IEEE 802.11p Linux Kernel Implementation

R. Lisový, M. Sojka, Z. Hanzálek
Czech Technical University in Prague
{*lisovros,sojkam1,hanzalek*}*@fel.cvut.cz*

December 10, 2014

This document describes the results of the project to create an IEEE 802.11p implementation suitable for inclusion into the mainline Linux kernel.

The IEEE 802.11p amendment (part of the IEEE 802.11-2012) specifies the extensions to the IEEE Std 802.11 MAC and PHY specification providing wireless communications for a *vehicular environment* – thereby serving as a solid basis for cooperative safety Intelligent Transportation Systems (ITS). The extensions focus mainly on improving signal properties and lowering the time needed for link establishment.

The document begins with more detailed explanation of the IEEE 802.11p amendment contents. The IEEE 802.11 Linux kernel implementation description follows which is then complemented with the IEEE 802.11p Linux kernel implementation description. The document concludes with possible future extensions of the Linux IEEE 802.11 subsystem implementation.

Contents

Abbreviations and Glossary

AP Access Point

BSS Basic Service Set

BSSID Basic Service Set Identification

CCH Control Channel

CRDA Central Regulatory Domain Agent

DCC Decentralized Congestion Control

DSRC Dedicated Short-range Communications

EDCA Enhanced Distributed Channel Access

IBSS Independent BSS

ITS Intelligent Transportation Systems

MAC Media Access Control

MLME MAC Sublayer Management Entity

OBU On-board Unit

OCB Outside the Context of BSS

OID Object Identifier

PHY Physical Layer

RSU Roadside Unit

SCH Service Channel

SSID Service Set Identification

STA Station

TA Timing Advertisement

TPC Transmit Power Control

WAVE Wireless Access in Vehicular Environment

WLAN Wireless Local Area Networks

WNIC Wireless Network Interface Controller

Nomenclature

Despite this document already contains the section *Abbreviations and Glossary* it is of great importance to describe the proper use of different abbreviations more thoroughly since there is widespread misapprehension in the naming of different standards and technologies related to the topic.

The following list contains the most important abbreviations together with their short explanation.

C2C (C2I) abbreviates the term *car-to-car* (*car-to-infrastructure*) used mostly in conjunction with the word *communication*. This is the **most generic** term describing any possible technology used for wireless data exchange among the cars (or cars and infrastructure). Sometimes the abbreviation *V2V* (vehicle-to-vehicle) or *V2I* (vehicle-to-infrastructure) is used with the same meaning as the C2C (C2I). C2X does embrace both the C2C and C2I abbreviations.

IEEE 802.11p is an amendment of the IEEE 802.11 standard extending its media access control (MAC) and physical layer (PHY) specifications. Nowadays the amendment is fully included in the IEEE 802.11-2012 standard thus using the name IEEE 802.11p is only to emphasize that we refer to the parts of the standard related to a vehicular environment.

DSRC stands for *Dedicated Short-Range Communication*. This is a **generic** term describing one-way or two-way short-range to medium-range wireless communication channels specifically designed for automotive use and a corresponding set of protocols and standards. This relates to technologies such as C2C communication or electronic toll collection (i.e. this describes **neither** a single standard **nor** technology).

WAVE abbreviates the expression *Wireless Access in Vehicular Environments*. Since such phrase has a general meaning it is included in official names of different standards – *IEEE 802.11p Amendment 6: Wireless Access in Vehicular Environments* as well as *IEEE 1609 – Family of standards for Wireless Access in Vehicular Environments (WAVE)*. The term by itself **does not** describe any complete communication stack nor any integrated technology that may be deployed in the field. Therefore the usage of the plain term *WAVE* is vague and always requires further clarification.

ITS-G5 is a set of standards related to the ITS issued by the European Telecommunications Standards Institute (ETSI). They define (among others) C2C communication protocols build upon IEEE 802.11p.

