Security of IEEE 802.11b

Thesis project done at Information Theory,
Linköping University
by

Johan Skoglund

LiTH-ISY-EX-3370-2003

Linköping, 2003

Security of IEEE 802.11b

Thesis project done at Information Theory,
Linköping University
by

Johan Skoglund

LiTH-ISY-EX-3370-2003

Examiner:Viiveke Fåk

Supervisor:Fredrik Johansson

Linköping, March 19, 2003

Avdelning, Institution
Division, Department

Institutionen för Systemteknik
581 83 LINKÖPING

Datum
Date
2003-03-19

Språk
Language
Svenska/Swedish
X Engelska/English

Rapporttyp
Report category
Licentiatavhandling
X Examensarbete
C-uppsats
D-uppsats
Övrig rapport
____

ISBN

ISRN LITH-ISY-EX-3370-2003

Serietitel och serienummer **ISSN**
Title of series, numbering

URL för elektronisk version
http://www.ep.liu.se/exjobb/isy/2003/3370/

Titel Säkerhet i IEEE 802.11b
Title
Security of IEEE 802.11b

Författare Johan Skoglund
Author

Sammanfattning
Abstract

The IEEE 802.11b standard is today the only commonly used standard in Europe for fast wireless networks. This makes it possible to connect computers to networks in places where it is not possible to use wires. Examples of such situations are internet access at airports, communication in emergency areas or for military communication. Common for all these situations is that network security is important.

This thesis consists of two different parts. The first part handles the security mechanisms and the second part is an evaluation of the possibilities to use IEEE 802.11b in embedded applications. The part that handles the security includes the security mechanisms found in the standard, flaws in these mechanisms and methods that try to reduce these problems.

Nyckelord
Keyword
wlan, IEEE 802.11, computer security, embedded system, 802.11, wireless lan

Abstract

The IEEE 802.11b standard is today the only commonly used standard in Europe for fast wireless networks. This makes it possible to connect computers to networks in places where it is not possible to use wires. Examples of such situations are internet access at airports, communication in emergency areas or for military communication. Common for all these situations is that network security is important.

This thesis consists of two different parts. The first part handles the security mechanisms and the second part is an evaluation of the possibilities to use IEEE 802.11b in embedded applications. The part that handles the security includes the security mechanisms found in the standard, flaws in these mechanisms and methods that try to reduce these problems.

Table of contents

1 Abbreviations

ACK Acknowledge

AES Advanced encryption standard

AP Access point
Is used in "Infrastructure mode" and is the same as a base station for cell phones.

ARM Advanced RISC Machines

CPU Central processing unit

CRC Cyclic redundancy check
An algorithm often used to verify that packets are transmitted without errors. It is good at detecting random errors in messages and requires little hardware. It is not useful as protection against deliberate modifications.

CTS Clear to send

DMA Direct memory access

DoS Denial of service

DS Distribution system
A network used in the 802.11 standard that distributes traffic between different Access points.

DSSS Direct sequence spread spectrum

EAP Extensible authentication protocol

FFI Forsvarets forsknings institutt

FHSS Frequency hopping spread spectrum

FOI Totalförsvarets forskningsinstitut

FPGA Field programmable gate-array

ICV Integrity check value. A value sent in packets whose purpose is to prevent modification of messages.

IEEE Institute of electrical and electronics engineers

IP Internet protocol

IV Initialization vector. This is used as a seed for the cipher to ensure that each packet is encrypted different, although the same key is used.

LAN Local area network

MAC Media access control

OFDM Orthogonal frequency division multiplexing

NIC Network interface card

RC4 Rons code 4 or Rivest cipher 4. A stream cipher invented by Ronald Rivest.

RTOS Real-time operating system

RTS Request to send

SSID Service Set Identifier

OSI Open System Interconnection

QOS Quality of service

TKIP Temporal key integrity protocol

VPN Virtual private network

WEP Wired equivalent privacy

2 Introduction

This chapter contains a short introduction to the thesis. Why this subject is interesting, and finally a reading instruction.

2.1 Background

The IEEE 802.11 standard, with its different versions, is nowadays the only commonly used standard for high-speed wireless networks for computers. This makes the standard very useful for a lot of applications. Not only for those applications it is primarily designed for, because it is often cheaper to modify an existing standard that is carefully tested and with high volume production than developing a new one.

2.2 Purpose

The purpose of this thesis is to introduce the IEEE 802.11 standard for Sectra Communications AB. The thesis mainly investigates the security aspects in the standard. What kind of security functions exist and how do they work. The second part of the thesis is a practical test that investigates the possibilities to use the standard in an embedded system. This part is done as a demonstration application that runs on an ARM based embedded system. This application shows the performance and practical problems with such systems.

2.3 Reading Instructions

Chapter 3 is an introduction to network security. It introduces the threats for secure communication and finally how stream ciphers works.

Chapter 4 continues with network security but this time from a military perspective.

Chapter 5 gives the reader an introduction to the 802.11 standard.

Chapter 6 describes more detailed how the standard works.

Chapter 7 is focused on Ad hoc networks and threats specific for such networks.

Chapter 8 describes the different methods given in the standard that aims to increase the security.

Chapter 9 explains why the security is not so good in spite of all the security methods.

Chapter 10 gives examples of methods that try to increase the security.

Chapter 11 describes what we can expect from the new 802.11i standard, which aims at the security flaws.

Chapter 12 is a summary of the security parts.

Chapter 13 contains the results from the practical part.

Chapter 14 gives a summary of the results.

In the appendixes are more detailed descriptions of the test program, RC4, Weak IV and CRC32. These parts should be read if detailed information are needed but are not necessary for understanding the report.

2.4 Limitations

This thesis is mainly focused on the IEEE 802.11b standard because this is the most commonly used wireless standard today. Since many parts e.g. the security is common in 802.11, 802.11a and 802.11b am I going to write 802.11 when I intend these three standards. When I intend the 802.11 standard am I going to write IEEE 802.11 and when I intend all different 802.11 standards, i.e. 802.11 plus 802.11a to 802.11i, am I going to write 802.11x.

The demonstration was supposed to give an example of how 802.11 could be safer. This was however not done due to limitations in time.

3 Introduction to network security

This chapter contains an introduction to network security, and also gives examples of possible threats. For more detailed information see [11].

3.1 Threats

As soon as something is transferred in a network there is a need of network security. One example that shows this is when A is sending a file to B and another person C is also listening to the transmission. In this case both A and B have some requirements for the transmission if it should be useful to transfer the file:

- Confidentiality
  Only authorized persons should be able to read the file. C might listen to the transmission but he should not be able to read the file.
- Authentication
  B must know that A and not C sent the file.
- Integrity
  The file received by B should be identical to the file sent by A. C should not be able to modify the transmission.
- Nonrepudiation
  A should be able to prove that B has received the file and B should be able to prove that A sent the file.
- Access control
  The transmission has access control that allows A and B to communicate. This should not be possible change by C.
- Availability
  A should always be able to send the file to B regardless what C does.

3.2 Different attacks

C has four general categories of methods to use if he wants to get information or disturb the transmission:

- Interruption
  Stop all traffic.
- Interception
  Eavesdrop the traffic.
- Modification or man-in-the-middle
  Modify chosen parts of the traffic.

- Fabrication
  Fabricate packets to send.

These attacks are categorized into passive attacks and active attacks.

3.2.1 Passive attacks

Passive attacks are attacks that listen to the traffic. There are two types of passive attacks, release of message contents and traffic analysis.

Release of message contents is when the data is directly accessible for the attacker.

Traffic analysis is more advanced. This attack does not use the data that is being sent. Instead, it extracts information from the pattern of the traffic. The attack looks at things like who are communicating, when and how much information is sent.

3.2.2 Active attacks

Active attacks involve some kind of modification of the data stream. Active attacks are divided into four categories: masquerade, replay, modification of messages and denial of service.

Masquerade is when one entity pretends to be another entity.

Replay is passive capture of data that later is being retransmitted to produce unwanted effects.

Modification of messages means that some part of a correct message is modified. One example of this is the so-called man-in-the-middle, where the attacker sets up a service that pretends to be the original one. When a user connects to the attacker’s service, the attacker might get the user’s username and password. This information is then used to connect to the correct service. This way the user will not recognize that he connected to the wrong service but the attacker gets access to the service.

Denial of service (DoS) is an attack that has a different purpose than the other attacks. The purpose of a DoS attack is to prevent the normal use of a service. The target for such attacks can be a specific server but it can also be a complete network.

3.3 Security methods

The solution to many of the security problems is to use cryptography and hash functions. This chapter is therefore going to show an example how these problems can be solved. There are mainly two problems that have to be solved when we want to increase the security. The first problem is that we want to prevent eavesdropping and the second problem is that we must know which one we are communicating with.

Encryption of the communication solves many of the problems in chapter 3.2 if the encryption is correctly used. Encryption prevents eavesdropping but it also ensures that the data is correctly received.

Some kind of authentication is needed if we want to be able to know to whom we are communicating. A good authentication algorithm should not only authorize the user but also the server. This is important because it prevents the man-in-the-middle attack.

