Non-invasive Node Detection in IEEE 802.11 Wireless Networks

Guido Piero Zanetti
Department of Pure and Applied Mathematics
University of Padova
Padova, Italy
gzanetti@studenti.math.unipd.it

Claudio Enrico Palazzi
Department of Pure and Applied Mathematics
University of Padova
Padova, Italy
cpalazzi@math.unipd.it

Abstract— **IEEE 802.11 wireless sniffers are useful tools for monitoring network traffic and for the acquisition of information at the MAC level and above. The standard approach for implementing a wireless sniffer is to setup a device in the so-called *Promiscuous* mode. In this mode, devices are able to acquire all traffic that passes through them, independently from its destination. On a wireless network, this means that it is possible to monitor all the activities inside the device's cover range. Unfortunately, the Promiscuous mode limits the use of the device by prohibiting the association with wireless networks, or in general, sending data. This limitation restricts uses of sniffers on specific scenarios. This paper shows how to remove such limitation by presenting our *Laurin Project*. Laurin is a wireless sniffer that provides sniffer's capabilities without limiting the device's use. This solution opens the doors to the development of new desktop-oriented applications that take advantage of network monitoring capabilities.**

Keywords: IEEE 802.11, sniffer, mac80211, Promiscuous mode, RFMON

I. INTRODUCTION

Traditionally network monitoring distinguishes two types of measurement: active and passive. The former is based on injecting traffic flow in a network, whereas the latter transparently collects and analyze observed traffic. Network monitoring offers very interesting functionalities such as: i) tracking down issues that are causing poor performance or security leaks, ii) detecting the most used protocols or the most active stations, iii) detecting intrusions or the presence of regular nodes in the network. Besides supporting network administrators, network monitoring can be exploited to generate new appealing applications for common network users. For instance, it could be used to detect which users are inside the coverage of a wireless device, triangulate their position, and generate new proximity- or location-based services.

Several tools and protocols (e.g. SNMP) exist that are able to monitor network activities [1]-[3]. Sniffers are common examples of how passive monitoring can be implemented. WLAN sniffer's capabilities are essentially the same as their LAN counterparts but challenged by the specific additional constraints that wireless networks impose [4]. Sniffers work collecting data from the network. To collect suitable network traffic a device must be able to read all the data that pass through it. The *Promiscuous* mode of a device permits to satisfy this requirement; unfortunately, it severely limits its use. In fact, a device in Promiscuous mode cannot send any data. In a WLAN scenario, this implies that no network association can be established; this is a huge limitation for users. Tools such Wireshark [1] or Kismet [2] are interesting network activity monitors. Wireshark is able to acquire all the data that pass through the device; it detects which protocols are used and shows all communication details. Kismet is very similar, yet it has different purposes: it embodies a wireless network detector, sniffer, and intrusion detection system which is able to acquire information about wireless networks topology through the raw monitoring (RFMON). Wireless networks are growing with the diffusion of mobile computers such as notebooks, netbooks but also mobile phones, smartphones, and PDAs [5]-[8]. These kinds of wireless node are often equipped with only one wireless interface. Users do not want to lose connectivity even for few second by setting their wireless interface in Promiscuous mode. For this reason, Wireshark and Kismet, and in general all sniffers, are confined on specific scenarios, where users are conscious of how these programs work and the connectivity limitations they impose.

Protocols such as SNMP [3] may seem valid alternatives to the Promiscuous mode as they monitor network activities without limiting the device's capabilities. SNMP is a UDP-based network protocol; it is used mostly to monitor network-attached devices for conditions that warrant administrative attention. Unfortunately, like other sniffers, it imposes some constraints: it requires specific configurations and must be supported by devices. Therefore, the SNMP protocol is not a widely accepted solution for network monitoring.

