp.1-6, Nov. 2006

[30] J. Robinson, K. Papagiannaki, C. Diot, X. Guo, L. Krishnamurthy, Experimenting with a Multi-Radio Mesh Networking Testbed, in Proc. of WinMee05, April 2005

[31] Takai, M., Bagrodia, R., Tang, K., and Gerla, M. 2001. Efficient wireless network simulations with detailed propagation models. Wirel. Netw. 7, 3 (May. 2001), 297-305.

[32] Portoles-Comeras, M.; Mangues-Bafalluy, J.; Requena-Esteso, M., "Multi-radio based active and passive wireless network measurements," Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on , vol., no., pp. 1-6, 03-06 April 2006

[33] David Clark, Scott Shenker, Aaron Falk, GENI Research Plan GDD-06-28, GENI: Global Environment for Network Innovations, Version 4.5 of April 23, 2007.

[34] Tianlin Wang; Refai, H.H., "Network performance analysis on IEEE 802.11g with different protocols and signal to noise ratio values," Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on , vol., no., pp. 29-33, 6-8 March 2005

[35] Giustiniano, D.; Malone, D.; Leith, D.J.; Papagiannaki, K., "Experimental assessment of 802.11 MAC layer channel estimators," Communications Letters, IEEE , vol.11, no.12, pp.961-963, December 2007

[36] Haris Kremo, Ivan Seskar, and Predrag Spasojevic, "An ORBIT Testbed Study of 802.11b DCF:Throughput, Latency, and the Capture Effect", Proceedings of IEEE Tridentcom 2006, Barcelona, Spain, March 1-3, 2006

[37] Giuseppe Anastasi, Eleonora Borgia, Marco Conti, Enrico Gregori, "Wi-Fi in Ad Hoc Mode: A Measurement Study," percom, p. 145, Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04), 2004

[38] Shakkottai, S.; Rappaport, T.S.; Karlsson, P.C., "Cross-layer design for wireless networks," Communications Magazine, IEEE , vol.41, no.10, pp. 74-80, Oct 2003

[39] Rayanchu, S.; Mishra, A.; Agrawal, D.; Saha, S.; Suman Banerjee, "Diagnosing Wireless Packet Losses in 802.11: Separating Collision from Weak Signal," INFOCOM 2008. The 27th Conference on Computer Communications. IEEE , vol., no., pp.735-743, 13-18 April 2008

[40] Chen, Q., Schmidt-Eisenlohr, F., Jiang, D., Torrent-Moreno, M., Delgrossi, L., and Hartenstein, H. 2007. Overhaul of ieee 802.11 modeling and simulation in ns-2. In Proceedings of the 10th ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (Chania, Crete Island, Greece, October 22 - 26, 2007). MSWiM '07. ACM, New York, NY, 159-168.

[41] Purushothaman I, Roy S Infrastructure mode support for IEEE 802.11 implementation in NS-2.

[42] Ryu, J., Lee, J., Lee, S., and Kwon, T. 2008. Revamping the IEEE 802.11a PHY simulation models. In Proceedings of the 11th international Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (Vancouver, British Columbia, Canada, October 27 - 31, 2008). MSWiM '08. ACM, New York, NY, 28-36

[43] Chen, Q., Jiang, D., Taliwal, V., and Delgrossi, L. 2006. IEEE 802.11 based vehicular communication simulation design for NS-2. In Proceedings of the 3rd international Workshop on Vehicular Ad Hoc Networks (Los Angeles, CA, USA, September 29 - 29, 2006). VANET '06. ACM, New York, NY, 50-56.

[44] Ilango Purushothaman, Sumit Roy, "Technical Report - IEEE 802.11 implementation issues/bugs in ns2", Department of EE, University of Washington, 2007.

[45] Lee, J. and Chen, M. C. 2008. "Sniffing out correct error frame model of ns-2 simulator". In Proceedings of the 11th Communications and Networking Simulation Symposium (Ottawa, Canada, April 14 - 17, 2008). CNS '08. ACM, New York, NY, 23-29.

[46] Shyamnath Gollakota and Dina Katabi,"ZigZag Decoding: Combating Hidden Terminals in Wireless Networks" SIGCOMM08, Seattle, Washington, USA, August 1722, 2008.

[47] Tristan Henderson and David Kotz and Ilya Abyzov. The Changing Usage of a Mature Campus-wide Wireless Network. In Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking (MobiCom), pages 187-201, September, 2004.