IEEE 1609 is a family of standards (IEEE 1609.0, 1609.1, 1609.2 1609.3, 1609.4, 1609.11, 1609.12) for Wireless Access in Vehicular Environments (WAVE) built upon IEEE 802.11p. It defines an architecture and a complementary, standardized set of services and interfaces that collectively enable secure C2C and C2I wireless communications.

Despite the very frequent usage of the term WAVE in conjunction with the standard name IEEE 1609, the standard name **cannot** be replaced with the plain WAVE (or DSRC/WAVE) term. The name IEEE 1609 WAVE is however possible.

1. Introduction

1.1. Intelligent Transportation Systems

Hundreds of thousands of people are killed and injured in car accidents all around the world every year. The effort do decrease the number of killed and injured people leads to the development of Intelligent Transportation Systems (ITS) such that they can actively prevent car accidents and improve the traffic flow.

The technology described in this document is generally denoted with the term **car-to-car communication**. Its applications are cooperative safety, traffic and accident control, intersection collision avoidance, and emergency warning.

1.2. C2X communication in ITS

The idea behind the car-to-car communication in ITS is that each car equipped with an On-board Unit (OBU) will periodically transmit the information about (among others) its speed and location. The communication between mobile OBUs and stationary Roadside Units (RSU) mounted on traffic signs and traffic lights will be possible as well. These information will be used to mitigate possible car accidents.

The technology used is based on the widely accepted IEEE 802.11 standard family. This should help faster implementation (fast learning curve) along with lower initial cost.

2. IEEE 802.11 standard and 802.11p amendment

The IEEE 802.11 is a set of media access control (MAC) and physical layer (PHY) specifications for implementing wireless local area networks (WLAN). Nowadays it is the most popular solution for wireless data exchange used by many device types ranging from general-purpose computers to smartphones and consumer electronics.

In July 2010, IEEE published the IEEE 802.11p amendment. The full name is *Amendment 6: Wireless Access in Vehicular Environments*. It defines mechanisms that allow the IEEE 802.11 to be used in high speed radio environments typical for moving automobiles. It addresses challenges such as stronger Doppler shifts, rapidly changing multipath conditions, and the need to quickly establish a link and exchange data in very short times (less than 100 ms). The IEEE 802.11p amendment is already included in the IEEE 802.11-2012 standard. This document refers to the technology with the name IEEE 802.11p in order to emphasize that we refer to the parts of the standard related to a vehicular environment.

There are higher-level protocol stacks built upon IEEE 802.11p designated for co-operative safety ITS applications. There are two main differing protocol stacks – the ETSI ITS-G5 and the IEEE 1609. They may coexist next to each other however they cannot be interconnected nor do they complement each other. The ETSI ITS-G5 will be used in EU countries while the IEEE 1609 WAVE in U.S. (and Canada and maybe some other countries).

To overcome most of the limitations that operation in the *vehicular environment* is imposing, an IEEE 802.11p compliant network implements the following features:

OFDM (Orthogonal frequency-division multiplexing) modulation scheme (the same as in IEEE 802.11a) is used. Only the mandatory datarates (3, 6 and 12 Mb/s) are used – that ensures that *all* the STAs in the network are reached.

5.9 GHz frequency band is mandatory for operation. Different countries (U.S. and Canada, EU, South Korea, Japan, Australia, China, etc.) reserved 5.9 GHz band with slightly different boundaries intersecting at around 5890 MHz. The band is divided to several channels. Due to the timing constraints, is not possible to scan which channel is used (as in classical Wi-Fi) and this the channel must be known beforehand. One channel is defined as a Control Channel (CCH) dedicated for ITS road safety. The other channels are so-called Service Channels (SCH) that might be used for ITS road safety as well as for arbitrary data exchange (weather forecast, in-field firmware update, etc.).

Half-/quarter-rate channels (5/10 MHz) are used. The OFDM approach uses the same modulations (BPSK, QPSK, 16-QAM, 64-QAM) as IEEE 802.11a, but each modulation scheme results in half the data throughput since the duration of each symbol is twice as long within the 10 MHz channel width (in comparison to 20 MHz channels used by IEEE 802.11a). This improves the resistance to the effects of Doppler spread and mitigates inter-symbol interference in a channel.