In summary, there are several tools to monitor network activities, but each of them is challenged with specific constraints. We aim at finding a solution for implementing applications that can take advantage of network monitoring in an easy and suitable way: no complicated setup, special hardware or user limitations. Sniffers seem to be a valid starting point for our goal. This paper describes how it is possible to develop WLAN sniffers not constrained by

Promiscuous mode's limitation of losing network connectivity, while maintaining the capability of monitoring the network activity. This solution is implemented in our WLAN sniffer project, named *Laurin*. Through the explanation of how Laurin works, we show that it is possible to develop desktop-oriented applications that take advantage of network monitoring without imposing constraints or requiring special hardware. The solution found is suitable for the GNU/Linux operating system for it adopts the new *mac80211* wireless stack [9] which is a key component for our solution. Through the use of this new stack, Laurin is able to collect network traffic without influencing the overall wireless performance and most importantly without losing the ability to send data.

The rest of this paper is organized as follow. Section II explains used terminology and concepts. Section III presents the Laurin project and shows our main idea. Section IV presents some benchmark results that test the performance of our solution and prove its efficacy. Section V provides some ideas for future work. Indeed, with the possibility of taking advantage of network monitoring, several applications that are still confined to specific scenarios can be now widely available.

II. BACKGROUND

In its common approach, network sniffing requires to set devices in Promiscuous mode. The Promiscuous mode is a configuration of a network card that makes the card elaborate all traffic it receives rather than just frames addressed to it.

When a network card receives a frame, it normally drops it unless the frame is addressed to that card. In Promiscuous mode, however, the card allows all frames through, thus allowing the computer to read frame intended for other machines or network devices. With a network card running in Promiscuous mode we are able to acquire information at the MAC level and above. In GNU/Linux operating system, a wireless device is normally handled as an IEEE 802.3 device [10]; hence, by setting it in Promiscuous mode, we are able only to acquire IEEE 802.3 frames. The wireless card's driver transforms an IEEE 802.11 frame into an IEEE 802.3 frame and vice versa. To overcome this problem, we need to consider the *RFMON* (Radio Frequency MONitor) mode [4]. This mode is very similar to the Promiscuous mode, but it has the advantage to read IEEE 802.11 frames and it is able to capture them without having to associate with an access point (AP) or ad-hoc networks first.

Obviously, RFMON mode applies only to wireless devices, while Promiscuous mode can be used on both wireless and wired devices. RFMON is one of the six modes in which an IEEE 802.11 wireless card can operate. The other five are: *Master*, *Managed*, *Ad-Hoc*, *Mesh*, *Repeater*. RFMON is often used for malicious purposes such as WEP cracking, but also to help network administrators. In fact, it is not uncommon to use it to collect data about the networks' topology (i.e. busy spectrums and channels) with the intent to help reducing interference with other wi-fi devices. Like Promiscuous mode, a wireless card in RFMON mode is unable to transmit, and it is restricted to a single channel too, even if this is dependent on the wireless card's driver and its firmware.

The data collected in RFMON consists of the IEEE 802.11 frames and corresponding payloads. Some wireless card drivers also add some additional information in a frame header. This information regards signal strength and, in general, the state of the wireless card. There have been various attempts to standardize these headers. Among them, the one that had the greatest impact and success is Radiotap [11]. Designed initially for the NetBSD system, the Radiotap header format provides more flexibility than other headers, like the AVS [12] or the Prism ones. In fact, it allows the driver developers to specify an arbitrary number of fields based on a bitmask presence field. Through the Radiotap header we are able to read information like the signal's strength and packet's timestamp, thus extending the sniffer's functionalities.

In GNU/Linux operating system wireless devices are handled through the Wireless Extensions (WEs) [13]. WEs were added to the GNU/Linux kernel in 1997. They are now deprecated. Only bug fixing are accepted. They are in fact completely abandoned in favor of the new wireless stack, the mac80211. The main reason for this choice is that WEs are based on *ioctl* calls and, although ioctl has been and still is used as a standard transport for communication between user and kernel space, new means are now preferred for several reasons, but principally for the ioctl unstructured nature that lead to a new, undocumented, system call for each of its use.

The mac80211 wireless stack is key point for our purposes. Its architecture is built around the concept of virtual interface. Virtual interfaces can be created to handle the different modes in which a wireless card can operate in: for example, when a wireless card is in Managed mode, a "managed virtual interface" is created. Instead, if the wireless card running mode is Mesh, the "mesh virtual interface" is created and no direct access to the physical card is allowed. Virtual interfaces are seen by the operating system like every other device. An important feature of mac80211 is that there can be multiple virtual interfaces for a single physical card in the same moment and it is possible to use them simultaneously. This represents a crucial feature for our aims as explained in the following section.