[48] Bicket, J., Aguayo, D., Biswas, S., and Morris, R. 2005. Architecture and evaluation of an unplanned 802.11b mesh network. In Proceedings of the 11th Annual international Conference on Mobile Computing and Networking (Cologne, Germany, August 28 - September 02, 2005). MobiCom '05.

[49] An Experimentation Workbench for Replayable Networking Research. Eric Eide, Leigh Stoller, and Jay Lepreau. In Proceedings of the Fourth USENIX Symposium on Networked Systems Design and Implementation (NSDI 2007), pages 215-228, Cambridge, MA, April 2007.

[50] David Kotz and Kobby Essien. Analysis of a Campus-wide Wireless Network. In Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking (MobiCom), pages 107-118, September, 2002.

[51] http://www.emulab.net/[Last visited: October 2007]

[52] Raychaudhuri D., Seskar I., Ott M., Ganu S., Ramachandran K., Kremo H., Siracusa R., Liu H. and Singh M., "Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols", WCNC'05, March 2005

[53] http://www.planet-lab.org/[Last visited: October 2007]

[54] http://one-lab.org/wiki/view/OneLab[Last visited: October 2007]

[55] http://www.paris-traceroute.net/ [Last visited: October 2007]

[56] http://www.nbee.org/[Last visited: October 2007]

[57] http://www.nbee.org/doku.php?id=netpdl:pdml_specification

[58] Codd, E.F. "The Relational Model for Database Management Version 2", Addison Wesley, 1990.

[59] N. Choi, J. Ryu, Y. Seok, Y. Choi, and T. Kwon, "Unicast-Friendly Multicast in IEEE 802.11 Wireless LANs", IEEE CCNC, Jan 2006.

[60] S. Gopal, D. Raychaudhuri, "Experimental Evaluation of the TCP Simultaneous Send Problem in 802.11 Wireless Local Area Networks", ACM SIGCOMM E-WIND workshop, August 2005.

[61] S. K. S. Gupta, V. Shankar, and S. Lalwani, Reliable Multicast MAC Protocol for Wireless LANs, wIEEE International Conference on Communications, Vol. 1, 2003, pp. 93-97.

[62] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b", IEEE Infocom, April 2003.

[63] C. Hoffmann, M.H. Manshaei, and T. Turletti, "CLARA: Closed-Loop Adaptive Rate Allocation for IEEE 802.11 WLANs", IEEE WirelessCom'05, Hawai, USA, June 13-16, 2005.

[64] A. Kamerman and L. Monteban, "WaveLAN II: A high-performance wireless LAN for the unlicensed band", Bell Labs technical Journal, Tech. Rep., Summer 1997.

[65] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and Chip Elliott, "Experimental evaluation of wireless simulation assumptions", ACM MSWiM, October 2004.

[66] J. Kuri and S.K. Kasera, "Reliable multicast in multi-access wireless LANs", Wireless Networks, Vol. 7, No. 4, pp. 359-369, July 2001.

[67] J. Lacan and T. Perennou, "Evaluation of Error Control Mechanisms for 802.11b Multicast Transmissions", WinMee'06, April 2006.

[68] W. Lauppe, "Multicast Video over Wireless LANs", Internship report, 'Ecole Polytechnique Promotion 2002, July 2005.

[69] A. Ng, D.J.Leith, and D.W. Malone, "Experimental Evaluation of TCP Performance and Fairness in an 802.11e Test-bed", ACM SIGCOMM E-WIND workshop, August 2005.

[70] A. Majumdar, D.G. Sachs, IV. Kozintsez, K. Ramchandran, and M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs , IEEE Transactions on CSVT, Vol. 12, No. 6, June 2002.

[71] H. Schulzrinne et al., "RTP: A Transport Protocol for Real-Time Applications", RFC-3550, July 2003.

[72] J. Vieron, T. Turletti, K. Salamatian, and C. Guillemot, "Source and channel adaptive rate control for multicast layered video transmission based on a clustering algorithm", in EURASIP Journal on Applied Signal Processing, Vol. 2004, No. 2, pp. 158-175, February 2004.

[73] J. Yeo, M. Youssef, A. Agrawala, "Characterizing the IEEE 802.11 Traffic: The Wireless Side", CS-TR-4570, March 2004.