3.4 Security in Wireless LAN

The security aspects in wireless LANs have the same problems as the security problem in ordinary LANs. The main difference is that a wireless LAN is much easier to access. To be able to use an ordinary network, physical access is needed. A wire needs to be connected to the correct socket but this is not required in a wireless network. The radio signals do not stop just because they are leaving the house. Instead is it possible to listen to such networks from long distances if the right equipment is available. Therefore it is very common that wireless networks contain both methods for encryption and authentication.

3.5 Stream ciphers

Ciphers can be divided into two different categories, stream ciphers and block ciphers. A stream cipher encrypts the data one symbol at a time. Block ciphers on the other hand encrypts a whole block consisting of several symbols at a time. 802.11 uses a stream cipher called RC4 when it encrypts the data. For a more detailed explanation of RC4, see Appendix B: RC4.

A stream cipher consists of two different parts. One part generates a sequence of symbols that is called the keystream. The other part of the cipher performs the encryption when it combines this keystream with the plain text. The combination between plain text and keystream is usually done with XOR but also other operations could be used.

If the same key is used to encrypt more than one message, the same keystream will be used to encrypt all of them. This can be used to get information about data encrypted with the same key. The frequency is different for different symbols in a text and this will be conserved if we look at a single position in the different ciphertexts. If the text for example is written in English, 12.75% of the characters will be “e“ but only 0.5% be “x”. If an attacker receives enough different ciphertexts, he can use this to get information about the plaintexts. Another problem is that if the attacker somehow manages to get the plain text, he will be able to calculate the keystream used for encryption. With this keystream it is then possible to read all other messages encrypted with this key.

The solution to this problem is to use the key for only one message. The best solution would be to generate a completely new key for each message. Unfortunately this solution is very impractical and therefore another solution is used. The solution is to use something called IV (initialization vector). This IV is mixed with the secret key to generate the key used for encryption. This IV is then sent in plain text to the receiver. Due to the problem with messages encrypted with the same key it is important to change the key before all IVs are used, because otherwise we will get the same phenomenon.

4 Wlan for military applications

Wlan is a standard that is interesting to use in military networks, see [3]. This chapter contains the background and an introduction to one such project done by NATO.

The 802.11 standard is interesting to use because it is a common standard for civilian networks and if it was possible to modify it to suit military needs, it would also be a very economic alternative. Military requirements are sometimes different than the civilian requirements and this chapter does therefore contain a summary for one such project.

4.1 Military needs

A product used by the military has requirements that are stricter than a similar product for civilian usage. The most important differences for the requirements are these:

- Useful for longer distances.
  The short range that 802.11 offers is seldom useful, especially in a forest.
- Not have a “single point of failure”.
  It is important that the system is robust. The whole network should not require that one specific unit works.
- Security/jamming.
  The network must be hard to eavesdrop and jam because the transmitted data is often secret.

4.2 Increase the range

The physical layer that exists in 802.11 is not directly useful for military use because it gives a too short range for the network. Instead the physical layer has to be modified somehow to give this longer range. FFI (counterpart to FOI in Norway) has done experiments by changing the frequency from 2.4 Ghz to 300 MHz using a transverter.

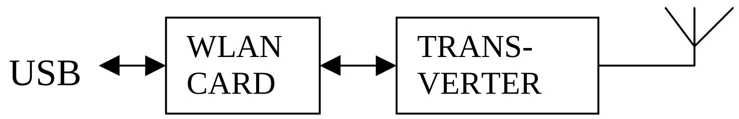

Figure 1. NIC using transverter to change frequency

There is a couple of attracting properties with this solution:

- Lower frequency gives longer range
- Military reserved frequency with a higher allowed transmission power
- Uses civilian Wlan cards which are cheap

The experiment showed that the range was extended to some hundred meters, but it also sometimes seemed to have problems with long multipath interference. One possible solution to this problem is to use even lower transmission rates.

Other organizations have done similar experiments but with other frequencies, e.g. is a Dutch company called TNO-FEL working on a transverter that changes the frequency to 4.4-5.0 GHz. This frequency band is reserved for military usage. The transmission power is allowed to be higher and therefore it is possible to compensate for the shorter range given by the high frequency.

Which band that is best to use depends on the requirements for the product. In general a lower frequency has a longer range than a higher frequency. This means that a network using a lower frequency requires fewer access points, but this will reduce maximum throughput because every cell will become larger i.e. more clients per access point, and it might also be easier to eavesdrop because of the larger range.

4.3 Single point of failure

One important property for a network that will be used by the military is that it is robust. It shall not be possible to destroy the whole network by destroying just one part of it. This implies that the network can not be used in “Infrastructure” mode because such a network completely relies on the central access point (AP) which corresponds to a base station in mobile telephony. Instead the network has to work in “Ad hoc” mode which allows the users to directly communicate with each other without the need of a central AP. However introduces Ad hoc networks new problems that have to be solved like routing and similar problems. Ad hoc networks are more carefully described in chapter 7.

4.4 Security/jamming

The security provided by WEP is not strong enough for military use. Therefore it has to be somehow improved, which means that some kind of encryption has to be added to the application. This encryption could be done at different layers in the OSI model. The OSI model is a theoretical model that describes how data is transported from one application to another using a network. This model is described in chapter 5.3. The encryption could either be at layer 3 i.e. encryption of the contents of the IP-packets, in layer 2 which will encrypt the information in the Ethernet packets or in layer 1 which will encrypt the data sent out in the air.

The problem with jamming is harder to solve. One possible solution could be to use frequency hopping instead of using a static frequency to make it harder to jam.

5 Introduction to 802.11

Almost every computer that is used today is connected to a network with a cable. This works very well when the computer is placed in areas with good possibilities to connect to existing networks, but what happens if that is not possible or when the computer is mobile?

One very interesting solution to this problem would be to use a wireless network, i.e. a network where the computers not are connected to each other using wires but instead are communicating using radio.

5.1 Where to get the standard

A standardization association called IEEE has written the specification. All different 802.11x standards that are older than six months are available for free downloading from their homepage, see [7].

Beside the standards there is also another association that is involved with the development of 802.11. This is the “Wi-Fi Alliance”, see [8]. They are testing different 802.11 products to assure that they follow the standard and therefore are able to communicate with each other. This is important, because a standard often is possible to interpret in different ways. Another problem is that some parts of the standard are optional, i.e. more or less suggestions on how to improve the product. Sometimes it might be important if these are implemented or not.

5.2 Different 802.11 standards

This is a short description of all the different 802.11 standards:

802.11
This was the first standard and contains many parts that are reused in the other 802.11 standards. WEP and the frame format are two such examples.
The standard defines three different ways to transmit data with a speed of 2 Mbit/s. Two ways to use the 2.4 GHz band and one way to use Infrared light.

802.11a
Introduces a new physical layer, which is OFDM (orthogonal frequency division multiplexing) at the 5GHz band. The data speed is between 6 and 54 Mb/s, which is higher than the data speed for 802.11b.

802.11b
802.11b is today the most used 802.11 standard. 802.11b is based on the DSSS (Direct Sequence Spread Spectrum) part off the IEEE 802.11 standard but the maximum speed is increased from 2Mbit/s to 11 Mbit/s.

The maximum allowed transmission power is different in different parts of the world, but in Europe it is 100 mW. Transmission power must at least be 1 mW.

802.11c
Handles bridging, e.g. it is needed when you construct an AP.

802.11d
Handles the different regulations in different parts of the world. How the radio transmitter should be constructed to be allowed to use in Europe or in USA. It also handles the problem with countries where the 2.4GHz band is not free to use.

802.11e
QOS (Quality of service) i.e. it tries to solve problems like which packet should be sent first and how much bandwidth each client should be allowed to use. This is important, because some services like IP-telephony require quite small bandwidth but also short delay.

802.11f
Handles communication between different AP. This is mainly required for roaming.

802.11g
Will increase the bandwidth useable in the 2.4 GHz band to 54Mb/s. It is probably going to use OFDM instead of DSSS. This means that existing NICs (network interface card) will probably not be possible to upgrade to the new standard. 802.11b cards will however be possible to use in 802.11g networks using the old speed.

802.11h
Modifications of the 802.11a standard, that better will suit the regulations that exist in Europe.

802.11i
Increased security compared to today’s standard. Will probably use AES for encryption and contain algorithms for key distribution.

5.3 OSI model

The OSI (Open Systems Interconnect) is a theoretical model, which describes how two computers communicate over a network. The OSI model is a theoretical model and is not used exactly as it is, however many real network systems has a structure that is similar to the OSI model. Therefore it is always good to have some basic knowledge about the model when you want to understand a new protocol.

5.3.1 Layers

The model is a hierarchical model, where each layer sends data to the same layer on the opposite computer. The hierarchical model is known as a stack, where each layer only has direct communication with the layers directly above and under.

