

Bit-error analysis in WiFi networks based on real measurements

Gábor Fehér

Budapest University of Technology and Economics,
Department of Telecommunications and Informatics,
Magyar tudósok krt. 2, 1117 Budapest, Hungary
feher@tmit.bme.hu

Abstract. The IEEE 802.11 standard is coming from 1999. Since that time lots of research paper were born analyzing WiFi networks. However, until the recent years, WiFi devices and drivers were on closed source, so measurements could rely only on those features that the vendors offered for them. For such reason there could be hardly any research focusing on the bit level internals of WiFi transmissions. Today we already have better tools to access the WiFi devices. This paper presents measurements in real WiFi scenarios and shows what happens with the message bits on their flight. The paper also highlights that the implementation of WiFi devices are very different and using a single parameter set to model them is inappropriate and might be misleading. *abstract* environment.

Key words: wireless, WiFi, bit-errors, measurement

1 Introduction

The IEEE 802.11, WiFi transmission is a frequent research topic. WiFi networks are everywhere and researchers want to tune WiFi networks to its best performance. The performance of this wireless network was investigated and published in many papers [1, 5, 4]. However, most of the investigations and measurements focuses on the frame transmission as an atomic event, and they have drops and successful transmissions, but never have errors within the transmitted frame. The 802.11 standard defines to use CRC checksum to protect the integrity of the frames. Whenever a frame gets corrupted during the transmission, the WiFi device at the receiver will check the checksum and drop the frame if it contains errors. People who are not hacking WiFi drivers are unable to control the check, so they are forced to use correct frames only. The drop is obvious, when the sender side sent the frame, but the receiver side did not get it.

There are only a low number of publications that really focuses the internals of the WiFi transmission. Giuseppe Bianchi and his group has a modified Atheros driver, which is able to show more than the everyday user or researcher can see from a wireless frame transfer. In their publications [2, 6, 10] they did measurements using the Atheros card and a modified open source driver (MadWifi [8]). In their recent publication they concluded, that taking measurements without

understanding the implementation details may lead to biased experimental trials and/or to erroneous interpretation of experimental results.

Hacking Atheros chipset based 802.11b/g cards and their drives was very popular among researchers and it is still popular even today. The reason is that there exists an open source driver, called MadWiFi, which allows modifications in the MAC (Media Access Control) layer. The Atheros card still has a closed source firmware, but it is thin and most of the MAC level frame processing is done in the driver. Using the MadWifi driver, it is possible to capture all the frames that the receiver card gets, even those frames that are corrupted during the transmission.

1.1 Novel measurements

This publication steps over the Atheros cards and the MadWifi driver. In the following sections we introduce how Linux systems were improved in the view of wireless drivers and their capabilities. We present the current technologies that are available to capture and also to transmit wireless data as the user, developer or researcher wants it. Using the capture and transmit functions we demonstrate that the implementation of wireless devices are so different that it is impossible to describe them with a model and a single parameter set. After the presentation of our initial measurements we show more measurement results analyzing bit errors during the wireless transmissions.

2 Linux support for WiFi capture

WiFi cards are very different in general, but there are a few things in their implementation that is common. Excluding the System-on-Chip design, they are all built around a chipset coming from a specific vendor. They have a code, called firmware to drive the chip inside and they have a driver software running on the host machine to communicate with the card. Regarding the firmware and the driver, at the beginning of the WiFi device productions, somewhere in the late 90's, vendors put all the card control software into firmware. This is called FullMAC, where almost anything related to the WiFi transmission managed on the card itself. The driver was thin, its function was to feed the card with outgoing packets and receive the incoming ones. Later, the implementation design changed completely. Nowadays vendors produce so called SoftMAC cards and drivers, where the firmware is thin and the driver part is responsible to do the MAC functions. Indeed, only the physical layer related codes (e.g. modulation) and some regulatory codes remained on the card, all the MAC function went to the host machine. This transition opened a path to make modification in the MAC functions.