Timing Advertisement (TA) is a new Management frame added by the IEEE 802.11p. This frame is used by STAs to inform other STAs about the value of time. The standard defines this extension as optional for implementation.

OCB (Outside the Context of a BSS) mode is a newly added mode that, simply put, enables all the STAs in the range to directly communicate with each other – neither authentication/association procedures nor security mechanisms are supported, thus the data exchange might be established in fractions of seconds.

The OCB mode properties are detailed below:

- **Wildcard BSSID** (i.e. BSSID where all bits are set to 1) is used by each STA.
- **No beacons** are being transmitted or received.
- **EDCA parameters** are different from those specified by IEEE 802.11e.
- **No Authentication** is used. Every node may *join* the network and be part of it.
- **No Association** is used since there is no notion of an AP in the OCB mode.
- **No Encryption** is used. Security properties are ensured by higher level protocols.
- **dot11OCBActivated** OID variable is set to *true*.

3. IEEE 802.11 Linux kernel support

In the past, the Linux kernel contained several different IEEE 802.11 driver frameworks. Nowadays there are two – Wireless-Extensions (Wext) being slowly pushed out of the kernel remaining just for the backward compatibility reasons[1] and the mac80211/cfg80211 which is the preferred framework used by most of the drivers.

The different components of the main Linux IEEE 802.11 framework (see Fig. 3.1) are described further in this section.

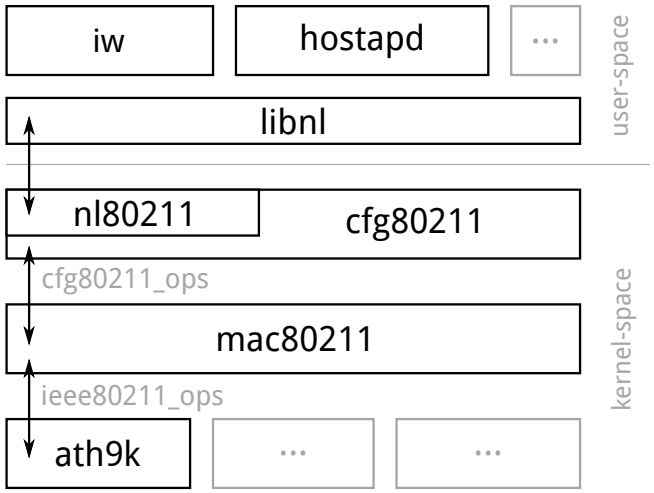

Figure 3.1.: IEEE 802.11 Linux kernel implementation architecture

3.1. mac80211

mac80211 is a framework used for writing drivers for SoftMAC wireless devices.

SoftMAC is the term used to describe a type of WNIC where the MLME is expected to be managed in software. Such devices allow for a finer control of the hardware, allowing for 802.11 frame management to be done in software for them, for both parsing and generation of 802.11 wireless frames. Most 802.11 devices today tend to be of this type.

[1] There is no further development of the Wext, only bugfixes are being accepted. Future versions of Linux kernel will not contain it at all.

In mac80211 the MLME is done in the kernel for station mode (STA) and in userspace for AP mode (hostapd).

mac80211 depends on cfg80211 for both registration to the networking subsystem and for configuration.

3.2. cfg80211

cfg80211 is the configuration API for IEEE 802.11 (mac80211) devices in Linux. It bridges userspace and drivers, and offers some utility functionality associated with IEEE 802.11. cfg80211 must be used, directly or indirectly via mac80211, by all modern wireless drivers in Linux, so that they offer a consistent API via nl80211.

Additionally, cfg80211 contains code to help enforce regulatory spectrum use restrictions.

3.3. nl80211

The purpose of the *nl80211* is to bridge communication between userspace configuration tools and the in-kernel cfg80211 subsystem. To do so in a flexible way the *Netlink* protocol is being used.