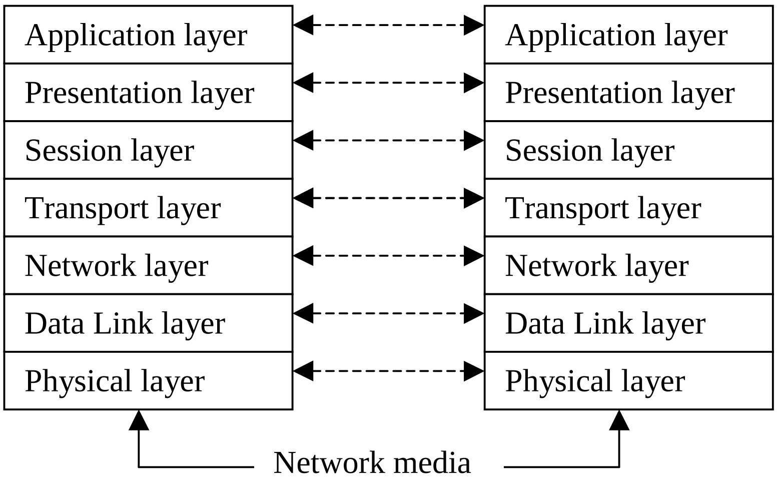

Figure 2. Layers in the OSI model

The purpose of this layer model is that a layer easily can be modified without the need of modifying any other layers. This is very useful, because you often want to change some of the layers. Examples when this is needed are when you want to use different protocols using the same medium, or using one protocol over different media.

Physical layer

The physical layer is responsible for translating the bitstream to the physical requirements for the network.

Data link layer

The main purpose for this layer is to verify that packets are received without errors and to handle the information in packets. This means that it has to put data being sent into packets and assemble received packets, sometimes fragmented into datagrams.

Network Layer

This layer handles routing, i.e. how the information shall be able to reach the receiver. One part of this work is to add information about the source and destination of the information. One example of a protocol that works in this layer is IP, which is used to route data over the Internet.

Transport layer

The main purpose of this layer is to fragment the information that is being sent into packets with a suitable size. It is also responsible for error recovery and for assembling received fragments in the correct order.

Session layer
This layer manages sessions between the communicating applications. This could be starting and ending the connection and keeping track of whose turn it is to send if the communication link is “Half duplex”.

Presentation layer
This layer works with the representation of the information. For example if the information should be sent compressed or encrypted or if any other kind of representation should be used.

Application layer
This layer makes sure that the resources needed to transmit the information are available. It is also responsible for translating the local information to the used protocol. This is needed because we sometimes want to run a server e.g. a webserver on one OS and the client on a different OS. One example of a protocol that uses this layer is FTP.

5.4 802.11 in the OSI model

The figure below shows how some of the 802 standards fit into the OSI model. As we can see the OSI model does not exactly fit with the 802.11 standard.

<table>
<tr><td colspan="3">802.2 Logical link control</td><td rowspan="3">DATA
LINK
LAYER</td></tr>
<tr><td colspan="3">802.1 Bridging</td></tr>
<tr><td rowspan="2">802.3
MAC
PHY</td><td rowspan="2">802.4
MAC
PHY</td><td rowspan="2">802.11
MAC
PHY</td></tr>
<tr><td>PHYSICAL
LAYER</td></tr>
</table>

Figure 3. 802 standards in the OSI model

The 802.11 standard covers both the Physical Layer and the lower part of the Data Link Layer, called the “MAC Layer”. The upper part of Data Link Layer is called “Logical Link Layer”. 802.11 covers a part of the Data Link Layer because the Physical Layer is only responsible for the translation to/from a physical signal and not for operations such as retransmission and error detection.

5.5 Hidden node problem

The hidden node problem is important to be aware of when we are discussing radio-based networks. This problem appears because radio transmitters have a limited range. Figure 4 shows an example where B is able to communicate with both A and C, but A and C are not able to communicate directly with each other. This causes problems because C is not able to know when A is transmitting and they can therefore start to transmit at the same time causing collisions. The problem is not even symmetric, it is not even certain you are able to send messages to all other stations that you can hear. E.g. A could increase the transmit power. This will make it possible for C to hear A but will C will still not be able to transmit to A.

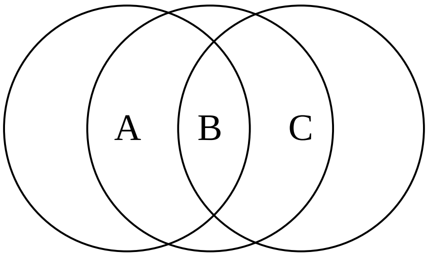

Figure 4. Example of the hidden node problem

5.6 Range

The range for a network using 802.11b is very dependent on the environment and it is therefore not possible to say much about it. This table from a datasheet for a NIC from Orinoco, see [9], gives some information about the range. The range given in the table is similar to the range seen from other sources.

Range	11Mbit/s	5.5Mbit/s	2Mbit/s	1Mbit/s
Open	160m	270m	400m	550m
Semi-open	50m	70m	90m	115m
Closed	25m	35m	40m	50m
Receiver sensitivity	-82dBm	-87dBm	-91dBm	-94dBm

Open is when it is free sight between the transmitter and the receiver and closed could be an office. Other environments are somewhere between.

5.7 Modulation techniques

A radio based communication system might want to use a wide frequency band for the transmissions for a couple of reasons. The most important are:

- To avoid disturbed frequencies
- Because of regulations for using the frequency. Many frequency bands have limitations for maximum allowed transmission power and bandwidth of a signal although the band does not require license to be used.

The 802.11 standard utilizes three different methods to make it possible to use a wide frequency band.

- FHSS Frequency Hopping Spread Spectrum.
- DSSS Direct Sequence Spread Spectrum.
- OFDM Orthogonal Frequency Division Multiplexing.

The main benefit with FHSS is that it jumps between different frequencies. This makes it possible to use the network although one frequency is jammed. Everything sent on the jammed frequency will be lost but it also uses many other frequencies and these will work. If DSSS or OFDM is used on this jammed frequency, communication will be impossible.

The choice between these techniques is mainly a tradeoff between transfer rate and how complex hardware and power consumption we can afford. OFDM gives highest transfer rate but has also most complicated hardware and highest power consumption. FHSS has the simplest hardware and smallest power consumption. The table shows some examples of where the different techniques are used and which transfer rate they offer.

Technique	Maximum transfer rate	Used in
FHSS	2 Mbit/s	IEEE 802.11
DSSS	11 Mbit/s	802.11b
OFDM	54 Mbit/s	802.11a and 802.11g

6 How 802.11 works

This chapter describes more detailed how 802.11 and especially how 802.11b works. 802.11b is as we have seen an improvement of the IEEE 802.11 standard.

6.1 IEEE 802.11

The 802.11 standard works as a frame that the other 802.11 standards modify to suit their specific needs. 802.11 defines for instance how what the different packets should look like, one way to improve the security by using cryptography called WEP (wired equivalence privacy) and three different transport mediums. The transport mediums that 802.11 defines are:

- IR (Infrared) probably never used in a product.
- DSSS (Direct sequence spread spectrum) 2.4 GHz
- FHSS (Frequency hoping spread spectrum) 2.4 GHz

The transfer rate is either 1 or 2 Mbit/s for all mediums.

6.2 Frame format

Everything that is sent over the network is sent in packets, see [6] for details. There exist three different main types of packets that are sent:

- Management
  This is for management functions like connecting and probing for networks.
- Data
  This is used for transmitting data in the network.
- Control
  This is used for controlling when and how packets are sent, e.g. to power save and to send ACK (acknowledge) for received packets.

The different packet types all look similar with a frame header first and then the actual data and finally a CRC to ensure that data is correct. The Frame Header is different between different packet types because all fields are not always needed. The data packet looks like this:

Octets:	30	0-2312	4
Type:	Frame Header	Frame Body	Crc

Frame Header is the header for the 802.11 packet. The header contains information like the sender, receiver and type of packet.
Frame Body is the actual data that is being sent in the packet.

Crc is a checksum (CRC32) for the packet to verify that it is correctly transmitted.

When we are using WEP to encrypt data packets the frame body also contains information about the encryption. Then the Frame Body looks like this:

Octets:	3	1	>=1	4
Type:	IV	Key ID 2 bits	Data	ICV

IV is the IV used for encryption.
Key ID is the id of the key used for encryption. It is possible to use up to four different keys.
Data is the data sent.
ICV is a checksum (CRC32) for the data to make it harder to modify packets.

The Data and ICV fields are encrypted when we are using WEP, all other fields are sent in plain text.

6.3 Power consumption

802.11 implements one algorithm for reducing the power consumption. This chapter describes how the algorithm works in infrastructure mode. The method allows a client to sleep a specified time. After that time the client wakes up and asks its AP if it has any new packet. During the time the client sleeps, the AP is responsible for caching packets that should be sent to the sleeping unit. There is also a similar system that is used in Ad-Hoc mode but it is somewhat more complicated.

Another possibility to reduce the power consumption would be to reduce the transmission power. To use this efficient it requires feedback from the receiver. Without this feedback, it is impossible to know if the received signal is strong enough. Without this knowledge it is therefore not possible to modify the transmission power to the correct level. Unfortunately the standard does not contain any such method and this is therefore not a useful method to save power.

6.4 Ad hoc compared to Infrastructure

802.11 can work in two different link-modes, either in “Ad hoc” or in “Infrastructure” mode.
Infrastructure mode is when the NIC always communicates with a base station. That base station is then responsible for passing the message forward to the receiver. If the receiver is in the same cell, the base station just retransmits the message. If the receiver is in another cell or network,

the base station transmits the message to the base station in that cell or to the other network.