2.1 mac80211 in Linux

WiFi device vendors usually have trade secrets, so at the beginning their drivers were not open for modifications. In the middle of 2003 an open source driver for

Atheros based cards, the Multiband Atheros Driver for WiFi (MADWIFI) came out. This driver still exists and maintained. Due to its open source, it is already possible to put any modifications to the driver. Later other open source drivers were developed for various chipsets. Some vendors, just like Ralink also helped the open source Linux system by offering official open source drivers. In 2007 a new wireless development paradigm appeared in Linux. Developers started to build a common platform for the various wireless drivers. This is the mac80211 development, where conceptually the MAC layer is the part of the Linux kernel. Firmwares and drivers are thin now, the MAC functionality is positioned to the common kernel. This movement has the great advantage that developers can place their codes into the common MAC code and it will run on all cards that fully support the new architecture. Various models from Atheros, Broadcom, Orinoco, Ralink, Realtek, ZyDAS cards are already supported. As time goes on more and more cards become available with mac80211 support. All the devices that run the recent Linux kernel already support mac80211 by default.

An exciting feature of the mac80211 code is that the virtual devices can be created easily. Moreover, there is an operation mode, called MONITOR mode, where the card is set to capture all the frames that it can get. The MONITOR mode can be instructed to capture not just the correct frames, but also the damaged ones. There are two kinds of damages a frame might have suffer. First, when the card is able to detect the PLCP (Physical Layer Convergence Procedure) preamble and is able to synchronize to it, but there is error in the payload. This is the CRC error, as the 32 bit Cyclic Redundancy Check (CRC) value will signal the problem. The second type of error is the PLCP error, where the card is unable to synchronize to the preamble.

2.2 The RADIOTAP header

Another exciting feature of the Linux wireless code that it supports various extra information regarding the receiving procedure of the actual frame. The information is collected into a field, called radiotap [9] header, when the wireless device is in monitor mode. In this case the radiotap header is inserted to the beginning of the frame. Through the radiotap header the following most important characteristics can be obtained for a received frame: the antenna number, on which the frame was received, antenna noise power, antenna signal power, channel, rate, CRC error, PLCP error.

As a bonus feature, radiotap header is not only meant for receiving frames, but frames can be transmitted with it as well. The radiotap header should be inserted before the frame and then the frame should be sent to the interface, which is in monitor mode. Adding the radiotap header, it is possible to set up for example the transmission rate and the transmission power of the frame.

2.3 Linux on the Access Point

The previously mentioned mac80211 code, monitor mode and radiotap header are available for all machines that run Linux. Linux is not limited to desktops

only, but we can find even Access Points that run the Linux operating system. In fact, it is a cheap choice for Access Point vendors, since they have a powerful operating system without costly licenses (Actually Linux has the GNU General Public License, but vendors tend to forget it). There is developer community that creates OpenWRT [7] a Linux based firmware for various Access Points. Currently the development supports 80 different Access Points coming from 37 different vendors. In addition plus 80 more Access Points are marked as work in progress. Since OpenWRT is Linux based, roughly all Linux based Access Point with not less than 4 MB ROM can run it. The kernel and the drivers are the same as for desktop Linux, except the CPU architecture is usually different. Naturally, OpenWRT is based on a recent Linux kernel, so radiotap functions are available during frame captures and transmissions.

2.4 Wireless card drivers in Windows 7

In the Windows operating system it was already planned in 2002 to introduce virtual WiFi adapters and share the resource of the single WiFi device [3]. Unfortunately, at that time there was no driver support from the vendors. Starting with Windows 7, Windows already implement a virtual WiFi interface, however its capability is limited. Hopefully in the future we can see more advancement on the Windows line as well.

3 WiFi measurements

We made various measurements using modified software on the Access Point and the WiFi clients. For all the measurements we used a relatively cheap ASUS WL-500gP Linux based Access Point. Due to some implementation problems, we replaced the original Broadcom WiFi card with an Atheros card. This modification was necessary, since at the time of the measurements, Broadcom had not released a mac80211 architecture based Linux driver yet, while Atheros did. Today we already have mac80211 support on Broadcom devices as well.

3.1 Measuring various clients

First of all we measured three different client side WiFi devices in order to get an initial picture of the radio chipset capabilities. All the tested devices were off the shelf USB stick. This test was a simple one. We had the Access Point to broadcast test frames on a 48 Mbps rate to a certain multicast address. The three different WiFi clients were switched to monitor mode and recorded all the transmissions that their radio were capable to receive. The three WiFi devices were placed about 7m away from the Access Point. The antenna of the Access Point was detached to have worse signals. There were no ACK frames, since the measurement frames were multicast frames and it is not acknowledged by the receivers. We repeated the tests three times and measured how many

valid frames were captured by each device. The repeated test happened roughly the same time, the background traffic and noise of the radio channel can be considered static during the tests. The measurement results are displayed in Fig. 1.

