



Secure WLAN Operation and Deployment in Home and Small to Medium Size Office Environments

Diplomarbeit

am Fachgebiet Telekommunikationsnetze
Prof. Dr.-Ing. A. Wolisz
Institut für Telekommunikationssysteme
Fakultät IV Elektrotechnik und Informatik
Technische Universität Berlin

vorgelegt von

Rodrigo Blanco Rincón

Betreuer: Prof. Dr.-Ing. A. Wolisz
Dr.-Ing. G. Schäfer

Rodrigo Blanco Rincón
Matrikelnummer: 205747
Siegmundshof 2-4
10555 Berlin

Abstract

This document¹ studies the way to provide network security to the information sent over a Wireless Local Area Network (WLAN). This kind of networks are specially vulnerable, since the information is transmitted through the air, over a certain physical extension. The IEEE 802.11 specification (for WLANs) includes an encryption protocol, WEP (Wired Equivalent Protocol), but it presents some weaknesses: there is no automatic key distribution protocol, and the WEP's security itself has already been exposed.

The purpose of this project is to fulfill the security needs of users of small to medium-sized WLANs' by providing them with a strong and easy to deploy network security. It seems like these WLAN environments will be of great importance in the near future of the wireless market. A security system tailored for them and their "average" users requires a series of particular features: strong security, simplicity of installation and use, password management policies, user roaming capabilities and no special software or hardware requirements. Hence, these are design conditions imposed in this project.

The approach consists of building a Virtual Private Network (VPN) over the WLAN, using IPSec as underlying security protocol. IPSec is a strong security protocol. However, it assumes that all configuration aspects have been completed by hand in the entities taking part in the secured communications.

The programs implemented as a result of this project's work attempt to solve these deficiencies. User authentication, automatic IPSec policies configuration and automatic IPsec session authentication keys (IKE Preshared Keys) generation are realized with the help of a negotiation protocol between the Mobile Nodes and a Security Gateway (a dual-homed host located between the WLAN and the wired network), which controls the traffic between the Mobile Nodes in the WLAN and between the Mobile Nodes and the wired network and the Internet. A server (which runs on the Security Gateway, optionally as a daemon) and a client (running on the Mobile Nodes) entities have been designed to accomplish the protocol. An IPSec tunnel is built between each Mobile Node and the Security Gateway. The traffic to / from the Mobile Nodes can only flow through these tunnels, which guarantee the information's privacy and integrity, as well as the entities' authenticity.

A trade-off between code simplicity and user-friendliness has been met in this project. The programs provide a reasonable configuration flexibility, although in some cases advanced knowledge is required in order to make use of it. However, the installation process and "basic" operation is quite simple to accomplish.

¹This work has been supported by a grant from Microsoft Research, Cambridge, UK.

Contents

1	Introduction	1
1.1	Issues about secure WLAN deployment	1
1.1.1	Viable solutions for small and middle-sized environments	1
1.1.2	Roaming	3
1.2	Task of this thesis	3
1.3	Overview of the rest of the Document	3
2	Background	5
2.1	WLANs and the IEEE 802.11 standard.	5
2.2	WLAN security risks	7
2.2.1	Unauthorized access to the WLAN	8
2.2.2	Interception and monitoring of wireless traffic	8
2.2.3	Misconfiguration	8
2.2.4	Jamming	9
2.2.5	Client to Client Attacks	9
2.3	Existing approaches to WLAN security	9
2.3.1	IEEE 802.11	9
2.3.2	IEEE 802.1x	12
2.3.3	PPTP / L2TP	14
2.3.4	IPSec	15
2.3.5	Other implemented options	17
2.4	Conclusion	18
2.4.1	Typical scenarios and requirements	19
2.4.2	Level of security	19

CONTENTS

2.4.3	Simplicity of use	19
2.4.4	Key management	20
2.4.5	Roaming and guest users	20
2.4.6	Interoperability	20
3	Requirements and design of a WLAN security solution	21
3.1	Network configuration	21
3.1.1	Access control and node authentication	24
3.1.2	Confidentiality, data integrity and data origin authentication	25
3.2	Choice of a security technology	25
3.2.1	Technology comparison	26
3.2.2	Choice of a technology	28
3.2.3	Adapting IPSec to the proposed scenarios	30
3.3	Configuration issues	34
3.3.1	General considerations	34
3.3.2	Configuration in Windows 2000 Professional/XP operating systems	34
3.3.3	Initial configuration	36
3.3.4	Policy negotiation traffic	37
3.3.5	Name collisions	37
3.3.6	Simultaneous IPsec policies: side effects of the WLAN IPsec security	38
3.3.7	Nodes' passwords	39
3.4	Configuration tools	39
3.4.1	The Security Gateway	39
3.4.2	The Mobile Nodes	41
3.5	The negotiation protocol	42
3.5.1	Security analysis of the dynamic IPSec policy negotiation protocol.	47
3.5.2	Random numbers	49
3.5.3	Security of the signatures: HMAC	51
4	Implementation and evaluation	53
4.1	Tools used in the implementation	53
4.2	The Mobile Nodes' clients: WLANClient	53
4.3	The Security Gateway's server: WLANServer	56

4.3.1	The request profile pool	58
4.4	The databases on the participating entities	58
4.5	Complementary applications	59
4.5.1	Security Gateway IPSec initial configurator	59
4.5.2	Random initializer	60
4.5.3	Security Gateway identifier configurator	61
4.5.4	Client WLAN disconnecter	61
4.5.5	WLAN Service installer	62
5	Conclusion and Outlook	65
5.1	Limitations of the solution and possible extensions	65
5.1.1	Adaptability	65
5.1.2	GUI	66
5.1.3	Scalability	66
5.1.4	Maintenance of the Security Gateway	67
5.1.5	Portability	67
5.2	Remaining security issues	67
5.2.1	Broadcast traffic	68
5.2.2	Other IPSec limitations	68
5.2.3	Unconfigured Mobile Nodes	68
5.2.4	Passwords	69
5.2.5	Responsibility of the users	69
5.3	Tests and results	70
5.3.1	Test 1	70
5.3.2	Test 2	70
5.3.3	Other tests	70
5.4	Outlook of the project	71
5.4.1	Future of the WLAN network security	71
5.4.2	Assessment of this project in the landscape of WLAN security	72
	Bibliography	73
	A Overview of the implementation program files	77

CONTENTS

A.1	Description of the components of the programs	77
A.2	Files contained in the programs	77
A.2.1	Common files	77
A.2.2	WLANServer	78
A.2.3	WLANClient	79
A.2.4	RandomInit	79
A.2.5	InitialIPSecConfigurator	80
A.2.6	SGNameConfigurator	80
A.2.7	WLANDisconnect	80
A.2.8	WLANService	80
A.3	Folder structure and build instructions	80

Chapter 1

Introduction

A WLAN is a LAN (Local Area Network) that holds its communications through high frequency radio waves, rather than wires. Due to their relatively low price and easy installation, this kind of networks represent a very convenient alternative to both wired networks and the connection of nodes to a wired network.

Many IEEE 802.11 WLANs are operated in a completely insecure manner, representing an easy-to-attack target for even the most unskilled attackers, who happen to pass by near a building where an IEEE 802.11 WLAN is operated. Therefore, in WLANs there is a stronger need for security than in their wired counterparts.

1.1 Issues about secure WLAN deployment

In this section, the scenarios considered are described. In principle, this project is intended to fulfill the security needs of small and medium-sized WLAN environments, which include both home and enterprise scenarios.

1.1.1 Viable solutions for small and middle-sized environments

The usual scenario of roaming access to wireless LAN infrastructures is the following: a number of mobile stations (typically notebooks with wireless cards) get connected to the WLAN in order to access the corporate resources. These resources include Internet access, printers and other devices.

We will consider two typical cases in which this kind of networks are used: in small corporate environments, and at home. Figure 1.1 shows a basic architecture for a corporate networking scenario.

In this architecture, we observe different parts. Normally, there will be one or more base stations that transmit the information belonging to one or more WLANs. The base stations are directly connected to the wired LAN (Ethernet), and constitute the actual interface between the wired and wireless LANs.

In principle, the IP addresses assigned to the mobile nodes in the WLAN are private. Hence, a NAT (network address translation) operation will need to be performed before traffic from these machines

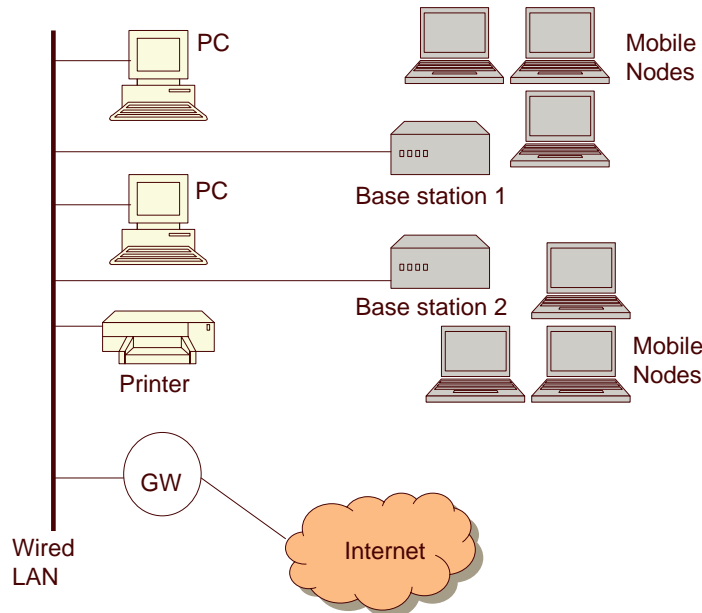


Figure 1.1: WLAN Scenario in Small to Middle-Sized Enterprise Environments

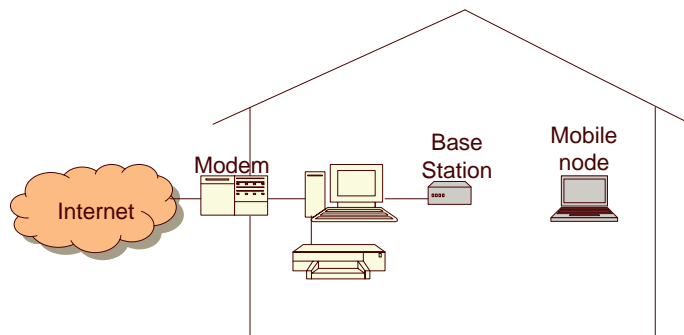


Figure 1.2: WLAN Scenario in Home Environments

can go out into the Internet (through the gateway, GW). Note that the network resources such as printers are shared among all users. Wireless users should also have access to these resources. But the network resources remain attached to the wired LAN.

Another typical environment is the usage at home. In this case, it is usual to have a computer connected to the Internet through some gateway device (ISDN adapter, DSL modem, cable modem, etc.). The base station may be directly wired to this computer. In some cases the home configuration could also include a small fixed network as depicted in Figure 1.2 connecting a few PC's with a printer and a PC accessing the Internet.

The mobile node is any notebook carrying a wireless card that accesses the other resources (Internet, printer, peripherals, etc.) through the PC which probably has no wireless card. This scenario is especially useful for those who use a notebook to work outside their home and want to get connected to

the Internet or have access also at home to other resources of the home wired PC with their notebook.

There is an issue inherent to both scenarios: users should be able to connect their wireless-enabled notebooks to one or more of these networks and, in each of them, be able to hold their communications securely. Just imagine the typical worker who uses his notebook in his company's WLAN, and also in his own WLAN when he gets home. These are two completely different environments, but he will still be expecting to have some network security features in both of them.

Thus, what we are looking for is a general solution, which can be applied to any scenario. This means that no complicated operations in the mobile nodes or in the wired networks (or in the home-case, in the PC) should be required. Users should remain relatively unaware of the details regarding the underlying security mechanisms: they should be able to just turn on their notebooks, authenticate themselves, and start working. In general, they should nevertheless be somehow conscious that they have secured their communications.

1.1.2 Roaming

It is also a desirable feature in these networks that the users can roam from one to another seamlessly. This means that the change of WLAN, access point and wireless environment remains to them as transparent as possible.

1.2 Task of this thesis

As the wireless medium is easier to attack than the wired one, special security measures must be taken in order to ensure the protection of the data sent over these networks.

The purpose of this project is to evaluate the different technological options available in order to achieve the security level required in enterprises or at home. Afterwards, a feasible solution is to be designed and implemented.

The program will be developed to run over Windows platforms, since Windows is to date the most deployed operating system.

1.3 Overview of the rest of the Document

The rest of the document is organized as follows: in section 2, a background of the WLAN security problems is provided, including WLAN security risks, available security technologies and other issues. Finally, some conclusions are drawn. In section 3, the requirements and design of this project's solution are described. In section 4, the actual implementation of the WLAN security configuration solution is discussed. In the last section, conclusions are reached and an outlook to the future and the initial purposes is made.