III. LAURIN SOLUTION

Laurin is a sniffer developed with the intent of implementing all the functionalities of a common sniffer with the additional constraint of not losing network connectivity. It is a simple example to demonstrate how it is possible to develop desktop-user applications that take advantage of network monitoring capabilities.

Basically, to detect the presence of other nodes around, Laurin creates a virtual network interface that sniffs all packets in the air and reads the addresses in the MAC header, that declare sender, receiver, and Base Station (BS).

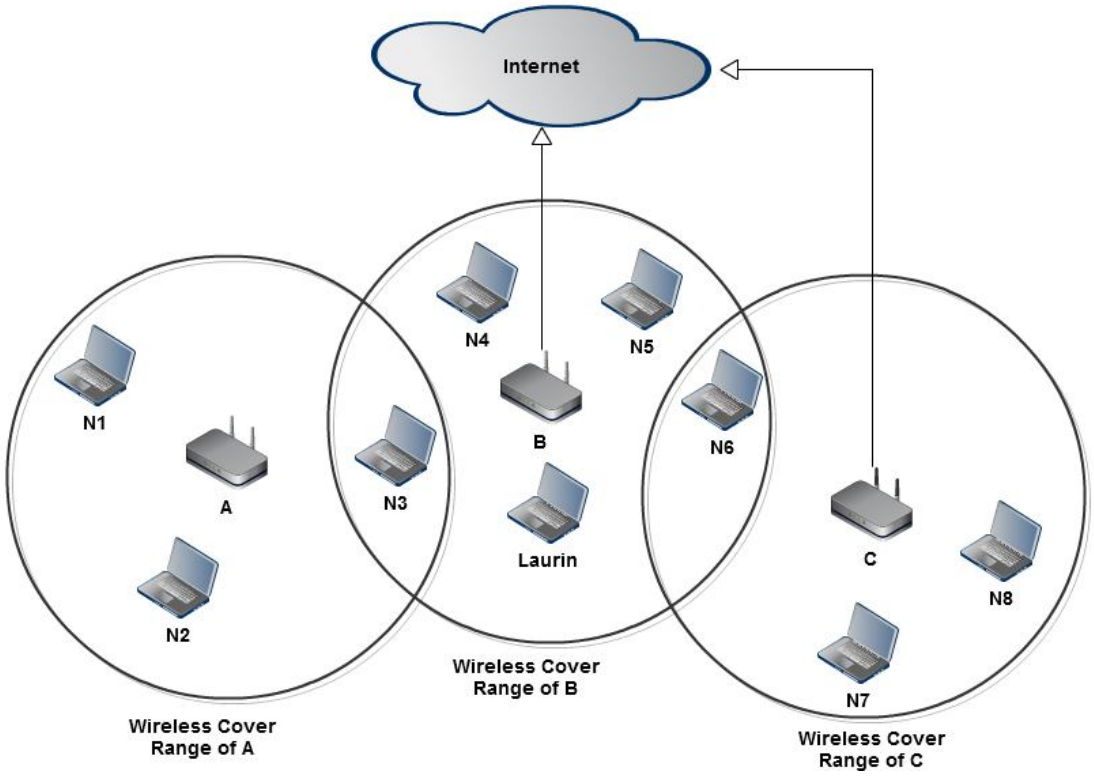

Figure 1: Test Scenario

Node Name	BS	Within Coverage of Laurin's Antenna?
N1	A	No
N2	A	No
N3	A	Yes
N4	B	Yes
N5	B	Yes
N6	C	Yes
N7	C	No
N8	C	No
A	A	No
B	B	Yes
C	C	No

Table 2: Test Scenario Wireless Topology

Ref.	Node Name	Node Name	BS
1	N1	N2	A
2	N2	N3	A
3	N7	N8	C
4	N6	N7	C
5	N6	N8	C
6	N4	B	B
7	N5	B	B

Table 2: Wireless Node Communications

As described above, the mac80211 wireless stack is able to create and handle virtual interfaces. The main contribution of this paper is to inspect how mac80211 works when a new virtual interface is added. Virtual interfaces can be added easily through the *nl80211* APIs. In essence, nl80211 is the new 802.11 netlink interface public header [12]. If it is possible to work with two virtual interfaces at the same time, it is reasonable to use one of them for regular network activities, such as wireless network association or data transmission and use a second one as a monitor. The latter can run in monitor mode and collect network traffic.