[74] [web page] Ethereal, tethereal: http://www.ethereal.com [Accessed 15 May 2006]

[75] Ethereal Users List [web page]: http://www.ethereal.com/lists/ethereal-users/200405/msg00305.html [Accessed 15 May 2006]

[76] Ethereal supported filters[web page]: http://www.ethereal.com/docs/dfref/ [Accessed 15 May 2006]

[77] iperf [web page]: http://dast.nlanr.net/Projects/Iperf/ [Accessed 15 May 2006]

[78] Kismet sniffer [web page]: http://www.kismetwireless.net [Accessed 15 May 2006]

[79] libpcap [web page]: http://www.tcpdump.org [Accessed 15 May 2006]

[80] Madwifi driver [web page]: http://www.madwifi.org/ [Accessed 15 May 2006]

[81] tcpdump [web page]: http://www.tcpdump.org [Accessed 15 May 2006]

[82] tcpflow [web page]: http://www.circlemud.org/ jelson/software/tcpflow/ [Accessed 15 May 2006]

[83] tcptrace [web page]: http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html [Accessed 15 May 2006]

[84] tstat [web page]: TCP statistic and analysis tool: http://tstat.tlc.polito.it/ [Accessed 15 May 2006]

[85] VideoLan Client (VLC) [web page]: http://www.videolan.org/vlc/ [Accessed 15 May 2006]

[86] WisMon [web page]: http://www-sop.inria.fr/planete/software/WisMon/ [Accessed 15 May 2006]

[87] Rajagopalan, R., Westerink, P.H., "Two-pass MPEG-2 variable-bit-rate encoding", IBM Journal of Research & Development; Jul99, Vol. 43 Issue 4, p471.

[88] Libpcap format definition [web page]: http://wiki.wireshark.org/Development/LibpcapFileFormat [Accessed October 2008]

[89] PCAP Next Generation Dump File Format [web page]: http://www.winpcap.org/ntar/draft/PCAP-DumpFileFormat.html [Accessed October 2008]

[90] Judd, G.; Steenkiste, P., "Design and Implementation of an RF Front End for Physical Layer Wireless Network Emulation," Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th , vol., no., pp.974-979, 22-25 April 2007

[91] Bianchi, G.; Di Stefano, A.; Giaconia, C.; Scalia, L.; Terrazzino, G.; Tinnirello, I., "Experimental Assessment of the Backoff Behavior of Commercial IEEE 802.11b Network Cards," INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE , vol., no., pp.1181-1189, 6-12 May 2007

[92] Kiess, G. Ogilvie, T. Mauve, M., "The EXC Toolkit for Real-World Experiments with Wireless Multihop Networks", EXPONWIRELESS 2008: Proceeding of the 3rd Workshop on Advanced Experimental Activities on Wireless Networks and Systems, Newport Beach, California, USA, June 2008.

[93] Wolfgang Kiess, PhdThesis, "On Real-World Experiments with Wireless Multihop Networks - Design, Realization, and Analysis",Heinrich Heine University, Dsseldorf, Germany, Jun 2008

[94] Bianchi, G.; Di Stefano, A.; Giaconia, C.; Scalia, L.; Terrazzino, G.; Tinnirello, I., "Experimental Assessment of the Backoff Behavior of Commercial IEEE 802.11b Network Cards," INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE , vol., no., pp.1181-1189, 6-12 May 2007

[95] Di Stefano, A., Terrazzino, G., Scalia, L., Tinnirello, I., Bianchi, G., and Giaconia, C. 2006. An Experimental Testbed and Methodology for Characterizing IEEE 802.11 Network Cards. In Proceedings of the 2006 international Symposium on on World of Wireless, Mobile and Multimedia Networks (June 26 - 29, 2006). International Workshop on Wireless Mobile Multimedia. IEEE Computer Society, Washington, DC, 513-518.

[96] I. Tinnirello, D. Giustiniano, L. Scalia, G. Bianchi, "On the side-eff ects of proprietary solutions for fading and interference mitigation in IEEE 802.11b/g outdoor links", accepted for pubblication in Elsevier Computer Network Journal

[97] Yu J. and Buyya R., "A Taxonomy of Workflow Management Systems for Grid Computing", Technical Report, GRIDS-TR-2005-1, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia, March 10, 2005.

[98] Eide E., Stoller L. and Lepreau J.: "An Experimentation Workbench for Replayable Networking Research", Fourth USENIX Symposium on Networked Systems Design and Implementation (NSDI '07), Cambridge, MA, Apr. 2007.

[99] Wang, Y., Rutherford, M. J., Carzaniga, A., and Wolf, A. L. 2005. "Automating experimentation on distributed testbeds". In Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering (Long Beach, CA, USA, November 07 - 11, 2005). ASE '05. ACM Press, New York, NY, 164-173.

[100] Albrecht, J., Tuttle, C., Snoeren, A. C., and Vahdat, A. 2006. "PlanetLab application management using plush". SIGOPS Oper. Syst. Rev. 40, 1, Jan. 2006, pp. 33-40.