Chapter 2

Background

This chapter provides a brief technical background of the IEEE 802.11 standard. The WLAN security risks that arise from relying on it are analyzed. Afterwards, the alternative available security technologies are studied: this project will be based on one of them.

Finally, some conclusions are drawn regarding the needs of a WLAN security solution (including both security and environment configuration). These conclusions will be the basis for the design of the configuration solution in Chapter 3.

2.1 WLANs and the IEEE 802.11 standard.

The standard IEEE 802.11 [20] describes the WLAN architecture. It defines two different network architectures: Ad-hoc Network and Infrastructure Network.

The Ad-hoc Network configuration (Figure 2.1) defines the following entities:

- Station (STA): terminal which accesses the wireless medium through a conformant MAC (medium access control) and PHY (physical layer) interface and contacts the access point (infrastructure configuration), or other Stations (ad-hoc configuration).
- Basic Service Set (BSS): group of Stations using the same radio frequency.

It allows direct communication between end systems within a limited range. These systems are grouped in Basic Service Sets (groups of stations using the same radio frequency). As there is no further infrastructure, no communication is possible between different Basic Service Sets.

The Infrastructure Network configuration (Figure 2.2) introduces a new few entities:

- Access Point(AP): entity with station functionality which provides access to the distribution services via the wireless medium for the associated stations.
- Portal: the logical point at which the medium access control (MAC) service data units from a wired network enter the Distribution System of an Extended Service Set (ESS).

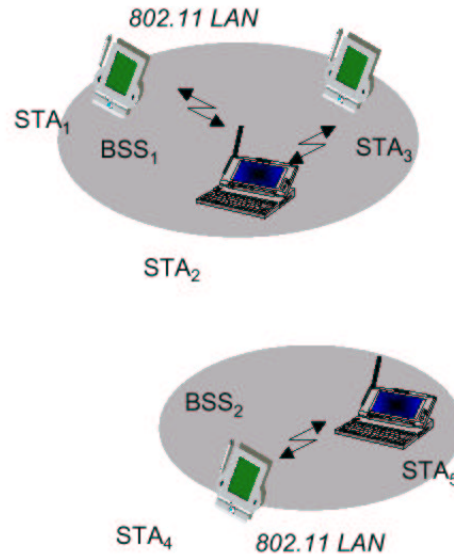


Figure 2.1: Ad-hoc Network Architecture [28]

- Distribution System (DS): a system used to interconnect a set of basic service sets (BSS).
- Extended Service Set (ESS): a set of one or more interconnected BSSs and integrated wired LANs that appear as a single BSS to the logical link control layer at any associated station.

The Access Points provide access to the Distribution System (DS). Through this infrastructure (APs and DS), the different BSSs can communicate with each other. The Distribution System connects the different BSSs with the wired network (typically an 802.x LAN) through a Portal.

In this project, the second configuration, Infrastructure Network, will be taken into consideration.

In general, wired data networks face the following threats:

- Masquerade: when an entity claims to be a different entity
- Eavesdropping: an entity can read information which it is not meant to access
- Authorization violation: an entity uses a service it is not allowed to use
- Modification or loss of transmitted information: data is altered or destroyed
- Repudiation of communication acts (repudiation): an entity later falsely refuses to have participated in a communication event.
- Forgery of information: an entity creates information in the name of another entity
- Sabotage: an action intended to reduce the availability and / or correct functioning of services or systems.

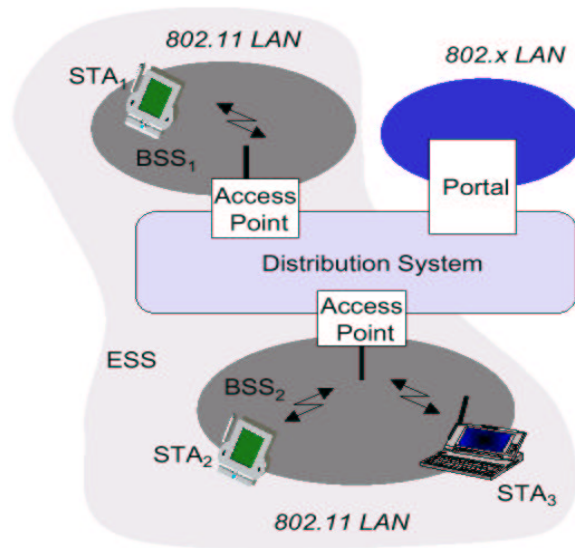


Figure 2.2: Infrastructure Network Architecture [28]

As wireless medium networks, WLANs must face specific threats:

- Wireless communications are more accessible to eavesdroppers
- The lack of physical connection makes it easier to access services

If the Mobile Nodes are expected to be working in truly mobile conditions, the following threats are also applicable:

- Key management is harder since the peer entities cannot be previously determined
- The location of a device / user becomes a more important information worthwhile to eavesdrop on and is, thus, to be protected.

The IEEE 802.11 also describes the authentication and privacy mechanisms available in WLANs. This standard comprises some primitive security protocols called “wired equivalent privacy” (WEP) and “Shared Key Authentication”. However, they provide a limited level of protection due to cryptographic weaknesses.

2.2 WLAN security risks

In the following sections, some inherent risks in the IEEE 802.11 standard will be discussed.

2.2.1 Unauthorized access to the WLAN

When unauthorized devices enter the wireless infrastructure without going through the corresponding security process and review, getting associated to a base station. This may happen for several reasons, such as:

- Wired Equivalent Privacy (WEP, see Section 2.3.1) is turned off
- enterprises where a base station has been attached to the wired LAN without taking any security measures.

2.2.2 Interception and monitoring of wireless traffic

This risk is present in all Ethernet-like networks (and hence in WLANs).

- Sniffers are devices that allow an intruder to monitor all the traffic sent over the network. In WLANs this is specially easy, since the access to the wireless medium is guaranteed in the emitting range of the base stations (which usually exceeds the physical perimeter of the enterprises).
- If an attacker is able to sniff all the traffic, he could find out the MAC and IP addresses of the entities present in the WLAN. With this information, he could then issue malicious commands impersonating his victim by injecting traffic and “hijacking” his victim’s session.
- If a base station is connected to a hub, rather than a switch, any traffic across the hub can be potentially sent over the WLAN, aggravating the risk: traffic from the wired network can reach the wireless medium.
- Arp spoofing: if an attacker has access to the arp traffic flowing in the WLAN, he could make use of tools like Dsniff to gain access to sensitive data from other subnetworks accessible from the wireless access point, although this traffic would normally never flow into the wireless network.
- BaseStation clone: if the attacker places a base station of his own in the proximity of the WLAN, the users might attempt to connect their terminals to the attacker’s server, giving away valuable information. For example, if a user maps a certain network drive, or just by exchanging NetBIOS traffic, his password could be exposed.

2.2.3 Misconfiguration

In most cases, the out-of-the box wireless base stations are by default set to the minimal security configuration. In principle, it is up to the network administrators to enable WLAN security features:

- SSID: The Server Set ID is an identifier shared by the base stations and the clients which are to get connected to them. It is possible to configure the environment so that only clients having

the same SSID can communicate to the corresponding base station (although this option is not enabled by default). If not changed after installation, this SSID will remain to be the default one, which is the same for all base stations of the same model. WEP, the encryption standard for IEEE 802.11, does not encrypt the management packets, and therefore the SSID goes through the air in clear text (and hence can easily be eavesdropped on).

- **SNMP:** Many base stations have SNMP configuration agents running on them. If the SNMP community word is not properly configured, an attacker could read and, in some cases write, data on the devices.
- The clients also store valuable information (SSID, WEP key) which could be exposed under certain conditions (client's misconfiguration). On Windows machines, for example, this data can often be found in the Windows registry.

2.2.4 Jamming

Illegitimate traffic overwhelming the frequencies used by the WLAN devices can avoid the legitimate traffic to get through, in a denial of service attack.

2.2.5 Client to Client Attacks

In an Ad-hoc Network configuration (Section 2.1), clients can talk directly to each other without going through a Base Station. It could be dangerous to allow other clients this kind of access. In this sense, file sharing services are specially dangerous. In any case, a client could always flood another one with bogus packets.

However, this risk does not affect the scenarios considered in this project, which correspond to an Infrastructure Network configuration.

2.3 Existing approaches to WLAN security

2.3.1 IEEE 802.11

The standard IEEE 802.11 (for WLANs) includes some basic security services which are integrated in the WLAN environment:

- Entity authentication
- Wired Equivalent Privacy, WEP

The authentication should be performed between stations and access points and could also be performed between arbitrary stations. In an authentication event, an entity acts as “requestor” and the other as “responder”:

Table 2.1: Notation of the Shared Key Authentication protocol

Notation	Meaning
A	Authentication requestor
B	Authentication responder
ID_A	Identifier of A
r_B	Challenge text proposed by the responder
$K_{A,B}$	Shared secret key

1. The requester sends a first authentication frame to the responder:

$$A \rightarrow B : (Authentication, 1, ID_A) \quad (2.1)$$

2. Before sending the second frame, the responder shall use WEP to generate a string of octets that shall be used as the authentication challenge step:

$$B \rightarrow A : (Authentication, 2, r_B) \quad (2.2)$$

3. The requester shall copy the challenge text from the second frame into the third frame. The third frame shall be transmitted after encryption with WEP, using the shared secret key:

$$A \rightarrow B : \{Authentication, 3, r_B\}_{K_{A,B}} \quad (2.3)$$

4. The responder shall attempt to decrypt the contents of the third frame in the authentication sequence. If the WEP check is successful the responder shall then compare the decrypted contents of the Challenge Text field to the challenge text that was sent in frame 2 of the sequence. If they are the same, then the responder shall respond with a successful status code in frame 4 of the sequence. If the WEP check fails, the responder shall respond with an unsuccessful status code in frame 4 of the sequence:

$$B \rightarrow A : (Authentication, 4, Successful) \quad (2.4)$$

There are two authentication modes:

- Open System Authentication: no authentication at all. If this type of authentication is chosen, the process ends with the second frame.
- Shared Key Authentication: supports the authentication of STAs as either a member of those who know a shared secret key or a member of those who do not. In this authentication mode, the four protocol messages are used. WEP is necessary to accomplish all the steps in the authentication process. The required shared key is supposed to have been delivered to each participating STA independently of IEEE 802.11.

The WEP protocol is supposed to ensure the following security features:

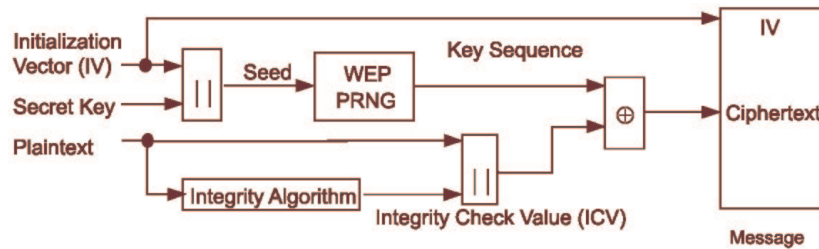


Figure 2.3: WEP encipherment block diagram [20]

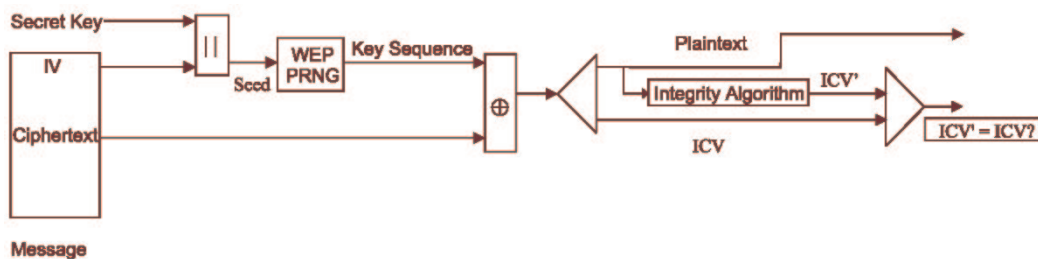


Figure 2.4: WEP decipherment block diagram [20]

- Confidentiality: only stations in possession of the shared key (Group Key) will be able to read the WEP-protected messages.
- Data origin authentication/data integrity: modified packets are easily detected by checking an integrity code.
- Access control: if set so, only WEP-protected messages will be accepted by receivers, so that intruders cannot send messages to those stations.

It works as follows: the 40 (or 104, depending on the WEP mode) most significant bits of the secret encryption key are concatenated with an initialization vector (IV) in order to seed a PRNG (pseudo-random number generator). With RC4, a pseudo-random bit sequence is computed and, later, XORed with the plaintext and a CRC of the plaintext. The cipher and decipher processes can be viewed in Figure 2.3 and Figure 2.4.