Note that working with two virtual devices is inherently different from having two physical devices because they share the same physical network adapter, so it is necessary to: i) ensure that it is still possible to implement all the sniffer's capabilities, ii) ensure that the network connectivity is not lost, iii) guarantee that the overall wireless performances is not negatively influenced. Yet, by doing so, we are able to simultaneously monitor the network environment without interrupting network connectivity supporting user's applications. This represents an important feature of our solution.

Laurin is developed with the intent to test our solution in practice: its goal is to enable the possibility for a node to detect the presence of other wireless nodes around, even a little further than the coverage range of its antenna. Laurin's purpose is twofold: first, to implement sniffers' capabilities in

a smart way and without interrupting regular network transmissions; second, to show that our solution opens the doors to the development of a new generation of desktop-oriented applications that take advantage of network monitoring service.

In order to work, Laurin implements three simple tasks:

- it adds a new virtual interface through the nl80211 APIs;
- it sets the newly created device in RFMON mode;
- it implements a simple sniffer routine through the pcap library [14] and collects information about wireless messages travelling through its cover area.

The pcap library is a set of procedures to collect network data. It is widely used and available for most common operating systems. Both the Wireshark and the Kismet projects use it for their low level operations

We have tried Laurin in the scenario of Fig. 1. In this scenario there are 12 wireless stations; Table 1 shows the wireless topology while Table 2 shows the ongoing communications among the considered wireless nodes. We refer to these communications through the value of the Ref. field. The BSs A and C are connected to the Internet. The Internet connectivity we utilized in our tests allowed for 5 Mbps as maximum download rate from a certain server and 0.4 Mbps of maximum upload rate.

Laurin collects wireless network traffic information and analyses it. For each frame acquired, it checks the sender, the receiver, and the BS field to detect the presence of wireless nodes even if they are not within the antenna's coverage range of the device implementing it. Table 3 shows, for each wireless node, whether its presence is detected or not. It shows the data flow and the field used for detecting the nodes. The outcome shows that all the wireless nodes discoverable through the analyses of network traffic are correctly detected by Laurin. Thereby, our solution is able to implement all the sniffer's capabilities. In the following section we demonstrate that while Laurin is monitoring the network, the wireless card is still able to transmit and receive data from the associated access point B. Moreover normal network operations are not influenced by the monitoring operations.

Node Name	Detected by Laurin	Detected Flow Ref.	Field Utilized
N1	×		
N2	√	2	Sender
N3	√	2	Receiver
N4	√	6	Sender
N5	√	7	Sender
N6	√	4, 5	Sender
N7	√	4	Receiver
N8	√	5	Receiver
A	√	2	BS
B	√	6, 7	BS
C	√	4, 5	BS

Table 3: Nodes detected by Laurin

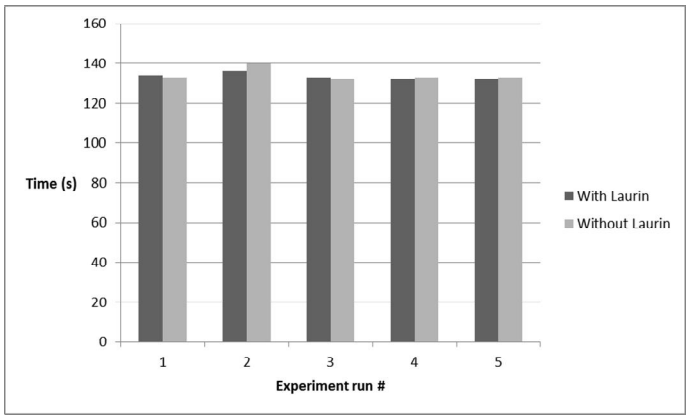

Figure 2: Download test at 5 Mbps.