Ad hoc mode is when two or more clients are able to directly communicate with each other.
Ad hoc is cheapest considering the needed resources because it does not need any base stations. This makes it possible for computers to create a network without the planning needed with in Infrastructure mode.

The conclusion from this is that “Infrastructure mode” is most useful for larger networks, i.e. networks where all participants can not hear each other. In such cases “Infrastructure mode” supports functions like routing to the correct cell and sometimes even roaming.

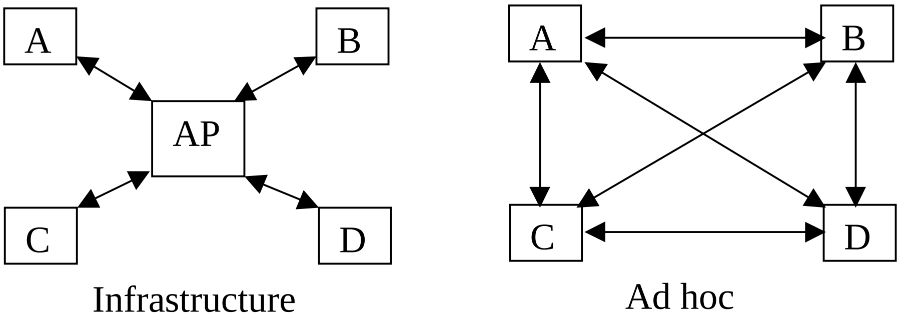

Figure 5. Principle of Infrastructure and Ad hoc mode

6.5 802.11 architecture

When the network is used in “Infrastructure” mode it is possible to connect different AP:s together in a network called DS (distribution system), see [6] for details. It is also possible to connect the DS to a custom 802.x network e.g. Ethernet. This connection is done through a “Portal”.

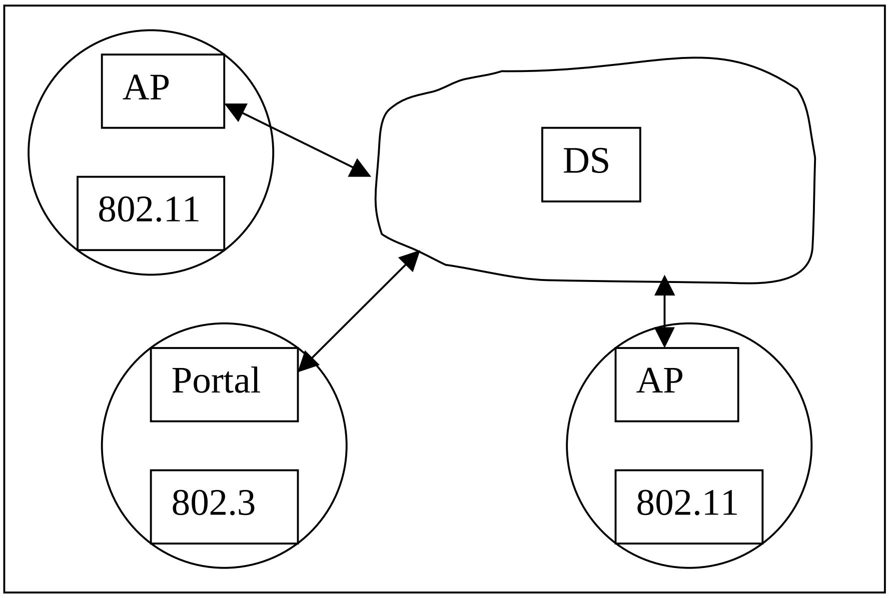

Figure 6. Description on the 802.11 architecture

6.6 RTS/CTS and Fragmentation

The 802.11 standard contains two functions that are useful for increasing the throughput in the network. The first function is the request to send/clear to send (RTS/CTS) function. The second function is a function that makes it possible to fragment large packets i.e. divide a large packet into smaller ones.

When a NIC wants to send a large packet, it has the possibility to send a RTS to its AP. The AP will then answer with a CTS packet, which means that a timeslot is reserved for the NIC where it shall send the large packet hopefully without collisions. This way the number of collisions with large packets will be smaller for the cost of the extra overhead given by the CTS/RTS packets. To maximize the performance of a network it is important to use the best threshold for the packet size when RTS/CTS should be used.

The fragmentation function aims at the same problem but with a simpler solution. When the NIC has to send a large packet it instead divides it into two or more small packets and sends these packets instead. Of course it now has to send more information because it also has to send information about how to reconstruct the original packet but the advantage is, that it is better to send smaller packets than larger ones. This is better because the probability to get a collision is larger for a large packet because it takes longer time to transmit, and the cost to retransmit a packet is larger for a large packet.

7 Ad hoc networks

Ad hoc networks offer many interesting possibilities but they also give many problems. This chapter contains information about the hardest problem to solve when we are using Ad hoc networks, the routing problem.

The main benefit with Ad hoc networks is that they do not require any hardware except the network cards. This gives a possibility to embed wireless network cards into many different products and then let them communicate directly with each other. Another possibility is to use Ad hoc networks in computers that are being used where no infrastructure is available. This gives an opportunity for people cleaning up after disasters or military personal in field to communicate with each other using a relatively fast network.

The topology of the network changes all the time. This gives new opportunities but it mainly introduces new problems. The opportunity is that you are allowed to take the equipment and move it to a new position, i.e. the equipment is not necessary stationary like a base station for cellular phones. The largest problem is that we very often want to be able to communicate with other users that are out of the range for your own transmitter, i.e. we need some kind of routing.

7.1 Routing

A very hard problem to solve is how we shall route messages between devices that are too far away from each other. In a traditional network this is done using the address of the receiver but this is not possible for Ad hoc networks, because the devices do not have fixed positions.

The routing protocol has to solve two different problems:

- Find a cheap allowed path between the devices.
- Protect against malicious devices.

7.1.1 Cheap path

The problem to find a cheap path between the devices is a problem that is quite easy to solve if we do not have strict requirements for the algorithm. This could e.g. be solved with “Dijkstras shortest path” algorithm, see [12] for examples of how to solve shortest path problems. The difficulties that could appear is that the topology of the network is not stable, new connections can appear and existing connections can break up. We also might have to reduce the traffic caused by the routing protocol. The path finding packet has to reach all members of the network because we can not know which one that has the best connection to the receiver. In a large network this means that many devices will get the packet. Many devices will send packets to all other devices, which might require a method for reducing the traffic.

It is not always necessary to find the cheapest path. Sometimes it could be better to just find a cheap enough path. This could be useful for two different situations, the cheapest path could be more exposed for eavesdropping or it could be used to reduce the traffic caused by the routing protocol.

7.1.2 Malicious devices

The biggest problem with a routing protocol is how to handle the security. This problem is mainly caused by the need of delegating much of the decisions to the different devices because they are the only ones that really know what the network looks like, who they can communicate with. This could e.g. make it possible for a malicious device to tell others that they have a very good connection to the receiver or to say that the connection between two other devices is broken. The first attack could cause all traffic to pass the malicious device, causing higher delays and lower throughput and is also perfect to eavesdrop the network. The second attack will also cause higher delays and lower throughput, because a useful link won’t be used. In an extreme case this could prohibit all traffic between different parts of the network if this was the only connection between them.

One way to reduce this danger is to divide the devices into different groups according to how secure they are. All devices in one group could share a secret key that prevents lower trusted devices from receiving and modifying messages attempted for higher groups. A solution of this kind would make it possible to specify how secret a message is. If the message is very secret only the most trusted group is allowed to relay it. If the message is not secret then anyone should be allowed to relay it. These kinds of solutions make it possible to increase the security, but it will also decrease the performance because all devices are not used.

7.2 Summary

As we saw in the chapter about routing in Ad hoc networks there are a lot of different problems to solve. Some of the problems do not exist in small networks, but in a huge network they can cause a lot of trouble. One example of such a problem is to reduce the traffic caused by the routing protocol. In a small network it is possible to let a message to be sent to all other stations and in this way find the shortest route to the target, but in a large network this will cause more traffic than we can accept.

The conclusion is therefore that we will need different solutions for different applications. In a small network we might want to cause some extra messages just to keep the route finder simple. We might also be able to simplify the solution in situations where maximum security is not always needed.

Two different military scenarios where Wireless LAN could be useful are first to let the staff communicate with each other over a high-speed link. The second is to equip each soldier in a platoon with a GPS and a camera that directly sends information back to the head quarter.
These scenarios show that the requirements for Ad hoc networks can be very different. The first scenario requires maximum security because no information is allowed to leak out. In the second scenario we could reduce the security if it makes the solution smaller and easier to implement. Another difference is that the equipment in the second has to be smaller because everything has to be worn. Smaller equipment has shorter range and therefore the nodes in the Ad hoc network must be used more efficiently.

8 Security in 802.11

This chapter contains information about the security methods included in the 802.11 standard. This chapter introduces the methods, and chapter 9 contains an evaluation about how well they actually work.