IEEE 802.11 limitations and WEP vulnerabilities

The Wired Equivalent Privacy can be configured in 3 modes: no encryption, 40 bit key length and 104 bit key length encryption. By default out of the box, no encryption is enabled in the base stations. If WEP is not activated in the wireless APs (access points), clients cannot use WEP encryption. Some access points use, by default, vendor-dependent default WEP encryption keys, which is obviously a wrong practice.

If the “Shared key” mode is chosen, the secret shared keys are supposed to have been previously delivered to the stations through some protected channel. This channel is not implemented within the

802.11 standard. The passwords need to be setup manually in every base station and mobile node. This can become a great deal of work depending on the number of mobile nodes (and base stations) in the WLAN, and the frequency with which this key is to be refreshed. This causes a large percentage of these networks to be operated with no protection at all and, therefore, open to even the most unskilled attackers.

Furthermore, the WEP protocol presents a series of cryptographic weaknesses that make it relatively easy to attack under certain circumstances. Apart from some security weaknesses initially found (view [6]), a much more important security flaw was discovered in early August 2001. A new attack to WEP had been discovered, with which the network key could be retrieved in less than 15 minutes provided that about 4 to 6 million packets have been recovered. The required effort grows only linearly with the number of bits used in the key, so using 40 or 104 bit keys makes virtually no difference at all. The weakness and the attack are described in [32] and [35]. This proves WEP to be insufficient to protect data flowing across WLANs.

2.3.2 IEEE 802.1x

IEEE 802.1x [21] is an IEEE standard approved in June 2001 that provides authentication and key management for IEEE 802 local area networks, including 802.11 WLANs. It does not provide encryption or encapsulation, and therefore adds no overhead to the packets.

It is supposed to ensure the following security properties:

- Entity authentication
- Key distribution

In the systems, the concepts of “controlled port” and “uncontrolled port” are introduced. The uncontrolled port allows to authenticate a device (and therefore communication prior to authentication), and the controlled port allows an authenticated device to access LAN services, only after a successful authentication of the device has taken place.

There are three distinguished roles (Figure 2.5):

- Supplicant: a device that wants to use the service offered by an IEEE 802.1x LAN and requests access to the controlled port.
- Authenticator: it is the point of attachment to the LAN infrastructure (i.e. a MAC bridge) demanding the supplicant to authenticate itself.
- Authentication server: the Authenticator does not check the Supplicant’s credentials itself. Instead, it sends them to the Authentication Server for verification.

IEEE 802.1x does not define its own security protocols, but recommends some already existing ones.

For basic device authentication, the Extensible Authentication Protocol (EAP) [5] may be used. The EAP messages are sent inside EAPOL (EAP over LANs) packets. EAPOL defines the encapsulation techniques used in order to carry EAP packets between supplicants and authenticator entities

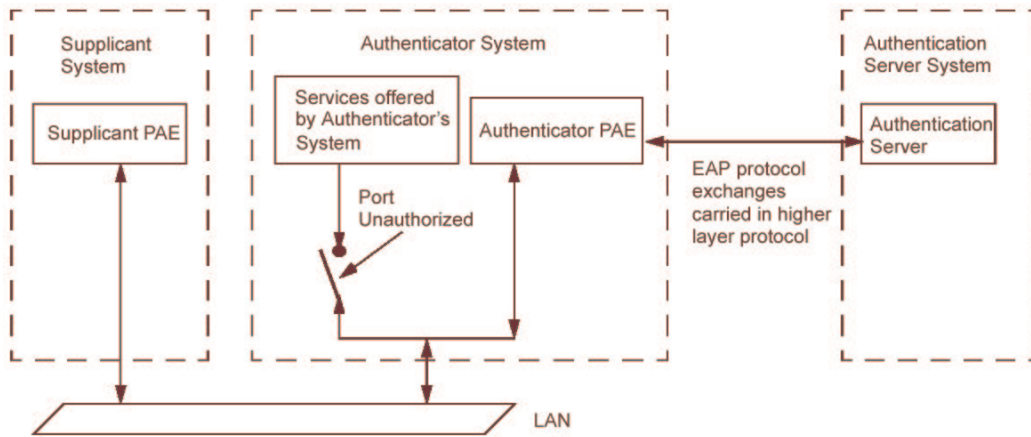


Figure 2.5: Authenticator, Supplicant and Authentication Server roles (taken from the IEEE 802.1x standard)

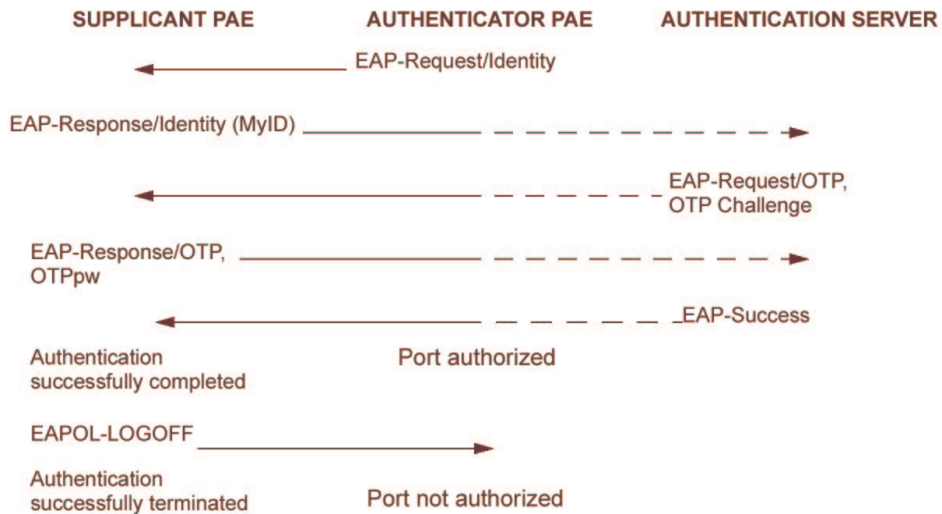


Figure 2.6: Successful authentication followed by Supplicant-initiated logoff (taken from the IEEE 802.1x standard)

in a LAN. If negotiation of a session key during authentication is required, the use of the PPP EAP TLS Authentication Protocol [1] is recommended. The Authentication Server is recommended to be realized with the Remote Authentication Dial In User Service (RADIUS) [27].

The authentication process can be initiated by the Supplicant or by the Authenticator. A typical authentication exchange is depicted in Figure 2.6:

The 802.1x standard is intended to solve the key delivery problem of the wired equivalent protocol (WEP) with the help of 802.1x. A key management protocol is to be achieved - which provides the STAs with keys automatically. For added security, the keys can be changed rapidly at set intervals. Through EAP-TLS, dynamic keys can be negotiated with the STA's at certain periods of time. This

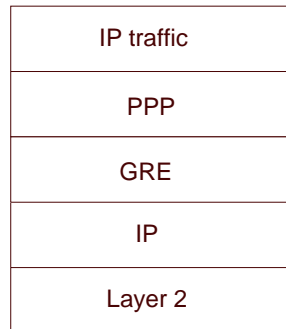


Figure 2.7: PPTP protocol stack

helps to enhance WEP-protected WLANs' security. The more often the keys are changed, the better security is achieved.

Windows XP, the new Microsoft operating system, for example integrates support for the IEEE 802.1x protocol. Access is controlled per-user and/or per-port (point of attachment to a network). Hardware vendors, including Agere, Cisco and Enterasys, are enclosing this technology in their devices.

In principle, no significant security flaws have yet been discovered in this standard (it is a brand new standard). However, WLAN environments using IEEE 802.1x keep relying on WEP as encryption protocol, which has been badly exposed. Another drawback is the possible limitations to upgrade the already existing wireless devices to this technology.

2.3.3 PPTP / L2TP

The Point-to-Point Tunneling Protocol (PPTP) is a protocol that enables the secure transfer of data from a remote client to a private enterprise server by creating a virtual private network (VPN) across TCP/IP networks. It is an extension of the Point to Point Protocol (PPP), described in the IETF RFC 1661 [33].

It can be used by computers connected to a LAN to create a virtual private network (VPN) across the LAN. In fact, it extends the reach of the Point to Point Protocol (PPP) to the whole Internet. PPP was intended to be run between “directly” connected entities at the link layer level. The basic idea is to define transport of PPP PDUs in IP packets. The PPP PDUs become, thus, the payload of the IP packets. More precisely, the PPP packets are encapsulated in GRE (General Routing Encapsulation) packets, and these are inserted into IP packets, as shown in Figure 2.7

So, PPTP implements a tunnel that carries PPP packets over the Internet. Its security properties are, hence, those of PPP:

- Authentication: if required, it is requested by the peer entity. Originally, two authentication protocols were defined: PAP (Password Authentication Protocol) and CHAP (Challenge Handshake Authentication Protocol). Later, other protocols were defined: EAP (Extensible Authentication Protocol, also present in 802.1x) and PPP-EAP-TLS (PPP EAP Transport Level Security Protocol).

- Encryption: it can be negotiated after authentication. Three protocols are available: ECP (Encryption control Protocol) for the negotiation, and DESE (PPP DES Encryption) and 3DESE (PPP Triple DES Encryption) for the exchanged data.

PPTP was strongly supported by Microsoft, who collaborated in its development. Unfortunately, some security flaws were discovered in PPTPv1 and PPTPv2 (see [29] and [30]), and PPTP could not be adopted by the IETF working groups as a standard protocol. Cisco had meanwhile developed a similar protocol, L2F, and it was decided to merge the strong points of both into a new protocol: L2TP [38].

PPTP and L2TP coincide in that both encapsulate the IP packets into PPP packets, and extend the reach of PPP to the whole Internet by allowing the endpoints to reside on different devices. A weak point for both is that they add a relatively high overhead to the exchanged information, thus wasting bandwidth (although L2TP supports, under certain circumstances, a header compression mode that reduces this problem [39]).

But while PPTP requires an underlying IP network, L2TP provides support for different technologies (IP, Frame Relay, ATM, X.25, etc.). It also allows multiple tunnels to be built between the endpoints (PPTP allows only one tunnel to be built). Furthermore, L2TP provides tunnel authentication, while PPTP does not.

2.3.4 IPSec

The basic architecture of IPSec is defined in RFC 2401 [4].

IPSec is widely used in VPNs (Virtual Private Networks) that implement secure channels over untrusted networks (Internet or, in our case, WLANs). It provides security services at the IP layer.

IP is a stateless, connectionless protocol. In order to provide stateful security, IPSec sets up Security Associations (SA), which are “simplex” connections that provide security services to the traffic carried by them. These Security Associations can be nested, allowing to have different IPSec relationships active on the same link. In order to establish an SA, IPSec relies on the Internet Security Association and Key Management Protocol (ISAKMP) [24] [26], which defines protocol formats and procedures for security negotiation. The Internet Key Exchange (IKE) [19] defines the IPSec’s standard authentication and key exchange protocol.

IPSec can assure the following security properties:

- Replay protection: the protected packets carry a sequence number to avoid replay attacks. A replay sequence number window is defined, and only packets whose sequence number is in the window are accepted (of course, after they have gone through the corresponding authentication process).
- Data origin authentication / connectionless data integrity: this means that in the IP datagrams, both the source and the destination address cannot be masqueraded (maliciously modified) without the receiver detecting it. The protected IP datagrams cannot be changed in transit without the receiver’s detection, and cannot be later replayed.

- Confidentiality is provided in two senses. In the first place, someone who is not taking part in the IPsec association cannot read the information contained in the protected IP datagrams. Secondly, when using “tunnel mode” (see below), there is also limited traffic flow confidentiality. This means that at least one of both, the source and target addresses can be read only by one of the IPsec tunnel endpoints.

An important feature of IPsec is its flexibility. It allows the IPsec communication endpoints to negotiate and agree what authentication/encryption protocols they want to use in order to protect the IP datagrams.

There are two main IPsec protocols: AH (providing data origin authentication and replay protection) [22] and ESP (providing data origin authentication, confidentiality and replay protection)[3] . A single SA can use one of them, but not both at the same time.

IPsec can be operated in two modes: transport mode (when the “cryptographic endpoints” coincide with the “communication endpoints” of the secured IP packets) and tunnel mode (when at least one of the “cryptographic endpoints” is not a “communication endpoint”).

IPSec is controlled by policies and rules. If an entity has IPSec switched on, that means that a certain IPSec policy is active. In a single entity, several policies may be defined, but only one may be active at the same time. From a certain active IPSec policy, it is possible to:

- switch to another (inactive) policy, so that it becomes active
- switch IPSec off and viceversa.

An IPSec policy consists of a collection of rules, which establish the behaviour of the IPSec policy over the traffic that flows to/from/over the entity. For the transport mode, normally a two-way rule is enough. In order to build a tunnel, two rules are needed (one for each sense of the communication). In principle, there is no limit on the number of rules an IPSec policy can contain. This implies that a single IPsec policy can define several different IPSec tunnel and / or transport mode relationships with different entities.