[101] Andersen D., Feamster N., "Challenges and Opportunities in Internet Data Mining". Parallel Data Laboratory, Carnegie Mellon University, Research Report CMU-PDL-06-102, Jan. 2006.

[102] Ahumada, L.; Feick, R.; Valenzuela, R.A., "Characterization of temporal fading in urban fixed wireless links," Communications Letters, IEEE , vol.10, no.4, pp. 242-244, Apr 2006

[103] J. Yeo, S. Banerjee and A. Agrawala, "Measuring traffic on the wireless medium: Experience and pitfalls", Technical report, CS-TR 4421, University of Maryland, College Park, December 2002. URL: http://www.cs.umd.edu/Library/TRs/CS-TR-4421/CS-TR-4421.pdf [last visited:November 2008].

[104] J. Yeo, M. Youssef and A. Agrawala, "Characterizing the IEEE 802.11 Traffic: The Wireless Side," CS-TR-4570, March 2004. URL:

http://www.cs.umd.edu/Library/TRs/CS-TR-4570/CS-TR-4570.pdf [last visited:November 2008].

[105] Y.C Cheng J. Bellardo, P. Benko, A.C. Snoeren, G.M. Voelker and S. Savage, "Jigsaw: Solving the Puzzle of Entreprise 802.11 Analysis," in Proc. of ACM SIGCOMM, Pisa, Italy, September 11-15 2006.

[106] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Analysing the MAC-level Behavior of Wireless Networks in the Wild," Proc. of ACM SIGCOMM, Pisa, Italy, September 11-15 2006.

[107] Athdebug and 80211debug documentation, Madwifi v0.9.4, URL: http://madwifi.org/wiki/DevDocs/AthDebug [last visited:August 2008]

[108] athstats.c, Madwifi v0.9.4 source code, URL: http://www.madwifi.org [last visited:August 2008]

[109] T. Henderson and D. Kotz, "Problems with the Dartmouth wireless SNMP data collection," Dartmouth Computer Science Technical Report TR2003-480, December 2003.

[110] Wireless Tools for Linux URL: http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/To visited:November 2008]

[111] Wrapi URL: http://sysnet.ucsd.edu/pawn/wrapi/ [last visited:November 2008]

[112] Wavemon tool URL: http://www.janmorgenstern.de/projects-software.html [last visited:November 2008]

[113] Airtraf tool URL: http://airtraf.sourceforge.net [last visited:August 2008]

[114] C. Hoene, B. Rathke and A. Wolisz: "EasySnuffle: A tool to measure the performance of multimedia flows over IEEE 802.11b", Technical University of Berlin TKN - Berlin - Germany, March 10, 2002, URL: http://www.tkn.tu-berlin.de/research/easysnuffle/EasySnuffle.pdf [last visited:November 2008].

[115] Jigsaw tool URL: http://sysnet.ucsd.edu/wireless/ [last visited:November 2008]

[116] Kismet tool URL: http://www.kismetwireless.net [last visited:November 2008]

[117] Mognet tool URL: http://www.monolith81.de/mognet.html [last visited:November 2008]

[118] Wifiscanner tool URL: http://wifiscanner.sourceforge.net [last visited:November 2008]

[119] Wireshark tool URL: http://www.wireshark.org [last visited:November 2008]

[120] WisMon tool URL: http://planete.inria.fr/software/WisMon/ [last visited:November 2008]

[121] Wit tool URL: http://www.cs.washington.edu/research/networking/wireless/ index.html[last visited:November 2008]

[122] Crawdad repository URL: http://crawdad.cs.dartmouth.edu/tools.php.

[123] F. Vacirca and A. Baiocchi, "Characterization of Service Times Burstiness of IEEE 802.11 DCF," Wired/Wireless Internet Communications, Springer, 2007, pp. 223-234.

[124] J. Bardwell, "Converting Signal Strength Percentage to dBm Values," November 2002, WildPackets Inc., URL: http://www.wildpackets.com/elements/whitepapers/Converting_Signal_Strength.pdf [last visited:November 2008].

[125] J. Barker, "You Believe You Understand What You Think I Said: The Truth About 802.11 Signal And Noise Metrics", Document D100201, 2004 - Connect802 Corporation, URL: http://www.connect802.com/download/techpubs/2004/you_believe_D100201.pdf [last visited:November 2008].

[126] D. Dujovne and T. Turletti, "Multicast in 802.11 WLANs: an experimental study," Proc. of the 9th ACM international Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM), Torremolinos, Spain, October 2-6, 2006. pp. 130-138.