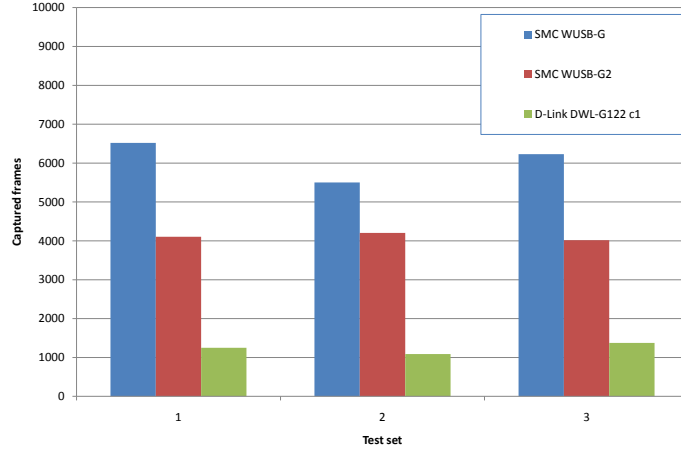


Fig. 1. Receiving WiFi transmission on different WiFi cards

Based on the measurement results we can observe that there is a huge difference among the capabilities of the tested devices. The best device in our tests was the old version of the SMC WUSB-G stick, while the worst performer was the D-Link DWL-G122 device capturing the least frames. The SMC devices had ZyDAS chipset, while the D-Link had Ralink. All of the devices gave a steady performance, as they reacted the same way in the same situation. As a first conclusion we can state that due to hardware or software reasons WiFi clients perform differently under the same circumstances. Thus we cannot make a simple model of a generalized WiFi client, where only the radio channel parameters are presented. In contrast, we can measure the actual performance of a given WiFi client and we can assume that this performance does not fluctuate while the conditions of the radio channel are the same.

In the further measurements we used the SMC WUSB-G WiFi client, as this device had the best performance in the previous tests.

3.2 Measuring different channel conditions

In this measurement configuration we had an indoor scenario presented on Fig. 2. There was one Access Point in one of the room, signaled with AP on the figure. We have 6 indoor positions for the wireless client. First the $4m$ scenario, where the client was in the same room as the Access Point, placed 4 meter away

from it. In the *6m* scenario there was already a thin wall between the Access Point and the client. The distance of the Access Point and the client is estimated for 6 meter. The *1 room* scenario has one room (i.e. 2 walls) between the Access Point and the client. The *2 rooms*, *3 rooms* and *4 rooms* has 2,3 and 4 rooms respectively in between the Access Point and the client. The Access Point was the modified ASUS WL-500gP device and a client was a laptop running Linux equipped with the SMC WUSB-G card.

The measurements were performed in the University’s building during the night. We tried to choose a silent period where other radio signals do not disturb the measurements. Also, we selected a WiFi channel, where the channel and its first and second neighbors were not allocated by other Access Points.

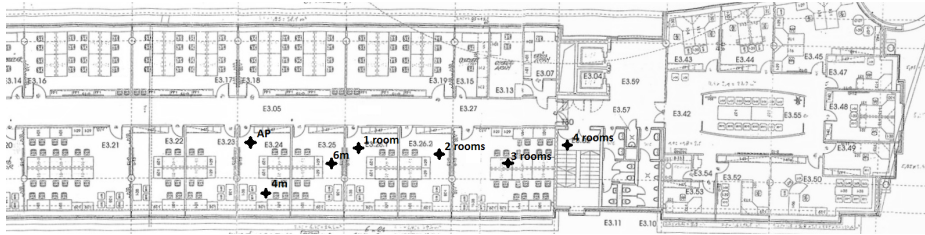


Fig. 2. Indoor scenarios

In each measurement scenario we took a 7 hour long measurement. The software modified Access Point sent out 1000 byte long, specially patterned measurement frames periodically in each 1/10th seconds. The transmission speed was redefined after each transmission and the values cycled through 1, 2, 5.5, 11, 6, 9, 12, 18, 24, 36, 48 and 54 Mbps. The first 4 transmission rates are IEEE 802.11b rates, while the latter 8 are for 802.11g. Also, in the latter cases, the modulation was OFDM instead of DSSS. The destination of the measurement flow was a multicast address, so the Access Point waited for no acknowledgments. The client was switched to monitor mode and recorded all the correct and damaged frames it could. The damaged frames suffered bit modifications during the transmissions. As we constructed the measurement flow in a special way using a recognizable bit pattern, at the receiver side we were able to identify the place of the erroneous bits.