An IPSec rule consists of a series of data which determine a certain treatment to a specified traffic. As an example, a policy to be configured for the Windows 2000 / XP operating system¹ consists of:

- IP filter list: it is a collection of traffic filters. Each traffic filter defines a traffic flow by its source address, destination address, transport-level protocol (TCP/UDP, etc.) and ports. It can be mirrored, implying that the traffic going in the contrary direction is also affected by the rule (this option is only used in transport mode; for tunnel mode, the filters cannot be mirrored).
- Filter action: it defines what treatment is to applied to the traffic flows defined in the IP filter list. There are three options: “allow” (let the packets through without any control), “block” (drop the packets) and “negotiate security”. If “negotiate security” is activated, different encryption (confidentiality) and hash algorithms (data origin authentication / data integrity) are available.

¹These are the parts as made accessible in the Windows IPSec implementation.

As encryption algorithms, DES is declared to be mandatory in the standards [4], and 3DES and Blowfish are recommended, among others. SHA-1 and MD5 must be supported as hash algorithms.

- Authentication methods: if the traffic is to be negotiated (filter action = “negotiate security”), an authentication method must be defined (see the options mentioned above) for the IPsec initialization processes. For the authentication, three options are available:
 - Preshared Key: a string known by both endpoints taking part in the IPsec association.
 - Certificates: digital certificates distributed and signed by issuers which are trusted by both parties.
 - Kerberos: it implies a Kerberos authentication infrastructure to be setup in the WLAN scenario, which is not present unless having a Windows 2000 Server or such an operating system.
- Tunnel setting: in case the rule specifies one of the senses of a tunnel, it must specify which is the endpoint of that tunnel, that is, where is the tunnel counterpart entity. If the rule is a transport-mode rule, no tunnel endpoint is specified.
- Connection type: it allows to define to which connections is this rule to be applied. There are three types of connections: “all network connections”, “LAN” and “remote access”.

IPsec has, though, a limitation: it does not protect broadcast traffic. Microsoft claims that its IPsec implementation protects multicast traffic [9] if operated in tunnel mode.

IPsec is a built-in feature in the Windows 2000 operating system, and tools are provided for its configuration (although the configuration is unfortunately still quite difficult for the average users).

2.3.5 Other implemented options

The most relevant software solution is NetMotion from NetMotion Wireless. The architecture introduces a “Mobility Server” as the interface between the WLAN and the wired infrastructure, that runs on a centrally managed Windows NT/2000 Server, providing authentication (Active Directory, NTLM, Kerberos, and PKI) through user logon, encryption of the data in the wireless links (AES, Twofish, 3DES, or DES), inter-network mobility (seamless roaming without reauthentication and session-persistence through roaming between different network technologies, such as wired networks, 802.11 networks, Bluetooth, CDPD, GSM, GPRS and 3G networks). Clients logon and access to the security and roaming services through the “Mobility Client”. No further information can be obtained in the web page, but it seems to solve the WLAN security problem. Its main drawback is its price [14].

The Wireless Firewall Gateway of the NASA’s Advanced Supercomputing Division [7], implemented in August 2001, stands as an example of possible future solutions. It introduces a Firewall, WFG, between the wireless and the wired infrastructures. This entity ensures access control through web authentication secured with SSL. It uses a RADIUS server as authentication server and is able to dynamically update the IP filtering rules. No confidentiality is included in the NASA WFG Whitepaper [7].

Table 2.2: Comparison with the prototype developed in this project and the NetMotion commercial solution.

	<i>NetMotion Wireless</i>	<i>NASA WFG</i>
<i>Roaming</i>	<i>Between wired networks, 802.11 networks, Bluetooth, CDPD, GSM, GPRS and 3G networks. No reauthentication required. Session persistence.</i>	<i>802.11 networks. Reauthentication required.</i>
<i>Security</i>	<i>Authentication: Active Directory, NTLM, Kerberos, and PKI. Encryption: AES, Twofish, 3DES, or DES. Interoperability for PPTP, L2TP/IPSec, IPsec, and Cisco VPNs. Configurability independent for each user.</i>	<i>Authentication: Web interface (SSL). RADIUS access to user profiles. IP filtering.</i>
<i>Management</i>	<i>“Configuration Manager”, interoperates with NTLM or Active Directory. Real-time monitoring capabilities.</i>	<i>Dedicated, secured workstation accessing the WFG through SSH.</i>
<i>Scalability</i>	<i>Any environment, up to thousands of users.</i>	<i>Not specified.</i>
<i>Architecture</i>	<i>“Mobility Server”, “Mobility Client”.</i>	<i>“Wireless Firewall Gateway”, Mobile Nodes, Radius Authentication server</i>
<i>HW requirements</i>	<i>One or more dedicated machines (Windows NT/2000 Servers).</i>	<i>1 dedicated OpenBSD Unix server (WFG) + 1 dedicated workstation (management) + [1 dedicated server (RADIUS)]</i>
<i>Price</i>	<i>Expensive (commercial)</i>	<i>Not distributed</i>

A comparison of the NASA prototype and the NetMotion commercial security solution is summarized in Table 2.2.

NetMotion is not a viable option for the scenarios considered in this thesis, since no special or expensive software should be expected. The NASA WFG is neither valid because although it ensures access control through entity authentication, it provides no confidentiality in the wireless network.

2.4 Conclusion

At this point, some conclusions can be drawn regarding the WLAN security.

First, WLANs are unsafer than their wired counterparts for various reasons. A wireless medium is of course more subject to attacks, since it can be more easily accessed. The IEEE 802.11 standard presents some security weaknesses, and the devices which are compliant to it are not easy enough to configure to take full advantage of the 802.11 security features in many cases. This is certainly a serious problem, since the most common users of this kind of networks are small to middle sized enterprises’ employees and home users, who are normally no security experts.

The security technologies which can be used in order to secure WLANs can be classified into two groups: WEP-based solutions and non-WEP-based solutions. The former are IEEE 802.11 and IEEE 802.1x. These options cannot longer be considered to be valid, since the WEP security has been completely exposed [32]. The latter correspond to the so called VPN-oriented technologies. These were not designed explicitly for WLAN environments but, instead, as general-purpose protocols that can be more or less adapted to the WLAN security needs.

For the time being, these VPN-oriented solutions have only been applied to LAN environments by some software vendors, such as Certicom (movianVPN) or Columbitech, among others. Unfortunately, they are not free (in some cases expensive) solutions. They are therefore inadequate for a large fragment of the potential users of WLANs.

Bearing all these factors in mind, a series of desired features can be listed for a viable WLAN security solution.

2.4.1 Typical scenarios and requirements

Due to their low price, simplicity of deployment and bandwidth limitations, the typical environments for WLANs will be small to middle-sized enterprises, as well as home users, as described in Section 1.1.1. This has a double effect on the desired solution.

First, no expensive or complicated software such as Kerberos or a Windows 2000 Professional Server running Active Directory should be required. In fact, it should run on a normal workstation.

Furthermore, no special or extra hardware devices should be used in order to implement the gateway to the rest of the corporate network. A normal workstation with a standard operating system should be enough.

Summing up, the solution should not be proprietary, but follow an open standard instead, and the software and hardware requirements to run it should not exceed the capabilities of a standard workstation.

2.4.2 Level of security

Also in middle and small enterprises, as well as in home scenarios, the network security can be essential. The chosen security protocol should be, therefore, strong enough to protect critical and security sensible data so that an attack with the known tools or techniques, or taking advantage of the corresponding protocol's security flaws is unfeasible. It is for this that the WEP protocol is not an acceptable solution, as it has been proven to be broken.

The conclusion drawn here is that the solution will need to make use of VPN technology instead of the IEEE 802.11 or 802.1x standards.

2.4.3 Simplicity of use

The target users should not need to be security experts to configure their WLAN infrastructures to attain security on their networks. Actually, the ideal solution would be that all the systems taking part in the WLAN had some sort of agent running on them that would make the configuration automatic and transparent to the users when they entered the WLAN (just as DHCP works). This would, of course, imply some sort of user authentication step (per-user passwords, etc.).

In short, the solution must be able to do the job of authenticating and configuring the users' machines in an automatic fashion, that is, either as a service daemon or button-like program (just press and everything works).

2.4.4 Key management

The key management is an essential part of a security solution. There are many possibilities, but all of them can be classified into two groups: automatic key distribution (through some key distribution protocol), and manual key setting (the user types in his password in some GUI on his machine), depending on how the keys reach the users' devices. Logically, the automatic key distribution simplifies the users' devices' configuration. In either case, it must be as safe as possible.

Furthermore, if one of the keys is exposed, as many keys as possible should remain unaffected by this fact. Apart from saving a lot of work, this principle minimizes the risks if a key is exposed. In short: the key management mechanism should be designed in such a way that the collateral effects of a key-exposition are minimal.

In this sense, the difference between individual keys (one different key for every WLAN user) and shared keys (the same key for every WLAN user) is vital: it is safer to have per-user keys, although effort should be made on keeping this complexity as transparent as possible for the users and administrators. If a shared key is used and it is exposed to some attacker, he could, under certain conditions, impersonate *any* legitimate entity in the environment.

2.4.5 Roaming and guest users

It would also be highly interesting that allowed users (and only allowed users) could easily gain access to foreign networks with little configuration effort on their devices. The typical case is, for example, a freelance worker who usually works at home and wants to get connected to the corporate WLAN of his customer. This, of course, should not immediately mean that the customer's network administrator gains administrative rights on the guest's workstation.

2.4.6 Interoperability

This feature hints at the need of a protocol specification as open as possible (from this point of view, only publicly documented solutions are acceptable).

If this configuration solution is adopted extensively, it would be highly desirable that software of different vendors can still interoperate. This would assure that no matter what the vendor of some WLAN client is, he can still roam into any WLAN running this configuration, and would still be able to make use of its services.

In this sense, the possibility of support under different operating systems is also an important feature.

Chapter 3

Requirements and design of a WLAN security solution

In this chapter, the wireless environment's particular features are analyzed. A security technology will be selected in order to protect the WLAN communications, and the way to adapt it to the proposed scenarios and automate its configuration will be studied. Finally, a configuration solution will be proposed.

3.1 Network configuration

The elements that can be found in the WLAN environment to secure are:

- Base Stations (BS): the wireless devices that provide the Mobile Nodes with access to the wireless medium (Base Stations are called Access Points in IEEE 802.11).
- Mobile Nodes (MN): the wireless enabled computers that take part in the WLAN communications as users (Mobile Nodes are called Stations in IEEE 802.11).
- Security Gateway (SG): device that acts as an interface between the WLAN and the wired network. It has gateway functionality. (Also called Portal in IEEE 802.11).
- Security Domain (SD): WLAN environment protected with a homogeneous security policy and unique configuration properties. It is identified by the Security Gateway installed within the WLAN.

Typically, the mobile nodes get attached to the base stations. Over them, they reach the wired network through the Security Gateway. In Figures 3.1 and 3.2, this topology is depicted for both environments described in Section 1.1.1. The Security Gateway is drawn as another workstation. A priori, it would not necessarily have to be a workstation but, as it will be explained later (see section 3.1.1), it turns out to be the best choice. Other options such as routers, switches or even a direct connection to the wired LAN through some hub, have been discarded for security reasons (see Section 3.1.1).

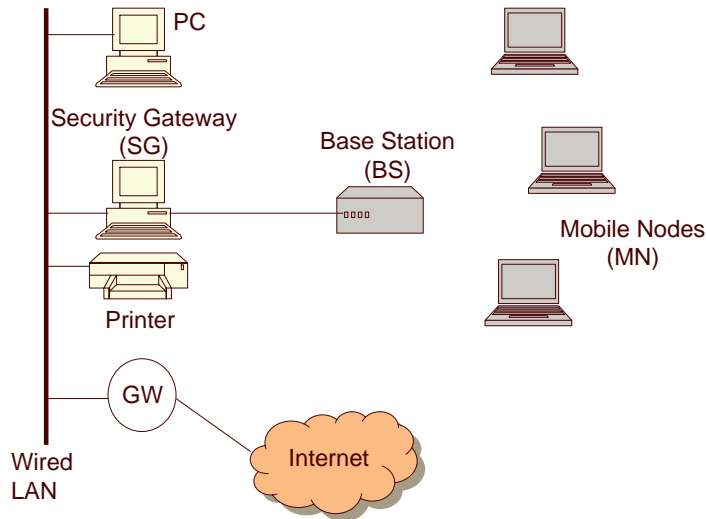


Figure 3.1: WLAN layout for an enterprise environment

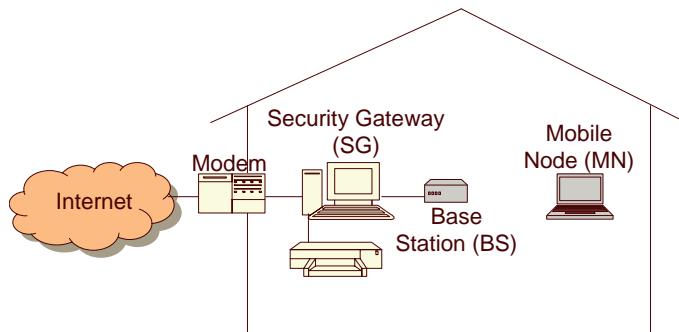


Figure 3.2: WLAN layout for a home environment

The interface between the wireless and wired network will be a workstation, which will be referred to as Security Gateway in this thesis. It is a dual homed host that acts as the router between the mobile nodes and the wired network infrastructure. The Base Station¹ is directly connected to the Security Gateway's interface (Ethernet adapter) to the WLAN IP subnetwork. This is depicted in Figure 3.1 and Figure 3.2.