3.4. iw

iw is a userspace nl80211-based command-line utility used for configuration of wireless devices.

The documentation may be obtained from `http://wireless.kernel.org/en/users/Documentation/iw` or by executing the *iw help* command.

Another tool – *iwconfig* – was used in conjunction with wireless extensions and is deprecated and will be dropped in the future.

3.5. CRDA

CRDA (Central Regulatory Domain Agent) acts as a helper for communication between the kernel and userspace for regulatory compliance. It relies on nl80211 for communication. CRDA is just the tool itself, the regulatory database used by CRDA is called *wireless-regdb*. It is used mainly by Linux but the intention is that it will be used on other platforms (open or proprietary) as well.

4. IEEE 802.11p Linux implementation

To posses a fully functional IEEE 802.11p Linux implementation it is necessary to perform the following changes to a generic Linux system:

- add the OCB mode support to the mac80211 subsystem
- add the OCB mode configuration to the cfg80211 subsystem
- modify a WNIC driver (e.g. Qualcomm Atheros AR93xx) to support the OCB mode as well as enabling the operation in 5.9 GHz frequency band
- add the notion of the OCB mode to the configuration tool iw
- add the 5.9 GHz band rules into the regulatory database wireless-regdb

All these changes were successfully implemented and tested. The most extensive changes were made in the mac80211 and cfg80211 subsystems. These are already on its way to the mainline Linux kernel[1]. All the other changes are planned to be upstreamed later on.

More detailed explanation of particular changes follows.

4.1. cfg80211

The very first change to the Linux kernel was the addition of the OCB mode configuration functionality to the cfg80211 subsystem. The properties of the OCB mode to be configured are:

- setting the mode of the interface to be OCB (i.e. `NL80211_IFTYPE_OCB`),
- virtually *joining* the OCB network on particular frequency with particular channel width (nl80211 message `NL80211_CMD_JOIN_OCB`),
- *leaving* the OCB network, i.e. disabling the RX/TX functionality (nl80211 message `NL80211_CMD_LEAVE_OCB`).

[1] https://git.kernel.org/cgit/linux/kernel/git/next/linux-next.git/commit/?id=6e0bd6c35b021dc73a81ebd1ef79761233c48b50
https://git.kernel.org/cgit/linux/kernel/git/next/linux-next.git/commit/?id=239281f803e2efdb77d906ef296086b6917e5d71

4.2. mac80211

The configuration messages directly influence the main logic implemented in the mac80211 subsystem. The changes to mac80211 comprised:

- setting the BSSID to a fixed wildcard value (all bits set to 1),
- keeping track of newly discovered STAs (marking them as Authenticated and Associated immediately),
- periodically invoking housekeeping thread to remove previously discovered non-active STAs,
- RX/TX path modification,
- 802.11p specific EDCA parameter setting,
- adding OCB related printouts to the debugging subsystem.

4.3. nl80211

Since the nl80211 subsystem is used to exchange configuration commands between the kernel and the userspace, it was necessary to add two nl80211 commands – one for joining the network, the other for leaving it.

4.4. ath9k

The previously described changes were made to the generic IEEE 802.11 subsystem. To actually test the implementation, a WNIC driver modification is required. Not every WNIC can operate in the 5.9 GHz band with 10 MHz wide channels or the driver modification to enable this operation mode is not publicly known. Such information is often available only after signing a non-disclosure agreement with the vendor. The WNICs belonging to the Qualcomm Atheros AR93xx family (supported by the ath9k Linux driver) are well known to support the required properties and the driver modification is straightforward. Channels with 10 MHz bandwidth are already supported, tuning to the 5.9 GHz band is supported and only needs to be enabled.

The changes made to the ath9k driver enabled the channels 170–185 (5850–5925 MHz) and added the OCB mode handling where necessary (in a very similar fashion as the IBSS mode is handled).