In the case of the first indoor scenario, where the distance from the Access Point to the WiFi client was only 4 meters, almost every frame were correctly transmitted even on the highest transmission rate. In the case of the 4 room scenario, there were hardly any frames received even on the lowest rate.

3.3 The good, the bad and the dropped

We present some measurement results from the middle ranges that demonstrate the receiver’s performance when receiving the same transmission on different transmission rates. The results are presented along the transmission speed and

we put the results into 5 groups. The first group is for the good frames, the frame was correctly received here. The second group is for the lightly damaged frames, we have byte changes here up to 1 percent of the whole frame (1-10 bytes). The third group is a moderate damage between 1 and 10 percent change in the frames (10-100 bytes). The fourth group indicates a severe damage, as more than 10 percent of the frames (more than 100 bytes) were changed during the transmission. The last group represents frame drops, here the receiver were unable to catch the frame in the air. We know about the drops since the measurement frames are sent out periodically.

During the 7 hour measurements there were certain periods, where the channel seemed to be better and other times worse. We selected an 2 hour interval where we had nearly steady performance and calculated the average values for the different transmission speeds. Since the measurement frames were sent out alternating the transmission speed one by one, therefore the same 2 hour period is used for all the different speeds.

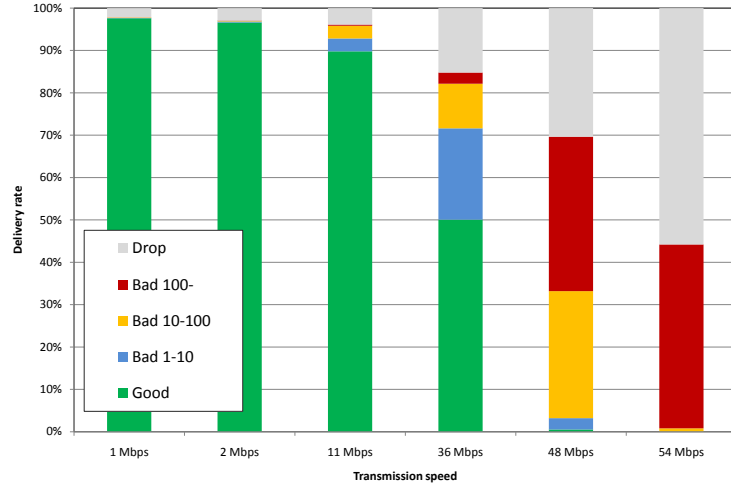


Fig. 3. Transmission results in the case of the 1 room scenario

Fig. 3 displays the 1 room scenario, where there was one room between the Access Point and the WiFi client. The 97.59 percent of the 1 Mbps measurement flow was received by the client correctly, 2.1 percent of the flow contained erroneous frames and 2.2 percent of the flow was lost. In a higher transmission speed, at 36 Mbps, the receiver was able to correctly capture only 50.07 percent of the measurement flow. There is a significant amount, 21.55 percent of the flow, where frames contain a small number of errors, up to 1 percent of the total length. 13.17 percent of the measurement flow suffered more than a light damage, while 15.2 percent of the flow did not reach the client at all. In the highest transmission rate, which was 54 Mbps, there is hardly any valid frame.

Just a small fraction, 0.81 percent of the flow was received with less than 10 percent of errors. 43.39 percent of the frames was received with more than 10 percent of errors and 55.79 percent of the measurement flow were lost during the transmission. This measurement result shows that there exists a situation where we can have a close to perfect transmission even on a moderate transmission speed, ie. 89.81 percent of successful delivery rate at 11 Mbps, while on higher speeds we already have a significant amount of error.