A Security Domain is a wireless environment managed by a single Security Gateway. However, one Security Gateway could manage more than one Security Domain, so that what really defines the limits of a certain Security Domain is the relationship between Mobile Nodes' names and passwords (namespace). A Security Domain is limited to a C-class IP subnetwork's address range (network mask 255.255.255.0, 255 addresses). For each additional network interface (adapter) installed in the Security Gateway, another C-class IP subnetwork allows a new Security Domain to be supported.

¹It could happen that there is more than one Base Station.

Summing up, each network interface defines a new C-class IP subnetwork. This IP subnetwork supports a Security Domain in which the Mobile Nodes' identities and credentials are independent to those valid for the rest of the Security Domains. In a Security Domain there is a theoretical maximum number of 253 Mobile Nodes (which is an acceptable number for small and medium-size environments), although in practice Base Stations cannot manage more than about ten Mobile Nodes at the same time.

Although in Figure 3.1 and Figure 3.2 only one Base Station has been drawn, it would be possible to install more than one Base Station if, for example, the physical extension of the Security Domain requires it. The solution is to connect a hub/switch to the Security Gateway's IP interface for the Security Domain. Different Base Stations could be then connected to the hub/switch.

The following security goals should be ensured:

- Only authorized Mobile Nodes can hold communications in the WLAN.
- Only authorized Mobile Nodes can reach the other Mobile Nodes in the WLAN, the wired network or, through it, the Internet.
- Unauthorized Mobile Nodes can neither eavesdrop the packets sent over the WLAN nor talk to any other Mobile Node (authorized or not).

And some networking issues:

- Authorized Mobile Nodes can talk to other Mobile Nodes in the WLAN, or to any other entity through the wired infrastructure (that is, they can also access the wired network, the Internet, etc.).
- The IP addresses of the Mobile Nodes in the WLAN should be private (although this is not compulsory). Generally, a whole private subnet (with IP mask 255.255.255.0) should be reserved for the WLAN. 255 IP addresses should be enough for home and small to middle-sized enterprise environments.²
- Additionally, none of the entities (Mobile Nodes, Base Stations, Security Gateway) must require special or expensive hardware or software components.

These security and networking goals must be respected throughout the different WLAN's visited by a Mobile Node. This feature is very important in relation with the roaming capabilities. However, it points to a problem which will need to be faced: the dynamic configuration of the Mobile Nodes, depending of the WLAN.

From the previously discussed conditions, it is possible to draw some conclusions.

²In any case, and due to the WLAN bandwidth limitations, so many devices are very unlikely to be operating on the same WLAN at the same time.

3.1.1 Access control and node authentication

There must be an Access Control mechanism, so that only registered (authorized) nodes can access the other Mobile Nodes or the wired network. The best choice to place such a control entity on seems to be the Security Gateway, for the following reason. The Base Stations are specialized hardware and, therefore, difficult to modify. Moreover, the Security Gateway can be a normal workstation on which it is possible to install some sort of authentication service.

This point also provides a hint on the Security Gateway's nature. Various options can be thought of: a simple hub, a switch, a router, etc. A simple hub is not a valid solution, since it directly connects the WLAN to the wired network, making it impossible to control what information flows from one side to the other, or between Mobile Nodes and other entities. A switch or a router do not resolve the problem either, since they have limited control capabilities over the information which flows through them. Furthermore, they are not flexible enough to support some sort of database containing the authorized WLAN nodes and other security-related information. This leads to the solution adopted in this project: the Security Gateway is an IP forwarding-enabled workstation running some easy to configure network security software (which is the subject of this project).

This service is, in fact, substituting the Base Stations built-in (WEP) access control functionality. It decides which Mobile Nodes are authorized to talk to the other Mobile Nodes, or to any entity through the wired network. Mobile Nodes need to "sign in" before they access the other Mobile Nodes or the wired network through the Security Gateway.

Unauthorized Mobile Nodes might try to contact directly authorized Mobile Nodes. This is possible, since the Base Station is performing no access control. In that case, those packets should be dropped by the authorized Mobile Nodes, since they are not coming from a "trusted" entity (as it will be later exposed, the only "trusted" entity for every Mobile Node will be the Security Gateway).

Another problem arises: two unauthorized (and hence not configured) Mobile Nodes might still talk to each other, since they do not need to go through the Security Gateway; just through the Base Station, which is applying no control over the packets that flow through it. However, they will never be able to access the properly configured Mobile Nodes or the wired network, because their packets will never be accepted by the Security Gateway, and they can only be accessed through it. This is the consequence of performing the access control at the Security Gateway, instead of doing it at the Base Stations. In any case, this does not endanger the security of the registered, signed-in nodes.

If unauthorized Mobile Nodes try to talk directly to the Security Gateway, or some entity belonging to the wired network, their packets would be dropped by the Security Gateway. A way to accomplish this is to configure the registered Mobile Nodes after the "sign in" step so that they send all their traffic initially to the Security Gateway, who will conveniently forward it (acting as a gateway). They must also accept traffic only if it comes through the Security Gateway.

Thus, all the traffic to / from the WLAN's registered Mobile Nodes must go through the Security Gateway, even if the communication is held between two Mobile Nodes. Hereby, the Mobile Nodes will need to talk only to the Security Gateway. This has a series of network design implications:

- The IP Default Gateway for the Mobile Nodes is the IP interface of the Security Gateway to the WLAN (this requires the IP settings, more precisely the Default Gateway, to be updated on the

Mobile Nodes).

- All the packets sent by the Mobile Nodes, independently of their target address, must go directly to the Security Gateway, which will forward them conveniently, depending on their address (this requires the routing table of the Mobile Nodes to be configured).
- The Security Gateway must have the routing capabilities enabled (this requires the IP forwarding option to be switched on).

3.1.2 Confidentiality, data integrity and data origin authentication

The data sent over the WLAN should be impossible to eavesdrop for unauthorized entities (nodes). Furthermore, authorized entities must be able to notice if the packets that they are receiving have been modified by a malicious third party, or sent by an attacker impersonating an authorized WLAN entity.

This implies the following security properties:

- Confidentiality
- Data integrity and data origin authentication

Confidentiality is guaranteed with encryption of the data sent. The data integrity of the data is checked on arrival at the target entity, through some sort of integrity check. The origin authentication is implemented by using some sort of digital fingerprint of the sent data.

This network security functionality will not be implemented from the scratch in this project. Rather, the target is to find an already existing security protocol which provides with these functions and adapt it to the actual needs and particular characteristics of the WLAN environments.

3.2 Choice of a security technology

As pointed out in the previous section (see section 3.1.2), a security protocol is needed in order to accomplish all these design conditions. WEP is not an acceptable security protocol (see section 2.3), since its security has been broken. Hence, the solution will implement a VPN over the WLAN. The most relevant VPN technologies available are:

- PPTP
- L2TP
- IPsec

Essentially, all of them are valid options. One of them must be chosen in order to secure the communications of the WLAN. An informal comparison of their features will be exposed. The choice will clearly have an important effect over the achieved WLAN security.

3.2.1 Technology comparison

While IPSec is a Layer 3 protocol, PPTP and L2TP are Layer 2 protocols.

As PPTP and L2TP are based on the PPP protocol (what they do is, roughly, encapsulate PPP packets in IP packets, see Section 2.3.3), they make use of its authentication primitives (PAP, CHAP, EAP, etc.). These are entity (user) authentication schemes; PPTP/L2TP have no packet data origin authentication / data integrity (which means that in each sent packet, some information is added to the payload, identifying the source of the packet to the receiver and assuring that it has not been modified), and do not perform replay protection. They do not make use these features taking them from PPP, since PPP does not implement them either. The lack of packet authentication / data integrity and replay protection is an important drawback for PPTP and L2TP. IPSec provides entity (host) authentication of the security association's endpoints during the IKE's Security Association negotiation through different authentication mechanisms (Kerberos, Certificates, Preshared Key), which are entity (machine) authentication schemes. IPSec, by itself, does not provide user-based authentication but it implements packet data origin / data integrity authentication and performs replay protection. An interesting option is L2TP/IPSec [16], which consists of encapsulating the UDP-wrapped L2TP packets in an IPSec ESP header and trailer (see Figure 3.3). This solution provides: L2TP's user-based authentication and IPSec's system authentication and packet authentication / data integrity and replay protection.

The same applies to the confidentiality: PPTP and L2TP provide no encryption primitives by themselves, but rely on those of PPP [33]. IPSec supports packet encryption, and allows to choose among a set of different encryption algorithms, which is implementation dependant (although a minimal sub-set is mandatory). L2TP/IPSec relies on IPSec's encryption.

In PPTP and L2TP, the tunnel configuration variables are negotiated dynamically, while in IPSec it is assumed that all the configuration has been done before the tunnel establishment, independently of the IPSec negotiation protocols. On the other hand, PPTP and L2TP require tunnel maintenance (for which a special tunnel management protocol is used), while in IPSec no tunnel maintenance is needed.

Regarding the key management, PPTP and L2TP rely on an initial key generated during the authentication step, and refresh it periodically. IPSec negotiates explicitly a common key with the help of the Internet Key Exchange (IKE) [19] protocol and also refreshes it periodically.

L2TP has an advantage over PPTP or IPSec. While PPTP and IPSec work only over IP networks, L2TP provides support for different networks, such as IP, Frame Relay, X.25 or ATM. However, this does not mean much in the WLAN environments proposed, since in them the communications are, in any case, IP-based.

L2TP (and therefore L2TP/IPSec) and PPTP can normally carry IP multicast traffic. IPSec cannot carry this kind of traffic in its "transport mode". However, Microsoft assures that its IPSec implementation can cope with it when operated in "tunnel mode". In these cases, the broadcast traffic principle is broken. Essentially, its advantage is that while only one packet is sent, all the entities in the network receive it. If broadcast traffic is sent on PPTP or L2TP, one packet is produced for each entity with which the sending entity has an established tunnel. Depending on the number of extra packets generated, it could have an adverse effect on the performance of the network.

PPTP:



L2TP:



IPSec:



Figure 3.3: Comparison of the different protocol overhead structures

Microsoft' Windows 2000 operating system supports PPTP, IPSec and L2TP/IPSec. Microsoft states: [10] *“The Windows 2000 IPSec APIs and policy schema have not been published yet. (...) Windows 2000 is compliant with RFC 2661 (“Layer Two Tunneling Protocol”). RFC 2661 indicates that L2TP traffic can be secured with IPSec, but does not provide details about how to implement this security. An Internet-draft document is currently being worked on that will specify the details of securing L2TP traffic with IPSec.”*

PPTP and L2TP tunnels can traverse Network Address Translators (NAT) used to hide one or both end-points of the communication. This limitation of IPSec is due to incompatibilities between the IKE protocol (used by IPSec for the Security Association negotiation step) and NAT. L2TP/IPSec is equally affected by this limitation [9].

L2TP's performance in terms of latency is superior to that of PPTP, partly because its management traffic flows on UDP packets (whereas PPTP's management traffic utilizes TCP connections). In terms of overhead (view Figure 3.3), a comparison can also be made (considering Microsoft's implementations [2] [15] [12] [8] [39]). For the following results, it is assumed that no IP header additional options are used, that the PPP header and padding have the maximal length (10 bytes and 4 bytes³ respectively), that the IPSec padding has a maximal length of 8 bytes (3DES encryption is considered) and that no HMAC is added to the IPSec packets (since in PPTP no packet authentication is present):

- PPTP introduces an extra IP header, a GRE header and a PPP header and padding. Assuming the Windows GRE packet format, the overhead is about 70 bytes.
- L2TP (as implemented in the Windows 2000 operating system, that is, L2TP/IPSec) introduces an extra IP header, an ESP header and padding, a UDP header, an L2TP header and a PPP header. Assuming no L2TP header compression⁴, the overhead is about 100 bytes.

³in order to align to the GRE packet

⁴As explained in the L2TPHC IETF Internet Draft [39], a number of conditions should be met in order to introduce L2TP header compression (L2TPHC). Most probably, some of these conditions will not be met in the proposed scenarios. When activated, L2TPHC reduces the protocol overhead.

- IPSec using ESP (3DES) introduces an extra IP header and an ESP header and padding. The introduced overhead is about 78 bytes.