4.5. iw

All the previously described changes were made in the Linux kernel code base. To make all of this usable and configurable by a user, some changes to the userspace

configuration utility iw are necessary. The changes include adding of the *ocb join* and *ocb leave* commands.

Appendix A shows an example of how these commands can be executed by a user.

5. Conclusion and future work

The work described in this document resulted in the fully functional IEEE 802.11p implementation that is on its way to the mainline Linux kernel. Despite this being a significant contribution to the Linux kernel, there is still a lot of work left to be done to have a fully functional integrated Linux-based C2C communication protocol stack for ITS.

The possible future extensions are listed below:

- IEEE 802.11 and regulatory requirements mandates requires Transmit Power Control (TPC) to be used for some channels (in 5 GHz band mostly) when higher transmit power is required. Linux implementation of the IEEE 802.11 does not support TPC thus the lowest possible transmit power is used on these particular channels. TPC implementation would be beneficial for the generic IEEE 802.11 Linux implementation as well as for the IEEE 802.11p.

- The only ratecontrol algorithm used in Linux is the *Minstrel* algorithm, however the ITS-G5 standard defines its own more complex ratecontrol algorithm called Decentralized Congestion Control (DCC). The proper implementation of the DCC would comprise of one part being in the kernel and another in userspace.

- To be able to use the 5.9 GHz band, the necessary channels were added to the *minimal set of supported channels* in the regulatory rules of the ath9k driver. This is just a temporary workaround. The proper implementation would require to modify the wireless-regdb (and CRDA) such that the notion of the 5.9 GHz band will be in the whole IEEE 802.11 subsystem.

- One possible interpretation of the IEEE 802.11p standard says that the OCB mode wireless interface may operate in other frequency bands than the dedicated 5.9 GHz as well as the *regular* IEEE 802.11 traffic might use the 5.9 GHz frequency band. Such operation would be against the intended purpose of the IEEE 802.11p network and may violate the regulatory rules.

  This kind of behavior might however be beneficial for C2C research and development on Linux. To satisfy the requirements of both sides this behavior might be allowed under the condition that the configuration option `CONFIG_CFG80211_CERTIFICATION_ONUS` (disabled by default in Linux kernel and in all Linux distributions) will be set to true.

- Adding support of the OCB mode to other WNICs would be beneficial.

- To be able to operate a fully functional C2C communication stack for ITS the upper layer protocols are needed. This may lead to the Linux implementation of the ITS-G5 (or IEEE 1609) protocol stack.

Acknowledgment

This work was supported by Volkswagen AG. We would like to thank namely to Jan-Niklas Meier and Dr. Burak Şimşek.

A. Wireless interface OCB mode configuration in Linux

The following set of commands shows how to properly

- set a regulatory domain (let's assume the chosen one does include rules for 5.9 GHz band),
- configure a wireless interface for the OCB mode operation and
- connect to the OCB network on a particular frequency (with 10 MHz wide channel).

```
iw reg set DE                       # Set the proper regdomain
ip link set wlan0 down
iw dev wlan0 set type ocb           # Set iface mode to OCB
ip link set wlan0 up
iw dev wlan0 ocb join 5890 10MHZ    # Use the ITS-G5 CCH channel
```

Bibliography

[1] "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 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, March 2012.

[2] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an international standard for wireless access in vehicular environments," in *Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE.* IEEE, 2008, pp. 2036–2040.

[3] E. G. Ström, "On medium access and physical layer standards for cooperative intelligent transport systems in Europe," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1183–1188, 2011.

[4] Y. Li, "An Overview of the DSRC/WAVE Technology," in *Quality, Reliability, Security and Robustness in Heterogeneous Networks*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, X. Zhang and D. Qiao, Eds. Springer Berlin Heidelberg, 2012, vol. 74, pp. 544–558. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-29222-4_38

[5] R. Uzcategui and G. Acosta-Marum, "Wave: A tutorial," *Communications Magazine, IEEE*, vol. 47, no. 5, pp. 126–133, May 2009.