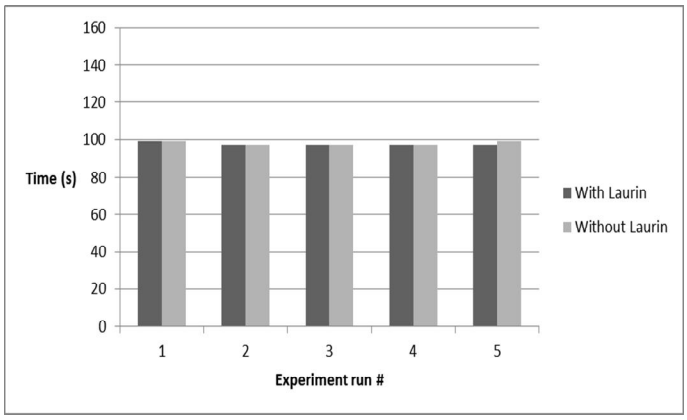

Figure 3: Upload test at 0.4 Mbps.

IV. BENCHMARKS

We need to guarantee that the simultaneous use of two virtual interfaces does not negatively affect the overall wireless performance. To demonstrate this point, we have made several download and upload tests, repeating experiments, with and without running Laurin.

More in detail, in Fig. 2 we report the average time necessary to download a 90 MB file with and without running Laurin. Looking at the results, we can easily infer that even if Laurin utilizes two virtual interfaces, the download performance is not affected at all. The charts show that the download times are independent from the use of Lauring (and hence of its simultaneous use of two virtual interfaces); they just depend on the network activities and bandwidth availability.

Furthermore, Fig. 3 shows the upload time for a 5 MB file. Again we can infer that the upload performance is not influenced by the presence of two virtual interfaces and hence the overall wireless performance is not influenced by using two or even more virtual devices

V. FUTURE WORKS

There are many interesting projects that are not widely diffused because they require dedicated hardware or because they limit the user network's operations. As we have shown, through the use of mac80211, we are able to use two or more wireless interfaces even if there is only one physical wireless adapter and would hence be interesting to revisit these projects in this new configuration. Among them, notably are the IEEE 802.11 location based services. Most of these services use the Time of Flight (TOA) approach, see [15]-[19]. The Radiotap header provides information about when the first bit of a frame of the MPDU arrives at the MAC level. If wireless card's driver exposes the Radiotap header, TOA technique, like the one presented by Hoene [19] can be implemented. Unfortunately, some statistical analyses are still mandatory due the low accuracy of wireless card's internal clock.

Other interesting initiatives are those based on finding efficient proactive mechanisms for the hidden node detection problem. In [20] knowledge of hidden nodes is obtained through the inspection of network traffic. With our solution, no change to the protocols is necessary, because, while using wireless card for normal user's operations, we are able to analyze network traffic too.

Again, intrusion detection systems or common sniffers, like [1]-[2] and [21]-[23] can now be implemented directly on users' computers, and hence it is possible to consider some kind of distributed intrusion detection system.

VI. CONCLUSION

With the new mac80211 wireless stack we are able to use more than one wireless device. We have shown that it is hence possible to monitor network activities by sniffing data without losing network connectivity. This is an important advantage with respect to state or the art sniffers such as Wireshark and Kismet, which, instead, set the wireless device in promiscuous mode, dropping any other prior communication. Furthermore, whereas Laurin has been specifically designed to show in a very clear way other wireless nodes around, solutions such as Kismet and Wireshark are pure sniffers that provide a trace list without extracting any information from it: the user has to check line by line the trace so as to find out the existence of other nodes.

Finally, we have shown that using more than one virtual device does not affect the overall wireless performance. Consequently, we are now able to implement applications that can take full advantage of network monitoring. Many interesting applications such as sniffers, intrusion detectors, and network monitors, so far bounded in specific scenarios can now be moved to a new use, being implemented in desktop-oriented applications.