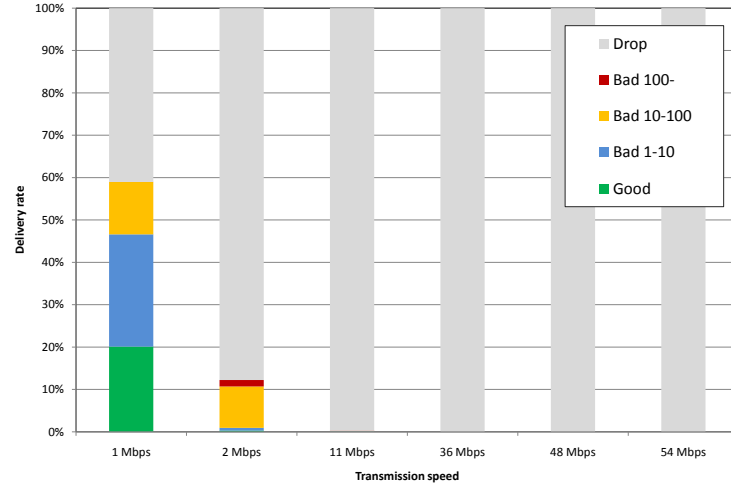


Fig. 4. Transmission results in the case of the 3 room scenario

In Fig. 4 we present the measurement results of the 3 rooms scenario. Here there were 3 rooms between the Access Point and the WiFi client. As the figure shows, the 4 walls and the distance had a serious impact on the transmission. At the 36, 48 and 54 Mbps transmission rates there were no frames at all received by the WiFi client. In contrast, 20.12 percent of the measurement traffic sent out with the 1 Mbps transmission speed was correctly received by the client. In this case, there is the 26.47 percent of the measurement frames that were received with less than 1 percent error in the frames. This is also a significant amount. Plus there is 12.4 percent of the measurement flow that was received with more than 1 percent errors. Here 41 percent of the measurement flow was not captured. This statistics becomes a lot worse in the case of the 2 Mbps measurement frames. The loss is already 87.79 percent, and only 0.03 percent of the measurement flow was received correctly. The weights of the damaged frames are 0.64, 9.77 and 1.51 percent respectively. This measurement also highlights the differences among the performances at various transmission speeds. Moreover, we can observe, that using the base rate, it is still possible to send frames to places, where the radio channel is already heavily distorted.

3.4 The number of errors and the signal strength

In the following measurements we analyzed the relation between the number of errors within the frame and the signal strength that was measured by the capturing WiFi device. The number of errors are expressed in bytes, while the official measurement unit of the signal strength is dB. This latter metric could be a little bit misleading, since it is measured to a fixed reference that can vary from driver to driver. It is impossible to compare the signal strength values among different cards, however it is a good indication when there is just a single card in use.

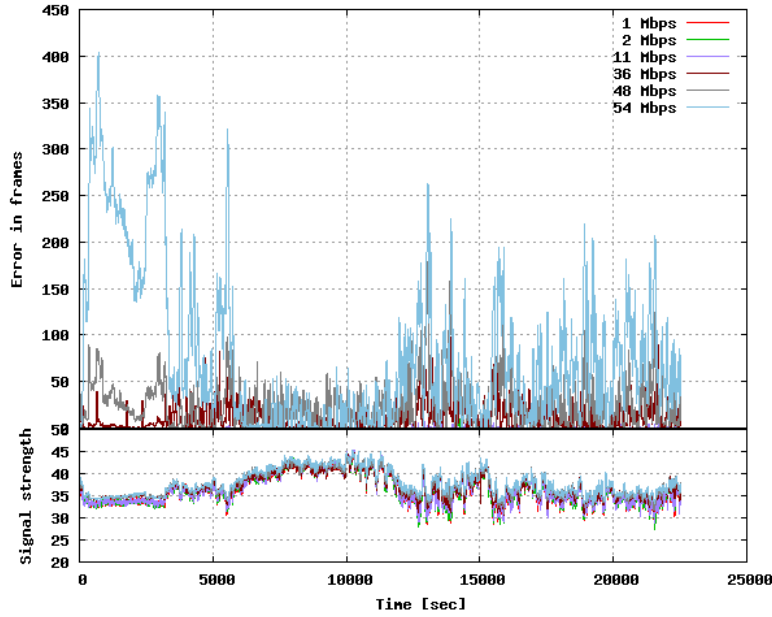


Fig. 5. Errors and Signal strength in the 6 meter scenario

Fig. 5 presents the measurement results in the 6 meter scenario. In this case there was 6 meter between the Access Point and the WiFi client, and there was also a wall between them. The figure shows the full length of the measurement, 25000 seconds that is around 7 hours. The different curves show different transmission speeds. Although the per frame number of errors in the case of the 1 to 11 Mbps transmission speeds are very low and therefore indistinguishable on the figure, we can observe that in the higher rates the number of damaged bytes are already significant in the transmitted frames. Moreover, we show that despite of our efforts to create an environment where the channel condition is stable, there are sections in the measurement where the receiving behavior differs a lot. During the first 5500 seconds the number of errors are really high for the 54