[127] J. Lacan and T. Perennou, "Evaluation of Error Control Mechanisms for 802.11b Multicast Transmissions," International Workshop on Wireless Network Measurement (WinMee), Boston, MA, USA, April 3, 2006.

[128] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla and M. Y. Sanadidi, "CapProbe: A Simple and Accurate Capacity Estimation Technique," Proc. of ACM SIGCOMM'04, Portland, OR, USA, September 2004.

[129] A. Johnsson, B. Melander, and M. Bjorkman, "Bandwidth Measurement in Wireless Networks," Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), Porquerolles, France, June 2005.

[130] S. Khurana, A. Kahol, and A.P. Jayasumana, "Effect of hidden terminals on the performance of IEEE 802.11 MAC protocol", Proc. of 23rd Conference on Local Computer Networks (LCN), pp. 12-20, 1998.

[131] K. Lakshminarayanan, V. N. Padmanabhan, and J. Padhye, "Bandwidth Estimation in Broadband Access Networks," Proc. of ACM IMC'04, Taormina, Sicily, Italy, October 2004.

[132] A. Goldsmith, "Wireless Communications", Cambridge University Press, ISBN 0-521-83716-2, 2005.

[133] Radiotap URL: http://www.radiotap.org [last visited:November 2008]

[134] M. Raya, J.P. Hubaux and I. Aad, "DOMINO: a system to detect greedy behavior in IEEE 802.11 hotspots", Proceedings of the 2nd international conference on Mobile systems (MobiSys), Boston, MA, June 2004.

[135] R. Presuhn et al., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)," IETF RFC 3416, December 2002.

[136] Easysnuffle tool URL: http://www.tkn.tu-berlin.de/research/easysnuffle/ [last visited:November 2008]

[137] AVS Capture Frame Format, URL: http://www.locustworld.com/tracker/getfile/prism2drivers/doc/capturefrm.txt [last visited:November 2008]

[138] "IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999), June 12 2007.

[139] Round Robin Database tool, URL: http://oss.oetiker.ch/rrdtool/ [last visited:April 2009]

Abstract

Since the inception of the 802.11 standard in 1999, WLANs, which used to be exceptional, became a massive phenomenon together with the evolution of portable devices. At the same pace, research on wireless networks, where both simulation and experimentation are used to validate protocols, has evolved rapidly. Nevertheless, the models used in this area were adapted from the wired paradigm, which has led to a significant gap between the simulated and the experimental results. Therefore, to validate wireless protocols or algorithms, wireless experimentation becomes an important resource. This thesis explores the improvements to experimentation on WLANs, from a methodological point of view. In this thesis, we first create a new model for data abstraction to represent network events and aggregated data logs; second, we design a methodology to manage data in order to support a database model; and third we replace custom made processing scripts with data-oriented filter modules. These three key points converge into a wireless experimentation methodology to specify the experimental conditions and improve reproductibility. Furthermore, we present Wextool, an implementation of a Wireless Experimentation Tool which applies the methodology and finally we show the improvements through a wireless multicast experimentation use case and a performance evaluation of the process.

Résumé

Depuis la création de la norme 802.11, en 1999, les réseaux locaux sans fil, qui étaient rares á l'époque, sont devenus un phénomène incontournable avec le développement spectaculaire des appareils mobiles. Au même rythme, la recherche sur les réseaux sans-fil a rapidement évolué suivant des modèles adaptés du paradigme des réseaux filaires, qui a conduit un écart significatif entre la simulation et les résultats expérimentaux. Or pour valider les protocoles ou les algorithmes sans fil, l'expérimentation est une approche incontournable. Cette thèse étudie les améliorations portant sur l'expérimentation sur les réseaux sans fil, d'un point de vue méthodologique. Dans cette thèse, nous avons d'abord proposé un nouveau modèle d'abstraction de données pour représenter les évènements de réseau et de l'agrégation de données statistiques. En deuxiéme lieu, nous avons élaboré une méthodologie pour la gestion des données utilisant un modèle de base de données. Enfin nous avons automatisé le déroulement de l'expérimentation avec des scripts sur mesure avec des modules de filtrage et traitement. Ces trois principaux éléments constituent une méthodologie d'expérimentation qui pointe sur les conditions des expériences et qui permet d'en améliorer la reproductibilité. Après, nous présentons Wextool, un outil d'expérimentation sans fil qui appliquant la méthodologie proposée et nous montrons les améliorations obtenues en terme de temps d'expérimentation à travers une étude portant sur un scenario de diffusion multipoint dans un réseau sans fil.