Both PPTP and L2TP provide compression functionality for the PPP payloads. This would improve the performance of these protocols. IPSec does not provide data payload compression itself, but Payload Compression Protocol (PCP [31]) can still be used on the IP payloads [4].

PPTP presents a series of security weaknesses (see [29] and [30]). L2TP lacks of solid tunnel protection mechanisms, including authentication and encryption services. In fact, the IETF's L2TP working group recommends that at least the L2TP tunnel management traffic is protected by IPsec. At least L2TP provides, optionally, own authentication prior to the establishment of the tunnel: *"The tunnel endpoints may optionally perform an authentication procedure of one another during tunnel establishment. This authentication has the same security attributes as CHAP, and has reasonable protection against replay and snooping during the tunnel establishment process. This mechanism is not designed to provide any authentication beyond tunnel establishment; it is fairly simple for a malicious user who can snoop the tunnel stream to inject packets once an authenticated tunnel establishment has been completed successfully. (...) Securing L2TP requires that the underlying transport make available encryption, integrity and authentication services for all L2TP traffic. This secure transport operates on the entire L2TP packet and is functionally independent of PPP and the protocol being carried by PPP (...) When running over IP, IPsec provides packet-level security via ESP and/or AH. All L2TP control and data packets for a particular tunnel appear as homogeneous UDP/IP data packets to the IPsec system."* [38].

PPTP does not even do that: *"Because the PPTP control channel messages are neither authenticated nor integrity protected, it might be possible for an attacker to hijack the underlying TCP connection. It is also possible to manufacture false control channel messages and alter genuine messages in transit without detection. The GRE packets forming the tunnel itself are not cryptographically protected. Because the PPP negotiations are carried out over the tunnel, it may be possible for an attacker to eavesdrop on and modify those negotiations. Unless the PPP payload data is cryptographically protected, it can be captured and read or modified."* [18]. Furthermore, important vulnerabilities have been found in the PPTPv1 and PPTPv2 Microsoft implementations [29] [30].

The availability of the different solutions is described in Table 3.1.

IPSec is the protocol recommended by the IETF for VPNs. Most vendors are now using it, as it is the protocol chosen by the IETF for both IPv4 and IPv6. In two years time, no relevant weaknesses have been found in IPSec.

3.2.2 Choice of a technology

After the technology comparison of Section 3.2.1, it is possible to decide which technology fits better in the proposed scenarios' requirements and features.

The authentication provided by PPTP and L2TP is that of PPP: user authentication, that happens only during the tunnel establishment. L2TP provides an additional authentication step. No packet data integrity / origin authentication, as well as no replay protection are provided. IPSec does not implement user authentication. Instead, it performs an entity-based authentication protocol (IKE). This

3.2. CHOICE OF A SECURITY TECHNOLOGY

Table 3.1: Comparison of the available VPN technologies

	<i>PPTP</i>	<i>L2TP/IPSec</i>	<i>IPSec</i>
<i>Security services</i>	<i>User-based authentication. No packet data origin / integrity authentication. PPP encryption. No replay protection</i>	<i>L2TP: User-based authentication. No packet data origin / integrity authentication. PPP encryption. No replay protection. IPSEC: Machine-based authentication (IKE). Packet data origin / integrity authentication. ESP encryption. Replay protection.</i>	<i>Machine-based authentication(IKE). Packet data origin / integrity authentication. ESP encryption. Replay protection.</i>
<i>Tunnels</i>	<i>Dynamic configuration of variables. TCP management.</i>	<i>L2TP: Dynamic configuration of variables. UDP management. IPSEC: Previous (static) configuration. No tunnel management.</i>	<i>Previous (static) configuration. No tunnel management.</i>
<i>Key Management</i>	<i>Initial key generation and periodic refreshment</i>	<i>L2TP: Initial key generation and periodic refreshment. IPSEC: IKE. Initial key generation and periodic refreshment</i>	<i>IKE. Initial key generation and periodic refreshment</i>
<i>Multi-network</i>	<i>PPP Payloads supported: IP, IPX, NetBEUI. IP-based PDU transport.</i>	<i>L2TP: PPP Payloads supported: IP, IPX, NetBEUI. Works in different network technologies: Frame Relay, ATM, X.25 and SONET. IPSEC: Payloads supported: IP. IP-based PDU transport.</i>	<i>Payloads supported: IP. IP-based PDU transport.</i>
<i>Broadcast</i>	<i>YES (with individual IP unicast packets)</i>	<i>YES (with individual IP unicast packets)</i>	<i>NO</i>
<i>Overhead</i>	<i>LOW</i>	<i>HIGH</i>	<i>Intermediate</i>
<i>Level of Security</i>	<i>LOW (no data integrity, no replay protection)</i>	<i>HIGH</i>	<i>HIGH</i>
<i>Availability</i>	<i>Windows, Linux, FreeBSD, Solaris, MacOS</i>	<i>Windows, Linux</i>	<i>Windows, Linux, FreeBSD, Solaris, MacOS, AIX</i>

could imply a weakness in multiple-user machines, in which perhaps not all the users are authorized to make use of the tunnel. However, that is not likely to happen in the WLAN: most probably, the Mobile Nodes are single-user notebooks. It also protects the packets with data integrity / origin authentication trailers and performs replay protection, while PPTP or L2TP do not (and do not inherit that functionality from PPP either). IPSec key management scheme (IKE) is much more flexible than that of PPTP or L2TP. It allows flexible authentication options, such as X.509 certificates and Kerberos.

The PPTP and L2TP tunnels are quite different to those of IPSec. PPTP and L2TP tunnels support dynamic configuration of their variables during the tunnel negotiation. They also need a continuous maintenance (implemented with a TCP connection in PPTP and a UDP protocol in L2TP). This requires both establishment time and bandwidth. IPSec tunnels perform no dynamic tunnel variable configuration. The drawback is that all the configuration must be performed previously and manually. However, IPSec tunnels require no further maintenance. PPTP and L2TP tunnels can traverse NAT, while IPSec cannot. Anyway, no NAT is to be traversed by the VPN tunnels in the proposed WLAN environments (since they are built between each Mobile Node and the Security Gateway).

The fact that L2TP tunnels can be built on different network technologies, such as Frame Relay, ATM or X.25 provides no advantage since in our scenario the underlying network is IP-based. Neither does the PPP's ability to process different payload protocols (IPSec can not), since the only protocol considered is IP.

Multicast and broadcast traffic is protected by PPTP and L2TP. Microsoft's IPsec implementation claims to protect multicast traffic, but that does not afford true multicast in the WLAN. If IPsec is deployed, the broadcast packets would be unprotected. This is perhaps the only item in which PPTP and L2TP are clearly preferable to IPsec. However, there are some inherent drawbacks of broadcast and multicast in PPTP and L2TP, as explained in Section 3.2.1.

The performance is quite a problematic feature to compare. In Section 3.2.1 it was shown that L2TP/IPsec introduces the biggest overhead and PPTP introduces the smallest overhead. Additionally, PPTP and L2TP support payload compression, and L2TP also header compression (under certain circumstances). In IPsec compression can also be performed with the Payload Compression Protocol. The improvement achieved using these options has been not tested. L2TP and PPTP they need extra control traffic for tunnel maintenance. IPsec introduces an intermediate overhead and requires no tunnel maintenance. In this sense, IPsec seems to be the preferable option (the overhead difference between IPsec and PPTP is only 8 bytes).

PPTP is discarded a priori, due to its security flaws [29][30]. It is recommended that L2TP uses some lower-level protection, such as IPsec [38][16], due to some limitations, for example the limited protection of the L2TP tunnels. The choice would be then between IPsec and L2TP/IPsec. L2TP/IPsec introduces more overhead than IPsec and it requires tunnel maintenance, which is performed through a UDP-based management protocol. These two features hint at bandwidth requirements that could have a certain adverse effect over the WLAN bandwidth-limited performance. The additional tunneling of L2TP appears to be unnecessary having in mind that IPsec tunnel mode already tunnels the IP traffic. L2TP/IPsec would also provide the ability of carrying payloads different to IP, but in our scenario that is irrelevant, since the payload will be solely IP. As exposed in Section 2.4.6, interoperability with other platforms might be an issue in the future. In this sense, IPsec is available in virtually all of the relevant operating systems, while L2TP/IPsec is not [11].

IPsec fits better the proposed environments' features and needs and it provides a stronger and more flexible security solution. Therefore, it has been adopted as the VPN technology to protect the WLAN environments.

3.2.3 Adapting IPsec to the proposed scenarios

Once IPsec has been chosen as security technology, some design decisions must be taken with regard to the network configuration and IPsec options. The IPsec protocol (AH or ESP), the IPsec mode (transport or tunnel) and the authentication method, among other issues, will be studied in this section.

Protocol

IPsec supports 2 protocols: AH and ESP. AH (Authentication Header) provides data origin authentication and replay protection. ESP (Encapsulating Security Payload) provides data origin authentication, confidentiality and replay protection. The choice is obvious: only ESP can protect data from malicious eavesdropping, since AH does not include encryption (confidentiality). 3DES is selected as encryption algorithm and SHA-1 as hash algorithm.

IPsec mode

It can be operated in two modes: transport mode (when the “cryptographic endpoints” coincide with the “communication endpoints” of the secured IP packets) and tunnel mode (when at least one of the “cryptographic endpoints” is not a “communication endpoint”). In our case, the cryptographic endpoints will not normally concur with the communication endpoints: the cryptographic associations will occur among members of the WLAN, but no further. In many cases, the Mobile Nodes will be communicating with entities outside the WLAN (through the wired network) that may not know about IPsec. For this reason, tunnel mode will be used.

The following question arises: between what entities should the IPsec tunnels be built? Can any Mobile Node build an IPsec tunnel to any other Mobile Node? Of course, this would be a possibility. But it would complicate too much the WLAN security client entities running on the Mobile Nodes. Just imagine that every node needs to be registered in every Mobile Node in order to be able to talk to it.

The tunnel partner of the Mobile Nodes must be able to perform a mutual authentication with them. It is much more sensible to centralize this complexity in a single workstation, the Security Gateway, which acts as a WLAN security server. All the tunnels are established between the Security Gateway and the Mobile Nodes, who need not know about the other WLAN Mobile Nodes. This way, the information about the different WLAN Mobile Nodes is centralized, and it is not so difficult to update client data, add/remove users, etc. This can be viewed in Figure 3.4.

IKE Authentication

In IPsec, the IKE authentication step, which is intended to negotiate the IPsec Security Association, can be performed using one of these three methods: Preshared Keys, Certificates and Kerberos. Home and small to medium-sized enterprise environments are unlikely to be deploying Kerberos⁵, so this option has not been pursued.

Using digital certificates would be an elegant solution to the authentication problem. The users, as well as the Security Gateway, would produce an RSA key pair. A WLAN-local Certification Authority (CA) would sign the users’ Public Keys, producing WLAN-local user certificates. With these signed Public Keys, their Private Keys and the Public Key of the local Certification Authority, users would produce Windows PKCS12 certificates and place them in the corresponding IPsec policies. In the proposed scenarios, this approach has a number of implications:

- The certificates used by the Security Gateway and *all* of the users for the IKE authentication are signed by the same CA. This implies that users trust not only the Security Gateway, but also *every* entity signed by the local Certification Authority, namely the other users. Essentially, the trust relationship should be restricted to each Security Gateway - Mobile Node pair. If the security of an entity registered under a certain Security Domain is exposed, this would imply that some attacker might have access to an entity that the other users of that Security Domain *trust*.

⁵After all, no special or sophisticated software components must be required in this solution

- If some client's certificate is exposed, the only solution in order to avoid endangering the rest of the users registered under the same Security Domain would be to enter the exposed certificate in a revocation list. However, this solution is not very convincing: in order to avoid accepting such certificates, the entity would need access to some server where the revocation list is available. In our scenario, this cannot be expected: when the IKE negotiation takes place, users have no Internet access. Another possibility would be to produce a new CA in the affected Security Domain, and renew *all* the users' certificates. Their old certificates would need to be removed from their machines and the new ones would have to be installed. In either case, this solution is inefficient.
- CAs are generally operated under specific security conditions. This kind of conditions cannot be expected from home and small to middle-sized environments. Operating a CA under poor security conditions is an unsafe approach.

These implications prove that using certificates for the IKE authentication step does not suit the proposed scenarios' needs.

So the only option left is Preshared Keys. From this standpoint, a Preshared Key (a simple string which matches on both IPsec partners) will be kept on both the Security Gateway and the Mobile Nodes, or created dynamically every time they want to establish an IPsec association. Of course, the Preshared Key of the Security Gateway with each Mobile Node is different. The Preshared Key of every Mobile Node with each Security Gateway is also different.

In the proposed environments, the Preshared Keys for every pair Mobile Node - Security Gateway are generated dynamically and refreshed for every new session, as explained in Section 3.5.

Configuration files containing information (including the Preshared Secrets) about the registered peer entities (Mobile Nodes for the Security Gateway, Security Domains for the Mobile Nodes) must be kept on every Mobile Node and Security Gateway.