Mbps transmission. In this section the signal strength is around 35 dB. In the second section, which is between 5500 and 12000 seconds, the signal strength is better, it goes up to 45 dB. The transmission has less errors, it is always under 70 bytes for all the measurement flows. In the third section, which is after the first 12000 seconds, both the signal strength and the number of errors in the frames are fluctuating. Interestingly, the signal strength is lower than it is in the first section, however the performance in the view of the number of errors is better. This measurement underlines that we cannot derive straight relationship between the signal strength metric and the amount of damages within the frames. On the figure we can see that it is only the 54 Mbps measurement signal that has a three different sections, the remaining 5 measurement flows show balanced performance during the whole measurement. The 54 Mbps flow was not a distinguished one, since the measurement signal was cycling through the transmission speed settings frame by frame, still we got this results. Our assumption is that there was a background wireless traffic and that created the three different sections.

On Fig. 5 the signal strength curves run together. This means that the signal strength is independent of the transmission speed. Moreover, as the measurement frames followed each other in a 0.1 second distance, we can show that the signal strength changes slowly in time assuming steady channel conditions.

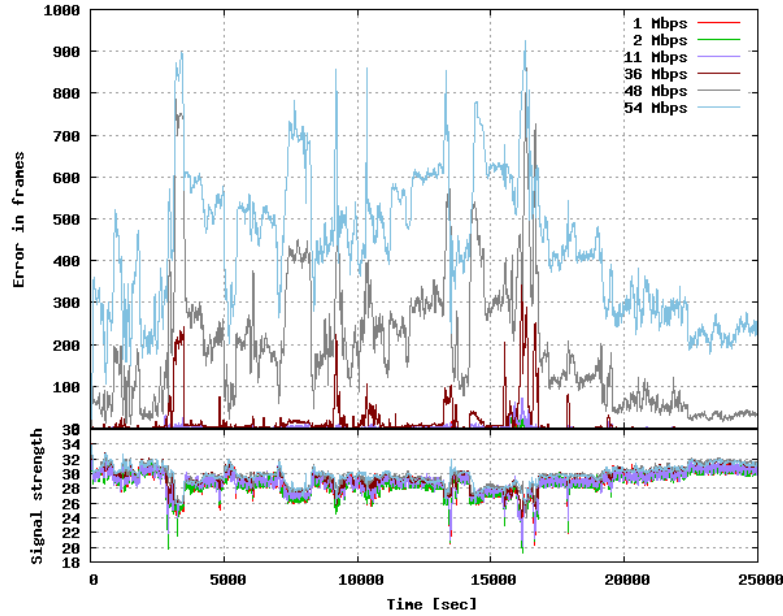


Fig. 6. Errors and Signal strength in the 1 room scenario

Fig. 6 presents the same error and signal strength metrics as it is on the previous figure. Here we displayed the results of the 1 room scenario, where there is 2 walls between the Access Point and the WiFi client. On the figure the signal strength curves stay together, showing the independence of the signal strength and the transmission speed. The figure perfectly displays that there is a connection between the number of errors and the transmission speeds. The number of error curves in the case of 48 and 54 Mbps are fluctuating similarly. Moreover, when the number of errors within a transmitted 36 Mbps measurement frame is high, the curve also follows the shape of the higher rate curves.