802.11 usually uses radio for its communication, which makes it vulnerable for eavesdropping. Therefore the standard has, in contrast to many other network standards, support for encryption.

The security in 802.11 consists of these different parts:

- WEP (Wired equivalent privacy)
- SSID (Service set identifier)
- Shared key or Open authentication
- MAC address filtering

8.1 WEP

The method for encryption that is included in 802.11 is called WEP. As the name WEP (Wired equivalent privacy) says, the intention of the encryption is not to give a completely secure connection but instead give a wired equivalent security.

The encryption in WEP uses a stream cipher called RC4 with a 24 bit IV, see Appendix B: RC4 for details. The key is 40 bits in the standard but a modified version with a 104-bit key is today common in NICs.

8.2 SSID

The SSID acts as a name for the network, which makes it possible for different networks to use the same radio channel. The name is needed when you want to connect to the network and this prevents unintentional use of other networks.

8.3 Authentication

When the NIC connects to an AP the NIC has to pass an authentication to be allowed to send any packets. The authentication is either “Open” which means that everyone will pass or it can be “Shared key” which is a “Challenge response” authentication. Challenge response authentication has the benefit that it does not send the key in plain text.

Authentication using shared key follows this scheme:

1. Requester: Send an authentication request
2. Responder: Send the authentication challenge
3. Requester: Send the challenge encrypted
4. Responder: Send the result, successful or not successful

8.4 MAC address filtering

Filtering by MAC address is not a part of the 802.11 standard but I will mention it in this chapter because it is a possible method for increasing the security of the network without the need of modifying the client hardware or software.

MAC address filtering uses the fact that all NICs are given a unique id, the MAC address, when they are manufactured. If we keep a database that holds all MAC-addresses that are allowed to use the network then we can filter out all traffic that comes from not trusted NICs.

9 Weaknesses in 802.11 security

When we read about all the different security methods that are possible to use, the network seems quite secure. This chapter contains information about the flaws that have been found in the security. For more information about the flaws see [10].

9.1 WEP

Due to the way WEP is used there exist some different weaknesses and attacks against it.

9.1.1 Reuse of IV

Because RC4 is a stream cipher, it is very unsuitable to reuse an IV. The reason for that is the simple relation between plain text and cipher text, **Cipher text = Plain text ⊕ Key stream**. This mean that if the eavesdropper has two or more messages with the same IV he might be able to get information about the plain text. To make the situation even worse the standard does not contain any information about how the IV:s should be used, the standard does not even say that the IV should be changed between different packets.

9.1.2 Message Modification

The unlucky combination of RC4 as encryption algorithm and CRC32 for integrity check makes it possible to modify a transmitted packet.
The linear property of RC4 ensures that if bit n in the cipher text is inverted it is guaranteed that bit n in the clear text also is inverted. This property makes it possible to create controlled modifications. Because it only is possible to invert bits in the plaintext we must know the corresponding plaintext if we want to set a specific value. This prevents us from modifying the whole message but some parts are often easy to guess. Parts of the packet header are one example of data that often are easy to guess and therefore possible to attack.
The ICV (Integrity check value) can also be fooled to accept the new message because of the, in some meaning, linear property of CRC32. This makes it possible to calculate how the new ICV differs from the old one without having to know the complete message.

9.1.3 Weak IV

One algorithm that uses, that some IVs leaks unnecessary much information about the key is described in [2]. A short explanation of the algorithm can be found in Appendix C: Weak IV. These IVs that give information about the used key are called Weak IVs. With this algorithm, it is possible to guess the value of a certain byte in the key, with a 5% possibility to guess right. 5% possibility does not sound so much but it is much higher than the 1/256 chance for ordinary bytes. It is also possible to collect a number of these bytes to improve the possibility to guess right. With this method it is possible to break the key in linear time compared to exponential for brute force.

9.1.4 Brute force

A brute force attack is an attack that tries every possible key until it finds the correct key. If the length of the key is n bits there is 2^n different keys and in average ½ of them has to be tested before the correct key is found. This attack is possible to use on small amounts of data, actually a single packet could be enough. It might be hard to know when the correct key is found and the time consumption quickly rises when the key length becommes longer.

The key length used in WEP is only 40 bits. When the key is so short, there is no problem to break it in days or maybe even hours using standard computers. The 104-bit key seems to be secure against this kind of attacks today.

9.2 SSID

Without the correct SSID it is not possible to send any data packets to the network and this prevents improper use of the network. This prevents unintentional use of the network but it hardly prevents someone who wants to abuse the network. The reason for that is that the SSID, according to the standard, periodically sent out in plain text in so-called “beacon frames”. This “feature” can often be disabled in APs. This violates the standard but does not seem to make any harm. The SSID is however still transmitted in plain text when someone connects to the network and this transmission can therefore be eavesdropped.

9.3 Authentication

The method for authentication in 802.11 can be abused. The reason for that is if someone eavesdrops step 2, the challenge, and step 3, the encrypted challenge, he can easily calculate the key stream for that specific IV. When the key stream is known it is just to reuse it for another login, this time by the abuser. Therefore, it is sometimes recommended not to use “Shared key authentication” because it does not actually increase the security.

Another attack that is possible due to problems with the authentication used is the Man-in-the-middle. This attack is possible because only the client and not the AP are identified. This makes it possible to set up a spoofed AP between the correct AP and the user. The user will then connect to the AP that has the strongest signal, which will be the spoofed AP because it is closer to the user. The spoofed AP becomes a perfect starting point for an attack because all traffic from the user will go through it. This makes it possible for the attacker to listen to all traffic from the user, and it is also perfect for modifying messages.

9.4 MAC address filtering

MAC address filtering gives a security that is quite similar to the security given by SSID. By eavesdropping valid packets and investigating valid MAC addresses this security mechanism can be broken. The benefit of this method compared to SSID is that the eavesdropped MAC address is only valid as long as the owner of the address does not log out from the network. This attack should also be possible to detect by the owner of the address either if the NIC discovers that someone else is using his address or if the NIC receives strange responses like ACKs for packets he has not sent.

Administration will probably become a quite costly problem when someone wants to use this. The reason for that is that the administrator has to keep a database that contains the MAC-address for all NIC:s that are in use. This means that he has to add the address as soon as someone gets a new card and remove the address when a card is not longer in use.

9.5 DoS attacks

This kind of attack tries to reduce the possibilities to use the network to a minimum to either prevent any useful traffic or delay it as much as possible. This can mainly be done in two different ways, either “filling” the network with garbage so no useful traffic can be sent, or by sending malicious packet to a client to use all its resources.

9.5.1 Radio jamming

All radio based communication systems are sensitive to radio jamming. An attacker can always send a signal on the same frequency as the network. If the signal is strong enough it will always interfere with the valid signal and cause some kind of bit fault. 802.11.b is very sensitive for this kind of attack. It is so sensitive because it always uses the same frequency and because its polite behavior when someone else uses the frequency. According to the standard a NIC has to first check if someone else is using the frequency and if someone else is using it, then the NIC is not allowed to send. This means that the attacker either can send a signal all the time that is strong enough to be heard by the others or it can start to send as soon as someone else is sending to cause a bit fault.

9.5.2 Malicious packets

Sending malicious “Management frame” packets could easily be done in 802.11. This could easily be done because management packets are not encrypted. Because of this anyone can send a “Management packet” faking any station on the network, even the AP. Malicious packets could either be used to attack one specific client or the whole network.
An attack against the whole network tries to fool all the participants that the network is busy. The RTS/CTS function could for example be used do this. The method for doing this is to send faked CTS packets, which says that the network is reserved for a specific NIC for a certain time period. By repeating this, the network will always seem to be busy without forcing the attacker to send much.

Attacks against a specific server either tries to keep it busy doing something that is not useful or spending a lot of power (Battery exhaustion). Both these attacks are mainly useful against small embedded applications e.g. cell phones rather than laptops because laptops usually have too powerful processors and spend so little power on the NIC that it hardly affects the performance (unless the laptop runs a vulnerable application).
The attack that tries to keep the processor busy sends packets that take lot of time to process. This could be e.g. IP connections which might require memory allocations or similar.
The “Battery exhaustion attack” sends packets that prevent the client from sleeping or even better requires some kind of response.

10 Maximizing security

If we however want to use 802.11 for transmissions of sensitive data, we must somehow increase the security. This can be done in two different ways. The first way is to decide that 802.11 is insecure and that all data that is being sent therefore has to be encrypted. The second way to increase the security is to use the security already existing in 802.11 as well as possible.

10.1 Enhanced encryption

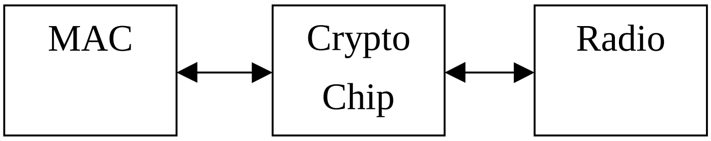

Figure 7. Block diagram for NIC with encryption

Using a new algorithm for encryption gives the possibilities to make a really secure system. One way to do this is to insert an encryption module between the MAC layer and the physical layer. This method is used in [4]. There are several reasons why this is a good point for encryption. The first reason is that this is a quite easy place to get the data that is being transmitted. This because the radio part and the MAC part of a network card very often are in different chips. The second reason is because all information that is being sent over the network will be encrypted, and not only the “Real data”. This makes it much harder for an attacker to know which units that are communicating because their MAC-addresses will also be encrypted.