REFERENCES

[1] Wireshark Project, http://www.wireshark.org

[2] Kismet Project, http://www.kismetwireless.org

[3] The SNMP Protocol, http://tools.ieft.org/html/rfc1157

[4] Working Group for Wireless Local Area Networks. IEEE Standard for Wireless LAN MAC and PHY Specifications. The IEEE 802.11 Protocol family, http://www.ieee802.org/11/

[5] G. Marfia, M. Roccetti, "Dealing with Wireless Links in the Era of Bandwidth Demanding Wireless Home Entertainment", in Proc. of 6th IEEE International Workshop on Networking Issues in Multimedia Entertainment (NIME'10), IEEE, Singapore, Jul 2010.

[6] A. Amoroso, M. Roccetti, "On the Making of an Ubiquitous and Altruistic Application for Medical First Responses", in Proc. of IEEE International Workshop on Ubiquitous Multimedia Systems and Applications (UMSA'09), St Petersburg, Russia, Oct 2009.

[7] C. E. Palazzi, "Residual Capacity Estimator for TCP on wired/wireless links", in Proc. of the WCC2004 Student Forum IFIP World Computer Congress 2004, Toulouse, France, Aug 2004.

[8] C. E. Palazzi, "Buddy-Finder: A Proposal for a Novel Entertainment Application for GSM", in Proc. of the 1st IEEE International Workshop on Networking Issues in Multimedia Entertainment (NIME'04), GLOBECOM 2004, Dallas, TX, USA, Nov 2004.

[9] The mac80211 wireless stack, http://linuxwireless.org

[10] Working Group for Ethernet based Local Area Networks. IEEE Standard for CSMA/CD (Ethernet) Access Method. The IEEE 802.3 Protocol, http://www.ieee802.org/3/

[11] The Radiotap Header, http://www.radiotap.org

[12] The AVS Header, http://multimedia.cx/avs-format.txt

[13] The Wireless Extensions http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html

[14] The Pcap Library, http://www.tcpdump.org

[15] M. Ciurana, F. Barcelo-Arroyo, S. Cugno, "A Novel TOA-based Indoor Tracking System over IEEE 802.11 Networks", in Proc. of IST Mobile and Wireless Communications Summit, Budapest, Hungary, Jul 2007.

[16] J. del Prado Pavon, S. Choi, "Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement", IEEE International Conference on Communications (ICC '03), Anchorage, AK, USA, May 2003.

[17] Y. Xiao, J. Rosdahl, "Throughput and Delay Limit of IEEE 802.11", IEEE Communications Letters, Vol.6, No.8, Aug 2002

[18] A. Gunther, C. Hoene, "Mesauring Round Trip Times to Determine the distance between WLAN nodes", in Proc. of IFIP Networking, Waterloo, Ontario, Canada, May 2005.

[19] C. Hoene, "Four-way TOA and Software-Based Trilateration of IEEE 80211 Devices ", IEEE 19th International Symphosium on Personal, Indoor and Mobile Radio Communications (PIMRC2008), Cannes, France, Sep 2008.

[20] F. Y. Li, A. Kristensen, P. Engelstad, "Passive and Active Hidden Terminal Detection in 802.11-based Ad Hoc Networks", in Proc. if 25th IEEE Conference on Computer Communications (Infocom 2006), Barcellona, Spain, Apr 2006.

[21] M. K. Chirumamilla, B. Ramamurthy, "Agent Based Intrusion Detection and Response System for Wireless LANs", in Proc of IEEE International Conference on Communications 2003 (ICC'03), Anchorage, AK, USA, May 2003.

[22] F. Kargl, A. Klenk, S. Schlott, M. Weber, "Advanced Detection of Selfish or Malicious Nodes in Ad Hoc Networks", in Proc. of 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 2004), Heidelberg, Germany, May 2004.

[23] M. Raya, J.P. Hubaux, I. Aad, "DOMINO: A System to Detect Greedy Behavior in IEEE 802.11 Hotspots", in Proc. of 2nd ACM International Conference on Mobile Systems, Applications, and Services (MobiSys2004), Boston, MA, USA, Jun 2004.