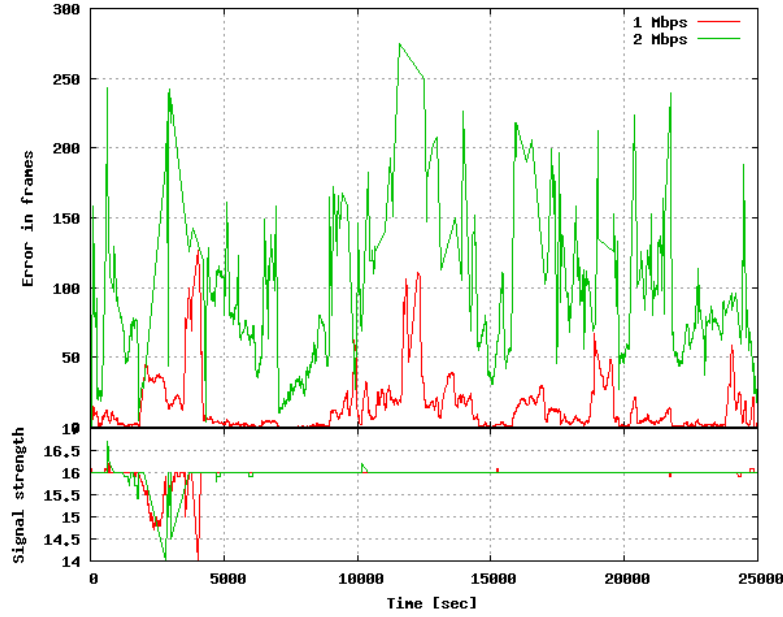


Fig. 7. Errors and Signal strength in the 3 rooms scenario

Finally on Fig. 7 the results of the 3 rooms scenario measurements are displayed. There are only two measurement flows on the figure, since on the higher rates we had hardly any captured frames. The per frame number of errors are high at both flows and we can identify again the connection between the number of errors and the transmission speed. The received signal strength is around 16 dB during the measurement, which value is considered very low.

4 Conclusion

In this paper we presented WiFi measurement results. We utilized the Linux mac80211 wireless driver architecture, and set the card to monitor mode. Thus

we were able to capture all frames in the air regardless they have correct CRC or not. With the help of the radiotap headers we knew the signal strength for the received frames. In our Access Points we ran Linux as well, namely the OpenWRT distribution. We sent the frames using the radiotap headers and set the transmission speed. This measurement system is available to everyone, since all the required components are in the common Linux kernel.

We made very long measurements, sending 250000 specially constructed frames in each scenario at various transmission rates. During the measurements we analyzed the bit errors that transmitted damaged the frames. Based on the measurement results we can state that there is a connection between the per frame number of errors and the transmission speed. Despite of the similar signal strength values, flows with different transmission speed have different number of errors in their frames.

We also pointed out that wireless devices are so different that making conclusions based on the observation of a specific card and driver pair is inappropriate and might be misleading.

5 Acknowledgments

The research leading to these results has received funding from the European Union's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n INFSO-ICT-214625.

References

1. G. Anastasi. Ieee 802.11 ad hoc networks: performance measurements. In *in: Proceedings of the Workshop on Mobile and Wireless Networks (MWN 2003) in conjunction with ICDCS 2003*, pages 758–763, 2003.
2. Giuseppe Bianchi, Fabrizio Formisano, and Domenico Giustiniano. 802.11b/g link level measurements for an outdoor wireless campus network. In *WOWMOM*, pages 525–530. IEEE Computer Society, 2006.
3. Ranveer Chandra, Paramvir Bahl, and Pradeep Bahl. MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In Bo Li, Marwan Krunz, and Prasant Mohapatra, editors, *INFOCOM 2004, 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 882–893, Piscataway, NJ, USA, March 2004. IEEE Computer Society.
4. Yu chung Cheng, John Bellardo, Pter Benk, Alex C. Snoeren, Geoffrey M. Voelker, and Stefan Savage. Jigsaw: Solving the puzzle of enterprise 802.11 analysis. In *In Proceedings of the ACM SIGCOMM Conference*, 2006.
5. Mirko Franceschinis, Marco Mellia, Michela Meo, Maurizio Munaf, Istituto Superiore, Mario Boella, and Torino Italy. Measuring tcp over wifi: A real case. In *In 1st workshop on Wireless Network Measurements (Winmee), Riva Del Garda*, 2005.
6. Domenico Giustiniano, Giuseppe Bianchi, Luca Scalia, and Ilenia Tinnirello. An explanation for unexpected 802.11 outdoor link-level measurement results. In *INFOCOM*, pages 2432–2440. IEEE, 2008.

7. David Heldenbrand and Christopher Carey. The linux router: an inexpensive alternative to commercial routers in the lab. *J. Comput. Small Coll.*, 23(1):127–133, 2007.
8. madwifi. madwifi homepage. <http://www.madwifi.org>.
9. radiotap. radiotap homepage. <http://www.radiotap.org>.
10. Ilenia Tinnirello, Domenico Giustiniano, Luca Scalia, and Giuseppe Bianchi. On the side-effects of proprietary solutions for fading and interference mitigation in ieee 802.11b/g outdoor links. *Computer Network*, 53(2):141–152, 2009.