The disadvantage with this solution is that it does not follow the standard any longer. One benefit however with this system is that the NIC looks almost like the same NIC without encryption. The only difference is that it must read the encryption key before it can be used.

10.2 Regularly change key

To change keys often is one method for increasing the security. This is mainly important for two different reasons.
The first reason is that the intruder needs quite a lot of traffic to be able to use the “Weak IV” method for breaking the key. Without a lot of traffic the possibility for reusing an IV also is lowered.
The second reason is that if someone breaks the key he will not be able to decrypt so much and hopefully, can not decrypt so many secret messages.

One problem with this is that the standard does not provide any kind of mechanism for distributing the keys. This means that it is necessary to install some kind of software that handles this problem. Another problem with this solution is that it will not be completely secure. Someone could still be able to eavesdrop enough packets to be able to break the key, even if the probability is much lower.

To be able to change keys frequently external programs are needed. One program that is able to do this is “Key hopping”. This program is able to generate new session keys from a seed at regular intervals. The program can for example generate a new key every 30 second. Another way to solve the problem is to use an authentication method that can distribute keys. One way to do this is to use 802.1x and EAP, see chapter 10.5

10.3 Avoid weak IV

Avoiding IV:s that are weak is probably the most important step for increasing the security if we want to use the existing standard. This is so important because the complexity of breaking the key using weak IV is linear compared to the brute force, which is exponential.
Unfortunately the producer of the NIC is the only one that actually can effect which IVs that are going to be used.

10.4 VPN

The mostly recommended method for creating secure wireless networks today is to use VPN. With this method the wireless network is seen as completely insecure. Instead, all communication between the wireless network and the secure network goes through an encrypted tunnel. The benefit with this solution is that we receive a high level of security, because the VPN solution is well tested and has high security. Another benefit is that the solution becomes very similar to VPN solutions for Internet.
The disadvantage with this solution is that it only protects the connection to/from the wireless network and not the network itself. This means that it might be possible for an intruder to connect to the network and attack other computers on the network if they are vulnerable, but he will not be able to reach anything outside the network.

10.5 802.1x and EAP

What we require from an authentication method useful for 802.11 is the following:

- Identify both user and server.
  Needed to prevent the “man-in-the-middle attack” which is a serious threat because it is quite easy to set up a malicious AP.
- Useful for key distribution.
  It must be easy to change keys rapidly.
- Secure enough for its applications.
- Simple enough.
  There is a tradeoff between security and simplicity. The algorithm has to be secure enough but it also has to be simple enough to be used in embedded applications.

One method proposed for authentication in 802.11 is 802.1x plus EAP.
802.1x is originally a standard used for “Port based access control”. It is however possible to use it in 802.11.
802.1x works as a frame for authentication where it is possible to use own methods, which are variants of EAP. The benefits for 802.1x compared to the current 802.11 is:

- Possible to use different and more secure authentication methods.
- Possible to identify both AP and client
- Possible to generate and distribute session keys.

Notice that these benefits require suitable EAP.

10.5.1 Authentication

Authentication with 802.1x follows these steps:

- Client is in the unauthorized state. This means that he is only allowed to send 802.1x messages.
- Client starts the authentication. Messages sent to AP are forwarded to the authentication server.
- Authentication between client and authentication server with AP as a bridge.
- If authentication succeeds, the client will enter state authorized which means that he is allowed to use the network.

There exist a couple of different EAP solutions and they all have pros and cons.

11 802.11i

The 802.11i standard is still under construction but it aims to solve many of the security problems the current standard suffer from. Because the standard is not yet finished it is not possible to exactly know what it is going to contain but people expect these parts to be the most important:

- TKIP, Temporal key integrity protocol
- AES, Advanced encryption standard
- Use of 802.1x and EAP, see chapter 10.5

11.1 TKIP

TKIP is an attempt to solve the problem with key distribution. TKIP is going to use RC4 as encryption algorithm as we are doing in 802.11 but the difference is that TKIP generates new keys very often. This makes it possible to change key often with a standardized method that prevents the eavesdropper from getting enough packets with the same key. This solution becomes similar to the solution “key hopping” but with two big differences. The first is that it will be standardized and the second is that the network driver will support it and therefore not require an external program.

11.2 AES

AES will probably be used for the encryption instead of RC4, which is used today. The main difference between RC4 and AES is that RC4 is a stream cipher and AES is a block cipher.
Block ciphers are much better suited for data protection than stream ciphers because they does not have the simple relation between cipher text and plain text, not(C(i))→not(P(i)), that stream ciphers suffers from. In block ciphers an inverted bit will instead influence the whole block in a random effect. Because of this is it much harder to modify messages.
The main disadvantage with AES is that requires much more CPU-power. That means that it will not be possible to just upgrade the firmware for a NIC, the NIC has to contain better hardware to be able to use AES. The second disadvantage is that it only is possible to encrypt data with length N*blocksize. Data with other lengths than that requires padding, which will make the packet slightly bigger, in average blocksize/2. Note that it is possible to create a stream cipher from a block cipher but then it will loose many of the advantages of using block ciphers.

12 Summary of security

This chapter contains a summary of the threats and methods for maximizing the security in 802.11.

As we have seen there exist many potential attacks against 802.11. The most serious attack is definitely the attack against weak IV, described in chapter 9.1.3. This attack gives an attacker the encryption keys. Another big problem is that there is no authentication of APs, see chapter 9.3. This attack requires more effort than breaking the keys but it is still possible to do especially for getting sensitive information.

It will be very interesting to see the security improvements that come with the 802.11i standard. Most of the security flaws that exist in 802.11 are caused by problems with the implementation of RC4. 802.11i is probably going to use AES instead and this will hopefully solve these security problems. Unfortunately, the existing NICs will not be possible to upgrade to the new standard and it is therefore still interesting to investigate the possibilities to increase the security.

My conclusion about the security is that it definitely is not good enough for transferring sensitive information. It could be secure enough to use for private persons because it takes some time to break, but private persons should definitely be aware of the security problems. The VPN solution seems to be a very attractive solution because it improves the security to a high level still using this cheap standard.

13 IEEE 802.11 Case Study

The following chapters in this report contain the results from the practical part of the thesis.

The study investigates the possibilities and practical problems with transferring data using 802.11b. The study was done on a platform with limited resources, limited amount of memory and a quite slow processor. This was done because it was interesting to see if the standard is useful in embedded products.

The demonstration was supposed to show how 802.11 could become more secure. This was however not done because the initial phase of the programming, mainly focused on the PC Card took too much time. Another reason why this problem was ignored was that the packet format appeared to be more or less the same as ordinary Ethernet (IEEE 802.3) packets. The problem with making 802.11 communication secure would therefore become the same problem as making Ethernet communication secure. This problem did not seem to be very interesting and was not given priority.

13.1 Requirements

For the case study the following main requirements were set up:

1. Demonstrate how to use 802.11
2. Measure performance
3. Show the usability

From these requirements the following requirements were derived:

1. The 802.11 NIC should have an existing open source driver. This was needed to make it possible to write the driver for the new platform
2. Communication should use TCP/IP. This was needed because we wanted to test the performance without having too much effort generating the traffic.
3. A second platform was needed to communicate with. This platform should preferable be a PC. This was preferable because writing programs for it should not be a problem and its performance should be high because we wanted to test the performance of the embedded system and not the PC.

13.2 Prestudy

The main task in the prestudy was to investigate what kind of hardware that was available on the market. The expectation was to find some kind of evaluation kit that could easily be connected to the embedded system. Unfortunately this appeared to be impossible, all 802.11 NICs that were available to buy were designed for computers with either PCI, PC Card or USB connection.

The choice was to buy 2 NICs from 3COM. A NIC called 3CRSHPW696 for the embedded system and a NIC called 3CRDW696 for the PC.

For the embedded system there were many equivalent cards and a couple of others could therefore also been used. The benefits for these cards were:

- Open source driver could be found, see [1].
  The driver was originally written by Atmel. Atmel is the producer of the used MAC chip and the driver was therefore fairly reliable.
- The NIC had been on the market for a while. This was important because Atmel did no longer support the driver and it was therefore not sure that the latest cards would be supported.
- The driver was logically written with different files for different tasks.

13.2.1 Hardware

While I decided which NICs that should be used I also decided which platform to use. Sectra already had developed a platform that supported PC Card. This platform was therefore very suitable for me to use because it would then be easy for me to get information about the hardware.

The following parts of the platform were interesting for me:

- ARM-processor running at 39 MHz
- PC Card interface
- Serial interface running at 115200 BPS.
- SRAM and Flash memory

This platform was using an RTOS called AMX, see [5]. This operating system was a good choice for the application because this company is also selling an IP-stack called KwikNet that is very easy to use together with AMX.

13.2.2 PC Card interface

The most interesting part for me was the PC Card interface and I am therefore going to describe it in more detail.

