IEEE P802.11
Wireless LANs

Translation vs Encapsulation: The need for an implementors Agreement

Date: November 1997

Author: Pablo Brenner
BreezeCom
Phone: +598-2-7112843
e-Mail:pablob@breezecom.com

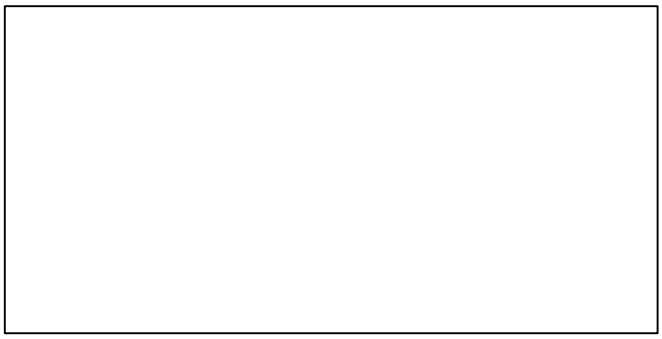
Introduction

The IEEE P802.11 Standard defines the MAC and PHY layers for an implementation of a Wireless LAN, and does specify the logical primitives for interfacing to higher layers. While this is enough for independent implementors to build interoperable systems at the MAC and PHY layer, there are still some implementation issues that need to be agreed in order to build products that are interoperable when connecting them in real world applications, and one of them is:
How does a bridging Portal (in most cases implemented in the AP) translate Ethernet frames into 802.11 frames and viceversa?

The objective of this document is to describe the different approaches of translation, their difficulties, and the advantages and disadvantages of each one, trying to give all the information in such a way that the majority of the 802.11 implementors will implement the same approach and by thus ensuring interoperability to the end-users.

An ideal world situation

Theoretically according to the OSI model and the 802 Layering approach, there should be no such problem, each layer has its own header (and sometimes trailer), which is added when transmiting and removed after reception, so an 802.1 bridge (sitting above two MAC Layers, for example 802.11 and 802.3), would just receive an indication of a frame reception from one of the layers, and depending on his own decisions decide to retransmit through the second interface by sending a Transmit_Request primitive to the other interface, as shown in the following diagram:



In this case an 802.11 frame which needed to be forwarded by the bridge to the 802.3 interface would be translated in such a way that the 802.2 header would remain the same and only the 802.11 would be translated to an 802.3 header as follows:

802.11 DA → 802.3 DA
802.11 SA → 802.3 SA
Calculated length → 802.3 Length
Calculated CRC → 802.3 CRC

And a similar translation could be done on the opposite direction.

But life is not ideal

And some of the most widely spread LAN applications do not use the 802.3 architecture (that has length as part of the header and assumes the use of 802.2 on the upper layer), but use the older Ethernet MAC layer (which is basically the same but instead of "length" has "type" in the header), and do not use an 802.2 layer above (the most popular protocol using Ethernet instead of 802.3 is TCP/IP).

In this case the transparent translation that we evaluated before is impossible, because there is a new field, "type" which does not exist as part of the 802.11 header, so we would loose it when translating from Ethernet to 802.11 or we could not know what to add when translating from 802.11 to Ethernet.

Let's do not reinvent the wheel

This problem is well known and was dealt with any time a new MAC Layer protocol was designed (e.g FDDI), and there always been two approaches: Encapsulation (or tunneling) and Translation.

Encapsulation: This is the most simple method, in this case the Ethernet packet (or 802.3) is encapsulated (including the Ethernet Header and Trailer) as data in the 802.11 packet as shown in the diagram.

802.11 Header	Ethernet Header	Upper Layers Data	Ethernet Trailer	802.11 Trailer

Advantages:

1. The method is completely straightforward, there are no special cases, and the implementation is easy
2. Keeping the original Ethernet CRC, end to end, allows to recognize packet corruption performed inside the AP (although this should never occur, a memory corruption could cause the packet to be corrupted inside the device, and if CRC is generated only upon transmission it would not be recognized as an error)

Disadvantages:

1. The method is not defined by any standard, neither compliant to the OSI Seven Layers, there are two MAC layers, where according to the OSI Layering (and 802) above the 802.11 we should find 802.2 (LLC) and not another MAC layer (802.3). There will be always standard purists that will claim that this is not permitted.[1]
2. It will require devices that have no reason to know the Ethernet packet structure (e.g. 802.11 NIC cards, or Token Ring APs) to use Ethernet Headers.
3. Adds an overhead of 18 bytes (14 bytes of Ethernet Header plus 4 Bytes CRC) to any packet.

[1] According to the author's experience, is hard to defend this mechanism when a competitor is claiming that the method is non-standard and not according to the OSI layering architecture

Translation: This method is somehow more complicated, for full description of this method the author recommends to check 802.1h and/or RFC 1042, but here is a short description on how the translation is performed differently for the different types of frames as follows:

Translating from Ethernet/802.3 to 802.11
If the frame received is 802.3 (we know that by checking the length/type field, if it is equal or below 1500, then the field represents length, and the frame is 802.3, otherwise is Ethernet), then it is safe to assume that above the MAC layer there is an 802.2 layer, so the translation is performed transparently, the 802.3 header and trailers are replaced by 802.11 headers and trailers respectively.

But if the frame is Ethernet, then we must save the "type" field in some place on the new 802.11 frame, which has no such field defined. The way for doing that is by using a well known extension of the LLC header, known as SNAP, which defines that when the DSAP and SSAP fields of the 802.2 header are both equal to 0xAA, then there are 5 more bytes added to the LLC header (which is normally 3 bytes long): the first 3 bytes are the vendors OUI (Organization Unique Identifier) and the last two are the PID (Protocol Identifier)[2]. RFC1042 and 802.1h define the use of OUI of all zeros (00-00-00) to indicate that the 2 bytes of the PID store the Ethernet "type" field.

So if the incoming packet is Ethernet then it will be translated into an 802.11 frame, with SNAP and an OUI of all zeros and the "type" will be copied into the PID field

Translating from 802.11 to Ethernet/802.3
In this case the reverse procedure is performed, if the packet has an SNAP header with OUI of all zeros then it is converted into an Ethernet frame, otherwise it is transparently translated into an 802.3 frame. Please note that 802.1h defines some special cases where the SNAP packet is not automatically translated..

Advantages:
1. The method is standard and any frame captured on the WLAN will be completely standard, having the 802.11 header, an 802.2 header and higher layers.
2. The overhead on 802.3 is zero.

Disadvantages:
1. The implementation is more difficult, where each packet has to be evaluated to decide how to translate it.

Conclusion

The document describes the two well known mechanisms for overcoming the said problem, the 802.11 implementors must select one and only one of this mechanisms, and everybody should implement the same, otherwise, we'll get into the absurd situation that after 6 years of standardization effort, we will still not be interoperable, and the market will get a very bad impression of the standard.
From the author's past experience, the 802.1h mechanism although more difficult to implement has better chances of being wide accepted, and is backed by many standards bodies like 802.1 and IETF, and sounds like the safer choice, and would recommend implementing this option, but much worst than choosing any of the options is not choosing one.

[2] This mechanism was originally defined for allowing vendors to develop private protocols over 802.2