Structure of the IPsec-secured WLAN

The WLAN environment secured with IPsec will have a number of particular characteristics. The Mobile Nodes communicate solely with the Security Gateway, which decides if the packets should be forwarded and in what direction. So, in fact, the network segment to be protected encompasses the connections of all the Mobile Nodes with the Security Gateway.

Every Mobile Node establishes an IPsec tunnel with a Security Gateway, which acts as the IPsec association counterpart for every mobile node. This must be accomplished whenever a Mobile Node roams into a new Security Domain. That means that the IPsec policy for both the Mobile Node and the Security Gateway must be dynamically updated whenever a new entity roams into a new Security Domain. This can be viewed in Figure 3.4. This tunnel is the product of a Mobile Node authentication protocol run, in which the Mobile Node and the Security Gateway negotiate dynamically the tunnel configuration parameters. This protocol has a double functionality: Mobile Node and Security Gateway mutual authentication and generation of a session IPsec Preshared Key for the IPsec tunnel.

If a mobile node wants to establish a communication with an IP address of the wired infrastructure or the Internet, it needs to send its packets through the IPsec tunnel to the Security Gateway, which

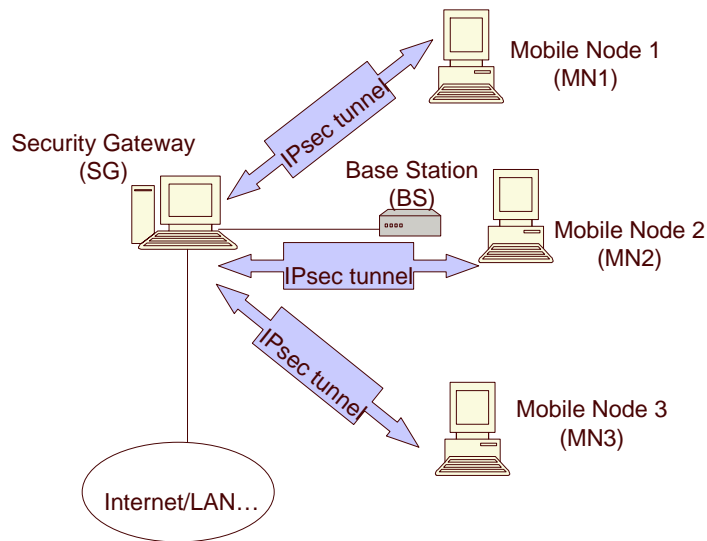


Figure 3.4: Scenario and Tunnel Configuration for IPsec Protection of WLAN Traffic

will route them outside. If it wants to talk to another node in the WLAN, it must first send his packets to the Security Gateway (through the IPsec tunnel), who will forward the packets to the other mobile node. Thus, the exposed part of the network, that is, the wireless links, is protected.⁶

The Mobile Nodes only accept traffic coming from the Security Gateway, through the IPsec tunnel. This way, non-authorized entities cannot access them: if they try to talk directly to the authorized (and properly configured Mobile Nodes), their packets will be blocked by the legitimate entities. If they try to talk to entities belonging to the wired network or to the Security Gateway, their packets will be blocked (unless they are tunnel negotiation protocol traffic). Unfortunately, two unconfigured Mobile Nodes could still talk to each other using the WLAN's bandwidth, since no access control can be performed in the Base Station on the basis of IPsec.

Further configuration

Up to now, IPsec seems to fit fairly well in the proposed scenarios' security needs: it provides packet data integrity and origin authentication. It also performs an initial IKE authentication (entity-based) in order to set up the IPsec Security Associations [19], based on some information generated independently of IPsec. However the configuration of the IPsec policies is assumed to have been performed by hand before the tunnel can be established. That implies that some additional authentication must be provided somehow, in order to setup the authentication material (Preshared Key) used in the IKE authentication.

Thus, some functionality must be added to automate the dynamic modification of IPsec policy that is required whenever the user roams into another security domain. These issues will be described in Sections 3.3, 3.4 and 3.5.

⁶This is why IPsec's tunnel mode is required: one part of the IPsec association must be obligatorily the Security Gateway, and at the same time, the Security Gateway does not necessarily have to be one of the communication endpoints.

3.3 Configuration issues

3.3.1 General considerations

There are three major configuration aspects:

- The distribution of the IPsec policy to the Mobile Nodes in an automatic fashion.
- The dynamic configuration of the IPsec policies on the Security Gateway, in response to the client's requests for WLAN security.
- The configuration of the Security Gateway so that it does the routing of the packets to / from the WLAN.

The IPsec security policies (or the sensible data, from which they can be derived) should be automatically delivered to clients when they switch on their machines so that they do not need to configure the IPsec settings by hand. This is even more important considering that they will be probably visiting different WLANs, and do not want to reconfigure their computers every time.

The Security Gateway is the other difficulty: it should be the partner of every IPsec tunnel with every mobile node, do the IP packet routing and, if private addresses are used, also perform a NAT (Network Address Translation) operation on the Mobile Nodes' IP addresses.⁷

The previously exposed problem will be now considered in terms of implementation in a Windows environment. As said before (section 2.4.1), this solution is intended for home users or small to medium-sized enterprises, so no expensive software (such as Windows 2000 Server) or hardware should be required.

3.3.2 Configuration in Windows 2000 Professional/XP operating systems

IPsec is implemented in the Microsoft Windows 2000/XP operating system. It provides ESP and tunnel mode, which are needed in our architecture. A Microsoft Windows 2000/XP machine is used as the Security Gateway described in section 3.2.3. The Mobile Nodes also run this operating system.

After switching on their mobile computer for the first time inside a Security Domain, users would need to set up the key for this Security Domain. That process should be kept as simple as possible. After this first step, they should remain unaware of the underlying security mechanisms that are set up for them. They should only be conscious that their IP communications are protected by IPsec, and that they are now inside a certain Security Domain that guarantees that security.

Since clients enter and leave the WLAN in a dynamic way, some automatic configuration mechanism is needed in order to provide them with an IP address and, if necessary, other relevant information

⁷If the addresses used in the WLAN are private IP addresses (which is quite a good practice for private networks), a NAT (Network Address Translation) must be accomplished somewhere, in order to have inbound IP visibility. Essentially, the obvious solution is to install the NAT-box in the Security Gateway. However, it could also be performed in some outer router in located in the wired network. This is, in any case, not a part of this project, since it is a normal network management task.

about the WLAN as they arrive. In this sense, the best alternative appears to be DHCP (Dynamic Host Configuration Protocol), which is also available for the MS Windows 2000/XP operating system. With DHCP, the Security Gateway can provide the incoming clients with IP addresses valid in the WLAN.⁸

The question is how to configure automatically the IPsec settings on the client's machine without the user noticing it. One part of the information required for IPsec is static, but there is also some necessary data that is dynamic. For an IPsec tunnel policy in the client machine, the following data is needed:

- The IP address of the mobile node, which, in principle, will be a provisional one (delivered through DHCP) and, hence, dynamic.
- The IP address of the Security Gateway, which constitutes the tunnel endpoint. It is essentially static, since the scenario is not likely to be changing (i.e. the IP address of the Security Gateway is not going to be changed very often). But to the mobile nodes, this information is dynamic: in each WLAN he visits, this IP address will probably change. Note that as in the WLAN the IP addresses are private, the IP address of the Security Gateway for two different WLANs might coincide, but this does not have to happen always.
- The piece of information for the authentication step in the IPsec negotiation. In our proposal, the IPsec authentication method used is "Preshared Key". Each Mobile Node holds in its IPsec policy a Preshared Key with the Security Gateway of the WLAN in which he is registered. However, these IPsec Preshared Keys are generated dynamically, from some pre-shared secret between each Mobile Node and each Security Gateway (this pre-shared secret, which is different from the IPsec Preshared Key, is the Mobile Nodes's "password" for each Security Domain). For the same Security Gateway, all the pre-shared secrets with its registered clients are different. This allows to limit the effects of an exposed client's preshared secret (for example, all the Mobile Nodes' secrets need not be changed). The pre-shared secrets of one Mobile Node with each Security Gateway (that is, for each Security Domain) are also different.
- The Security Domain name (it is not strictly necessary for the IPsec configuration, but the Mobile Nodes need to know in which Security Domain they are in order to pick up the preshared secret which corresponds to that Security Domain). The Security Domain name will be directly identified by the Security Gateway's identifier.

The first two pieces of data (IP addresses of the Mobile Node and the Security Gateway) could be quite easily delivered through DHCP, since this protocol provides the required options to send additional information regarding the network environment.

Let us consider the two latter pieces of information. The Security Domain name is a concept that is not supported within the DHCP protocol, but it could still be delivered to the mobile nodes with

⁸DHCP clients are built in Windows 2000 Professional workstations, but no DHCP server is available, unless installing a Windows 2000 Server. However, this operating system is expensive, and it might not suit the budget of the average users. There are two alternative solutions: to enter the IP information by hand on the clients' Mobile Nodes, or to set up a cheap shareware DHCP server on the Security Gateway.

DHCP through the DHCP option extensions. Unfortunately, Windows 2000 DHCP built-in clients do not support these option extensions [13].

The information concerning the IPsec authentication (preshared secrets) is supposed to be present both in the mobile node and in the Security Gateway before the node's "sign in" step occurs. The "sign in" is a policy negotiation protocol. It consists of the negotiation of an IPsec policy that establishes a tunnel between the Mobile Node and the Security Gateway.

In order to achieve a better understanding of the problem, we will do a step-by-step analysis of what happens since the user arrives at the WLAN until the moment in which he can start his IP communications in a safe way:

1. The user switches on his computer and broadcasts a DHCPDiscover query in the WLAN.⁹
2. The DHCP server delivers to him the IP address. It is a good practice to assign addresses statically to the MAC addresses of the WLAN adapters of the mobile devices. This should not be a problem since there are quite a lot of private addresses available and we are building a solution in which not many users will participate in the WLAN (small to middle-size environments). DHCP supports this feature. However, the configuration software must be able to cope with both statically and dynamically assigned IP addresses. Other network information, such as the default router (Security Gateway) IP address or the local DNS suffix can also be delivered through this mechanism.
3. The client machine must now find out in which Security Domain he has entered. If it is a registered node of the Security Domain, it must now proceed with the configuration of the IPsec policy. The Mobile Node retrieves some Security Domain-relative information and uses it to negotiate the IPsec tunnel with the Security Gateway. This negotiation (or "sign in" step), which is implemented as a protocol, as well as the implicated software entities are the focus of this project. If it is not an authorized node, the IPsec tunnel cannot be built.

The whole process should be as transparent as possible to the user: all he/she needs to know is that from now on, his/her IP packets are protected over the WLAN with an IPsec tunnel.

3.3.3 Initial configuration

If no initial IPsec settings are set up in the Security Gateway, an entity could arrive at the WLAN, assign himself an IP address of the subnet and communicate with the Security Gateway (assuming that this IP address has never been used to build an IPsec tunnel before). On the other hand, if the IP address has already been used to build a tunnel with the Security Gateway, the IPsec rule will still be active on the IPsec policy of the Security Gateway, blocking the entrance to the unauthorized client (since he does not know the preshared key which was set up for the previous secure WLAN session).

This first-time effect is to be avoided. This points at the need for an initial IPsec policy configuration in the Security Gateway, in order to "block" the entrance to unauthorized nodes. If the WLAN IP

⁹Optionally, if no DHCP facilities have been installed, both DHCP steps are skipped and the users enter these data by hand on their notebooks or wireless-enabled workstations.

address range consists of a D-type (subnet mask: 255.255.255.0) private subnet, the 255 IP addresses will have to be initially blocked in order to prevent this from happening.

In IPsec filters, there are three negotiation options for the packets sent over a tunnel: “permit” (allow them to go through the filters without further control), “negotiate security” (including cryptographic and authentication functionality) and “block” (drop the packets). In this initial Security Gateway configuration, the “block” option seems to be the most sensible to activate for the whole address range.

There is one exception to this initial configuration. Some of the addresses within the private subnetwork range will correspond to the Base Stations. The Base Stations will normally exchange packets with neither the Security Gateway nor the Mobile Nodes, so in principle “blocking” their IP addresses would not be a great loss. However, these addresses possibly should not be “blocked”, since the Base Stations will most probably not understand IPsec. By “blocking” their addresses, they will become unreachable for configuration through SNMP, if the option is available.

On the Mobile Nodes, this previous step is not needed. Supposing they roam from a WLAN to a new WLAN, the previous IPsec policy will still be active, blocking any other attempt to contact it inadvertently. If they are coming from a wired LAN environment, they should be aware to run the WLAN security configuration software as soon as they switch on their notebooks inside the new WLAN. If they do not, their Mobile Nodes are open to attackers, posing a serious security threat (see Section 5.2.5).