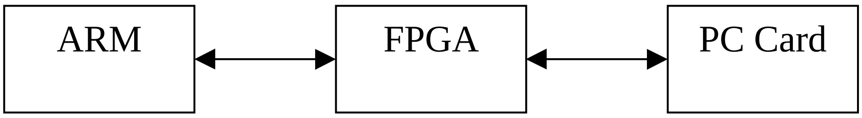

Figure 8. Block diagram for PC Card

The communication between the ARM and the PC Card is done through a FPGA. This FPGA is responsible for creating the different control signal that are required. The FPGA is connected to a bus on the ARM called the “External bus”. This bus has several “Chip Select” signals and configuration registers that make it possible to connect different types of units to the bus like SRAM, Flash and even LCD-displays. The following configuration registers in the ARM were used for the communication with the FPGA:

- Type of access. Controls if we are doing 8 or 16 bit accesses.
- Wait states. Controls how long time the access will take. This can be done in two different ways. Either each access takes a specific time or is it possible to let the external hardware indicate when the transfer is completed. The first method when all accesses take the same time is used in this project.
- GPIO register. The ARM has several pins that can work either as input or output. Each pin is easy to control through a register in the ARM. One of these pins is used to tell the FPGA whether an access is to its configuration register or to the PC Card.

In the FPGA a couple of configuration registers are found that controls which type of access that is done. The ARM also has some configuration registers that are involved in the process to do correct accesses:

In the PC Card a couple of different access types are defined but in this project two of them are going to be used. These types are:

- CIS-access. CIS is an acronym for Card Information Structure and is the system used for getting information about the card used and for configuring it.
- IO-access. When the card is configured 8 and 16 bit IO-accesses are used to communicate with the card.

To change from one type of access to another the following algorithm was used.

1. Set GPIO to indicate access to configuration of the FPGA.
2. Set the ARM to do 8-bit accesses.
3. Set the FPGA to the correct type of access, CIS or IO and 8 or 16 bit.
4. Clear GPIO to indicate access to PC Card.
5. Set the ARM to 16-bit access if necessary.

All these steps also have to be thread safe, either by disabling interrupts during the process or by restoring the original state to prevent strange results.

13.3 Implementation

The main purpose of this step was to get the network driver and the IP-stack to work under AMX. The network driver is a modified version of the Linux driver for the card, see [1] for the Linux driver. The following were the reasons why I chose to reuse this driver instead of writing a new designed for AMX:

- Faster development because this kind of drivers are quite complicated.
- Hard to get the specifications for the card.
- Investigate how hard it is to use a Linux driver in another project.
- Possible to fairly easy modify the driver to work with similar NICs. This is probably the most interesting argument because new and better NICs are released quite often. Therefore it is very good with the possibility to easy support them.

The driver appeared to be very well suited for reusing and this was therefore a good choice. The driver was divided into separate files that each was responsible for different functions. Because of this it was easy to find those files that were responsible for system specific functions and those which were the core of the driver. The most important functions that needed to be rewritten were:

- Communication with the PC Card. This function needed to be completely rewritten because it looks completely different. This also caused most problems because it needed synchronization between different threads.
- Communication with the operating system. Initialization of the driver is done completely different in Linux compared to KWIKNET and therefore needed a complete rewrite. However I reused some parts of it, e.g. the part that handles IOCTL:s because this was working well.

- Slight modification of the interrupt handling. The interrupt handling had to be done in a separate thread instead of the ordinary interrupt routine. This was needed because the interrupt routine takes too long time. Running with interrupts disabled for too long time is not good because it prevents other interrupts to be served. Especially the RX interrupt from UART has to be served quickly.
  Moving the interrupt serving to a separate thread was easily done, just letting the interrupt routine trigger the new thread that runs the old interrupt routine. However this problem made the PC Card routines more complicated.

However the reuse of the code also had disadvantages. The main disadvantage is that the driver is more complicated than necessary. The embedded system has limitations that not necessarily are found in an ordinary PC. An example of this is that the embedded system used can only handle one PC Card and it is not possible to remove or insert the card when the system is started. This makes for example many parameters passed in function calls unnecessary because we already know which card that is operating. Such problems use more memory and processor power than needed.

13.4 Verification/Evaluation

The purpose of this phase was to write a program that verified the function of the network driver and to test the performance. The program was mainly written to test the performance of the network, but it also verifies that the network driver and IP-stack are working. It uses TCP and just responds with the same data sent to it, see Appendix A.

13.4.1 Performance test

To see how good the performance was on the ARM system, the test was also done using the NIC on a computer running Linux with the original driver. The time it took to send 100 packets of the desireb packet size was measured. This procedure was done a few times and extreme values was discarded. From the remaining measurements was the average speed calculated.

Packet Size (Bytes)	Speed(Kbytes/sec) ARM	Speed(Kbytes/sec) PC
100	13	35
200	23	60
400	40	100
600	45	122
800	58	145
1000	75	158

Table 1. Performance for ARM and PC system

To note about these values is that the values for the PC is almost the same every time the program runs, but the result differs quite much for the ARM values. This is probably caused by a remaining bug in hardware or software.

Another thing to note for the results is that for the ARM, the speed is quite linear to the packet size. This indicates that the overhead for each packet influences the performance more than the time it takes to copy data. If the limitation was to copy data, we could expect that the speed would be more or less the same for big packets.

Another interesting measurement would be to measure the time it takes from when the ARM system receives a packet until it sends a response. This would remove the influence from the PC. It would also make it possible to estimate the delay in the NIC, i.e. how long time it takes between the NIC receives a packet until it is actually sent in the air.

To do this kind of measurement another computer would have been needed and they were therefore not done. If another computer had been available it would be connected to the same network. Then would a program like TCP-dump show all the traffic and make it possible to measure the time between different packets.

One problem with this limited performance is that the system probably is easy to oppose for a DoS attack. Just sending as much packets as possible through the Wlan to the ARM could probably do this. Due to the limited performance by the ARM and PC Card it would probably not be possible to read the packets from the PC Card as quickly as they are received.

13.5 802.11 in embedded products

The main problem with an embedded product that uses 802.11 is how to get information about the wireless network that is going to be used. To be able to connect to a network, you need encryption keys and the name of the network and eventually also which frequency to use. The main problem is how to get this information into the product. The name and the frequency could be programmed at manufacturing but the keys have to be possible to change.

13.6 Increasing performance

The accesses to the PC Card are convenient when we are only using one type of access. The problem with the PC Card module is when we start to do both 8 and 16-bit accesses. Unfortunately the NIC driver utilizes both kinds of accesses to improve the speed when we are transferring large packets. This however gives troubles to synchronize the accesses. One good improvement would therefore be to get rid of these changes. Either if it is possible to see from the FPGA if the access is 8 or 16 bit, or to use different memory addresses.

The next problem with the FPGA is that each access takes the same amount of time. This is set to a value that hopefully is larger then the longest time an access can take. A better solution would be to let the FPGA tell the ARM when the transfer is completed. This way an access does not take longer time than necessary. A further improvement of this would be to let the FPGA cache writings. Then the ARM immediately could continue when the FPGA completes the writing.

Another interesting possibility would be to investigate the possibilities with the built-in DMA-controller. This DMA-controller can work in two modes, IO and memory. Memory mode is used when we want to move a memory block from one position to another. IO mode is used when we want to transfer data to/from one specific address from/to a FIFO. Using DMA would hardly make the driver faster but it would make it possible for the processor to do something else during the transfer.

14 Conclusions

Some weaknesses in the 802.11 standard have been pointed out. These show that the standard is not secure to use. Therefore extra encryption and key exchange are needed to make it possible to transmit sensitive information.

The choice to use the Linux driver was a good choice. Writing a new driver would have taken too long time and would probably introduce new bugs. Therefore I can recommend looking at existing open source drivers when a driver for a new platform is needed.

Using an IP-stack in an embedded application both gives possibilities and problems. The good side is that it makes it very easy to send and receive data. The bad side is that it is large and complex. This makes it hard to actually know what is happening especially when something is wrong.

802.11 seems to be a standard that definitely is possible to use in an embedded system. However, the CPU is too slow to fully utilize the speed that the standard offers. Therefore the application is probably possible to expose for DoS attacks just by flooding packets to it.

15 References

[1] "Home of the OpenSource Linux Driver for Atmel AT76C503A based Wireless Devices", http://atmelwlandriver.sourceforge.net

[2] Fluhrer S., Mantin I., Shamir A. "Weaknesses in the Key Scheduling Algorithm of RC4", 2000

[3] "Nato C3 agency" http://www.nc3a.nato.int/mwlan.html

[4] "Secnet 11 – Secure Wireless Local Area Network" http://www.secnet11.harris.com

[5] Homepage for the producer of Amx and KwikNet http://www.kadak.com

[6] IEEE, "Part 11: Wireless LAN Media access controller (MAC) and Physical layer (PHY) specification", http://grouper.ieee.org/groups/802/11/index.html, 1999

[7] "Get IEEE 802", http://standards.ieee.org/getieee802/

[8] "wifi-alliance index", http://www.weca.net

[9] Datasheet for "Orinocco World PC Card" http://www.orinocowireless.com/products/all/orinoco/docs/ds/PC-card.pdf

[10] Karygiannis T., Owens L. "Wireless Network Security 802.11,Bluetooth and HandHeld Devices", "NIST special publication 800-48" http://cswww.ncsl.nist.gov/ publications/nistpubs/800-48/NIST_SP_800-48.pdf

[11] Stallings, W. "Cryptography and Network Security Principels and Practise". Prentise-Hall, 1999

[12] Holmberg, K. "Flöden i nätverk och kombinatorisk optimering". Matematiska institutionen Linköpings universitet, 1993

All internet references available online February 2003

16 Appendix A: Test program

This code shows the test program. The program first sends a packet with a specified packet size and then receives the same packet. The code only shows the part that sends and receives data. Another function measures the time taken and calculates the transfer speed.

Test on the client side:

```
void test1(void)
{
        int i,j;
        int recieved;
        printf("\n");
        for(i=0;i<nrpkt;i++)
        {
                printf("\r%d",i);
                memset(buff,i,pktsize);
                if(send(outsocket,buff,pktsize,0)!=pktsize)
                        printf("send error\n");
                recieved=0;
                while(recieved<pktsize)
                {
                        recieved+=recv(outsocket,&in[recieved],
                                                pktsize-recieved,0);
                }
                if(memcmp(in,buff,pktsize)!=0)
                        printf("recv != send %d\n",i);
        }
}
```

Interesting part of the test program on the server side:

```
FD_ZERO(&socks);
FD_SET(service_sock,&socks);
status=1;
while(status!=0)
{
        status=recv(service_sock,buff,2000,MSG_DONTWAIT);
        if(status>0)
        send(service_sock,buff,status,0);
        if(status<0)
        select(-1,&socks,NULL,NULL,NULL);
}
```

17 Appendix B: RC4

RC4 is a stream cipher developed 1987 by Ronald Rivest and was kept as a trade secret by RSA Data Security. It was anonymously posted on the Internet September 9 1994.

RC4 uses a key with a length of 1 to 256 bytes to initialize the 256 bytes state table. The algorithm is very useful for embedded applications because it requires few and especially very simple operations. All arithmetic is done “modulo 256” which makes it easy to use in 8 bit processors. The memory requirement is 256 bytes for the state table and two bytes for variables.

Encryption and decryption uses the same algorithm i.e. if you encrypt something two times you will end up with the plain text. Encryption consists of two stages. The first is to initialize the state machine utilizing the key. The second step is to use the state machine to generate a bit stream which is xored with the plaintext. When we discuss the security in 802.11 the simplified model in Figure 9 is often good enough because we seldom need any details about how RC4 works.

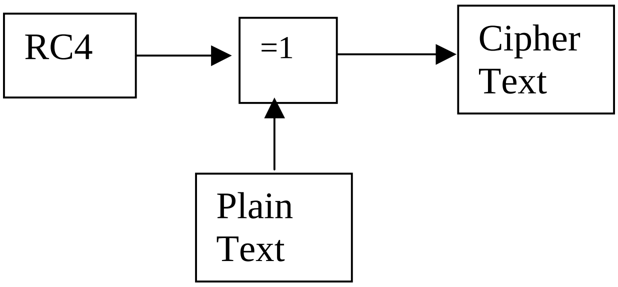

Figure 9. Simplified picture of RC4

```
Variable i,j
Variable Sbox[256]
Init state table:
For i=0 to 255
        Sbox[i] = i
j=0
For i=0 to 255
        j = ( j + Sbox[i] + Key[ i % keylen ] ) % 256;
        swap(Sbox[i] , Sbox[j])
i=0
j=0
```

encrypt:

```
    i = (i+1) % 256;
    j = (j+Sbox[i]) % 256;
    swap(Sbox[i] , Sbox[j])
    t = ( Sbox[i] + Sbox[j] ) % 256;
    K = Sbox[t];
    output K xor c;
```

As we can see in Figure 9, RC4 generates a byte stream that is xored with the plain text. As we are using the same key for several different plaintexts, we do not want to get exactly the same byte stream for every encryption. Therefore we are using an initialization vector (IV) that is mixed with the key. In WEP the IV precedes the encryption key when we generate the key used in RC4. I.e. RC4 key=(IV,WEP key).

18 Appendix C: Weak IV

This chapter contains information about how the “Weak IV” attack works. A more detailed description of the algorithm can be found in [2]. Good knowledge about the RC4 algorithm is needed to be able understand this chapter. Information about RC4 can be found in Appendix B: RC4.

The attack uses that some IV makes it possible calculate the most probable symbol in the RC4 key stream. The first output in the keystream will be Z when the Sbox looks like this:

Position:	1	X	X+Y
Value:	X	Y	Z

As we can see the attack requires that we know three specific values in the Sbox to be able to predict the first output.

This is an example of how the attack can work when we attack a cipher like WEP that has the IV first in the RC4 key. It is important to remember that there exist more IV:s that are vulnerable than those used in this example. N can be almost any value. We will later se which N that will cause problems.

IV = [3,255,N] + KEY = [X,X…]→ RC4key[3,255,N,X,X…]

With this IV we are able to calculate the first byte in the RC4key. This attack tries to calculate the first unknown byte in the key. When this one is calculated it goes on to the next one and so on.

The first step in the attack is to calculate the values in the Sbox after the third step (we know the first three bytes in the RC4key) in the initialization function. Before the loop is:

Sbox[i] = i, j=0.

The table shows the values after each loop. All arithmetic is done modulo 256.

Iter-ation	I	J	Swap	New1	New2
1	0	0+0+3=3	[0,3]	Sbox[0]=3	Sbox[3]=0
2	1	3+1+255=3	[1,3]	Sbox[1]=0	Sbox[3]=1
3	2	3+2+N=N+5	[2,N+5]	Sbox[2] =N+5	Sbox[N+5] =2
4	3	N+5+3+X =N+X+8	[3,N+X+8]	Sbox[3] =N+X+8	Sbox[N+X+8] =1

This assumes that N and X are values that will not cause any problem in the calculations above.

After this the start of the Sbox looks like this:

3	0	N+5	N+X+8

The most probable first output in the keystream in this case is N+X+8. This will be the first output if neither position 0, 1 or 3 is swapped in any of the remaining steps. We know that the remaining steps will swap 5, 6, 7 and so on but we do not know the other positions in each swap. If we assume that the second position is completely random, the possibility that 0, 1 and 3 will not be swapped is: $\left(\frac{256-3}{256}\right)^{252} \approx 0.05$. This means that we can calculate X with 5% probability if we know the first byte in the keystream. This can be done because we have the ciphertext and the first byte in the plaintext is determined by the transfer protocol and therefore easy to guess.

19 Appendix D: CRC32

CRC (Cyclic Redundancy Check), or also known as checksum, is a very common method for validating that data transmission has occurred without errors.

The transmitter calculates a checksum for the packet that is being transmitted and adds the checksum to the end of the packet. When the receiver gets the packet he again calculates the checksum and compares the two sums. If they are equal, the receiver can be quite sure that the transmission had no bit fault.

In different applications different variants of CRC algorithms are used separated by the data length and the feedback polynomial.

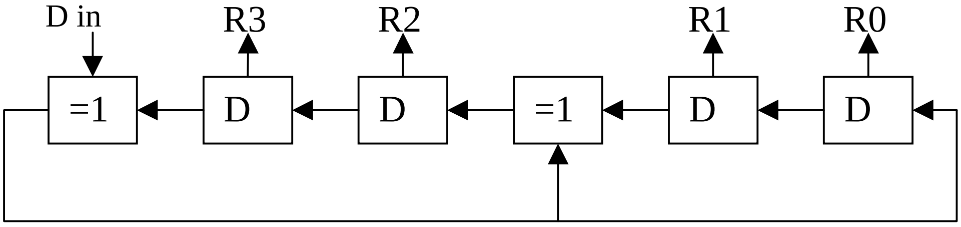

Figure 10. Example of a simple CRC

The CRC algorithm follows these steps:

- Initiate the registers to the start value.
- Input the data in D, usually MSB first.
- Read the checksum from R0 to Rn.

To choose a nonzero start value is important because we otherwise wont detect the number of starting zeroes. If R0-Rn is 0 then a 0 on “D in” will not change the value in R0-Rn from 0 i.e. we could miss a starting zero without detecting it.

The figure shows an example of what a very simple CRC could look like. As we can see this kind of algorithm is very simple to implement in hardware. The only needed hardware is the registers to save the result in and a couple of XOR gates. Little logic between the registers makes it possible to make the hardware fast.

In Ethernet the checksum is 32 bits wide and the polynomial is (32,26,23,22,16,12,11,10,8,7,5,4,2,1,0) i.e. xor gates before R0, R1, R2 and so on. This makes it possible to detect all odd numbers of errors, double bit errors, burst errors of length less than 33 and most of the other burst errors.

Apparently this method is very good for detecting random errors. Unfortunately it is not useful when we wants to protect a message from modification. This can easily be done because the XOR function is linear i.e. $XOR(a+b) = XOR(a) + XOR(b)$. Because of this it is easy calculate the difference for the CRC if you add a known error.