3.3.4 Policy negotiation traffic

When the Mobile Nodes arrive at the WLAN, they will want to negotiate a tunnel with the Security Gateway through some negotiation protocol¹⁰ in order to perform some sort of machine-based authentication, independent of that provided by IPsec (as commented in Section 3.2.3). They will probably also want to obtain some IP settings through DHCP.

If all the IP addresses in the WLAN subnet range are blocked, they will not be able to make their packets arrive at the Security Gateway (which will drop them). That is why additional rules are needed in the Security Gateway’s IPsec policy. The packets belonging to DHCP or the negotiation protocol must be allowed to reach the Security Gateway directly without further processing.

These rules must apply for the whole address range of the WLAN. They must also outlive the following negotiation processes on that IP address. If they do not, the first client will succeed to perform the DHCP protocol run or the tunnel negotiation protocol. But the next clients will be blocked by the previous IPsec policy for that IP address, and will not manage to reach the Security Gateway.

3.3.5 Name collisions

In this WLAN security system the Security Domain is identified by the Security Gateway’s hostname (essentially) and the clients by their Mobile Nodes’ hostnames.

¹⁰This protocol has nothing to do with IPsec’s security negotiation. It is part of this project, and will be thoroughly described in Section 3.5

In a Security Gateway, two users may have given their computers the same hostname (i.e.: MobileNode). If both want to be registered in the same Security Gateway, a name collision will occur. The negotiation protocol by which a client builds up a tunnel with the Security Gateway is based on some node-dependent information stored in the Security Gateway. But if two nodes are registered under the same user's name, how will the Security Gateway know what user information to use for the negotiation protocol when it receives a negotiation request of one of them?

In order to handle this problem, the following solution will be adopted. If a user wants to get registered in a WLAN Security Domain in which there is already a client using the same hostname as his, the Security Domain administrator will provide him with an alternative user's name, which will be unused in that Security Domain. When going in the rest of the WLANs in which he is registered, the WLAN's client will use its hostname. But when negotiating a tunnel with this Security Gateway, it will use the alternative name, only valid for this Security Domain. This must be registered in some Mobile Node's configuration file (see section 3.4.2).

The desired roaming feature of the system involves yet another difficulty. The opposite situation could also happen. It might happen that a user wants to be registered in two Security Domains, which happen to have the same name (their Security Gateways have the same hostname, i.e.: SecurityGateway). A similar solution as that for the user's name collision could be conceived, but it would come across another problem. If also Security Gateways allow alternative names for certain users, a double name collision might occur. This can lead to unsolvable name conflicts or a too complicated negotiation protocol.

The solution adopted is to assign unique Security Gateway names instead of user-chosen names, avoiding both the Security Gateway name collision and the double name collision problems. Thus, the Security Gateway names will consist of the concatenation of the Security Gateway's hostname and some other random characters.

3.3.6 Simultaneous IPsec policies: side effects of the WLAN IPsec security

The users of the Mobile Nodes may have previously configured some other IPsec policy on their workstations, corresponding to some other scenario. It is not desirable that the IPsec policy that is installed on their machines as a result of the tunnel negotiation protocol disables the previous IPsec settings. This is a logical side effect, bearing in mind that IPsec accepts concurrently only one policy.

The logical solution would be to read the active IPsec policy on the Mobile Nodes before activating the WLAN's tunnel policy. Afterwards, the read policy's rules would be added to the WLAN's tunnel policy¹¹. Unfortunately, in Windows it is not trivial to read automatically an IPsec policy's features: another solution will need to be found.

When users decide to leave a certain Security Domain, they should also be able to turn off the IPsec policy that was produced and activated for them within that Security Domain. This operation should also respect the previously configured IPsec policy settings (before the negotiation between the Mobile Node and this Security Domain's Security Gateway).

¹¹In fact, IPsec allows nested relationships or simultaneous tunnels or transport mode associations on the same policy

3.3.7 Nodes' passwords

The nodes are identified within a Security Domain (that is, in a Security Gateway) by an identifier (their Mobile Node's hostname, unless a name collision occurs).

There are two kinds of authentication: policy negotiation authentication and IPSec (IKE) authentication.

The policy negotiation authentication is implemented with a password that constitutes the credentials of an entity (node) in a certain WLAN. For each WLAN in which a node is registered, its password will be different. This password is a secret string shared by every pair Mobile Node - Security Gateway.

The IPSec authentication is performed by the IKE protocol. It is based in one of the three authentication methods defined by IKE. In our case, and as discussed in Section 3.2.3, the IKE authentication method will be Preshared Key. This Preshared Key is a string related to parts or the totality of an IPSec policy in each of the peer entities establishing an IPSec Security Association between themselves.

The node's password will be used in the IPSec tunnel negotiation protocol, that aims at the eventual generation of an IPSec Preshared Key. This Preshared Key is a string known by both peer entities that is used during the authentication step of the IKE protocol, which aims at the negotiation of an IPSec Security Association. It will be inserted in the corresponding IPSec tunnel policy that is configured in order to secure the communications between the peer entities.

This points at the Access Control requirement explained in section 3.1.1. Only the Mobile Nodes in possession of that piece of information (or "password") will successfully complete the IPsec tunnel negotiation protocol.

These passwords will be saved in a configuration file (see sections 3.4.1 and 3.4.2), and related to their corresponding Security Gateway identifiers, in order to be retrieved automatically during the IPsec tunnel negotiation protocol.

3.4 Configuration tools

In section 3.3, the way to accomplish the IPsec configuration was described in some abstract way. But in real life, the administrator of the Security Gateway does not have to be a networks expert to configure it. Clients need not have a special knowledge of either IP networks or the IPsec protocol configuration: they just want to know that their IP traffic is protected when they enter a WLAN.

Therefore, it is very convenient to build some applications that simplify the configuration tasks of the WLAN's users (Security Gateway administrator and clients).

3.4.1 The Security Gateway

The Security Gateway needs some initial configuration:

- IP forwarding enabled (this is accomplished through the corresponding Windows registry en-

try).

- The unique Security Gateway identifier.
- A random source generation program. This random source will be afterwards used by the IPsec tunnel negotiation entities.
- The initial IPsec blocking policy.

After the initial configuration has been performed, the Security Gateway must configure:

- the IPsec policy (dynamically)

The enabling of the IP forwarding must be accomplished only once, as the Security Gateway is set up. It consists of changing a value in the Windows registry. In order to prevent users from changing the registry themselves (which is usually an error prone practice), a “registry file” can be provided. The registry file updates the desired value (the corresponding to IP forwarding) in the Windows registry just by double-clicking on it.

A unique Security Gateway identifier must be produced in order to avoid Security Gateway name collisions (see section 3.3.5). A few randomly chosen characters will be concatenated to the Security Gateway’s name. If the random source is good enough, name collisions will be very unlikely to happen. An application is therefore needed to calculate those random bytes and produce an acceptable (truly unique) identifier and store that string in a file.

As it will later be exposed (see Section 3.5.2), a good random source file will be needed in order to undertake the policy negotiation protocol. This random source file can be produced from many random events / variables. A program obtains some measurements from a random process and produces a file with values generated from the deterministic random samples. This file will be later used by the software entity that negotiates the IPsec tunnel on behalf of the Security Gateway.

In order to initially block all the IP addresses of the WLAN to unauthorized entities, an application must go through the specified IP address range, which must correspond to that of the WLAN. For each IP address, a block filter is added to the IPsec policy of the Security Gateway, making it impossible for non-configured nodes to access the properly configured Mobile Nodes or the wired network.

Finally, a server entity must run on the Security Gateway. It must await the Mobile Nodes’ “sign in” requests, and negotiate the IPsec tunnel which is to be established between it and them. This process is dynamic, which means that as the Mobile Nodes enter the WLAN and send their requests, the Security Gateway must update its IPsec policy in order to accept the new users’ Mobile Nodes. The negotiation is based on two pieces of information: the node’s user name and the node’s password (that is, the pre-shared secret of the Mobile Node with the Security Gateway). The server will use a configuration file where the data of all the nodes (username + pre-shared secret) are registered.

Summarizing, the following tools are required on the Security Gateway:

1. DHCP Server (if DHCP is used)
2. Security Gateway Id Configurator

3. Random Initializer
4. Initial IPsec Configurator
5. Policy negotiation server

3.4.2 The Mobile Nodes

In the Mobile Nodes, the following configuration is needed:

- IP address, default gateway, DNS server, etc.
- A random source generation program. This random source will be afterwards used by the IPsec tunnel negotiation entities.
- IPsec tunnel settings: these settings depend on which WLAN we are in, and hence the mobile nodes need to "discover" in which Security Domain they are and their local Mobile Node's profile (that is the preshared secret to be used in the negotiation protocol with the Security Gateway)

With DHCP, the IP address, default gateway, DNS server, etc. are quite easy to automatically configure for the clients.

The IPsec tunnel settings are not easy to understand and, hence, to configure. Users should not be bothered to deal directly with the IPsec MMC console (which is actually the only GUI means to configure IPsec in Windows 2000/XP). The information relative to the Security Domain (name) should be retrieved for them and the IPsec tunnel to the Security Gateway should be automatically negotiated and set up.

A client application should be developed that allows the Mobile Nodes to "sign in" in the different Security Domains. It must automatically discover the Security Domain in which the Mobile Node is, negotiate with the corresponding Security Gateway's WLAN server an IPsec tunnel (based on an agreed-upon Preshared Key) and configure the IPsec policy with that key on the Mobile Node. The negotiation is based on two pieces of information: the Security Domain's name (Security Gateway's name) and the pre-shared secret of the client with the Security Gateway. The client will use a configuration file where the data of all the Security Domains (Security Domain's name + pre-shared secret with the Security Gateway of that Security Domain) are registered.

In short, the following tools are required on the Mobile Nodes:

1. DHCP Client (if DHCP is used)
2. Random Initializer
3. Policy negotiation client

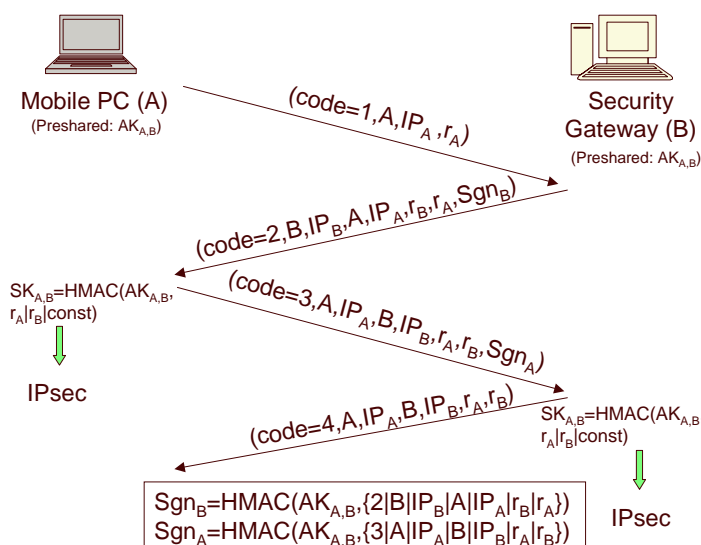


Figure 3.5: Example of a negotiation protocol run between a Mobile Node (A) and a Security Gateway (B)

3.5 The negotiation protocol

In the previous sections, a policy negotiation protocol was introduced. When a Mobile Node enters a new Security Domain, it must run the authentication and IPsec tunnel negotiation protocol. This protocol allows the mutual authentication of the Mobile Node and the Security Gateway of the Security Domain. It also derives a fresh session IPsec Pre-shared Key (see Section 3.2.3) from the pre-shared secret between the Mobile Node and the Security Gateway and some dynamically generated random material.

Two entities take part in the protocol: the Mobile Node (running the *WLANClient.exe* application), acting as client, and the Security Gateway (running the *WLANServer.exe* application), acting as server. A schema of the communication steps can be seen in Figure 3.5.

As the clients arrive at a certain WLAN and switch on their Mobile Nodes, they receive IP settings, such as their IP address, the Default Gateway, etc. through DHCP. If no DHCP entities are available, users enter this information by hand, depending on which Security Domain they are in.

Next, they run their client applications, *WLANClient* (described in Section 4.2), starting the negotiation protocol:

1. The *WLANClient* application sends a request packet to the Security Gateway (Figure 3.6) with the following information:
 - its hostname, MN (that is, the client's name)
 - its IP address, IP_{MN} (recently acquired for this Security Domain)
 - a random number generated by the *WLANClient*, r_{MN}

Table 3.2: Notation of the tunnel negotiation protocol

Notation	Meaning
MN	Identifier of the Mobile Node
SG	Identifier of the Security Gateway
IP_{MN}	IP address of the Mobile Node
IP_{SG}	IP address of the Security Gateway
r_{MN}	Random number (challenge) generated by the Mobile Node
r_{SG}	Random number (challenge) generated by the Security Gateway
SGN_{MN}	Signature of the Mobile Node over frame 3
SGN_{SG}	Signature of the Security Gateway over frame 2

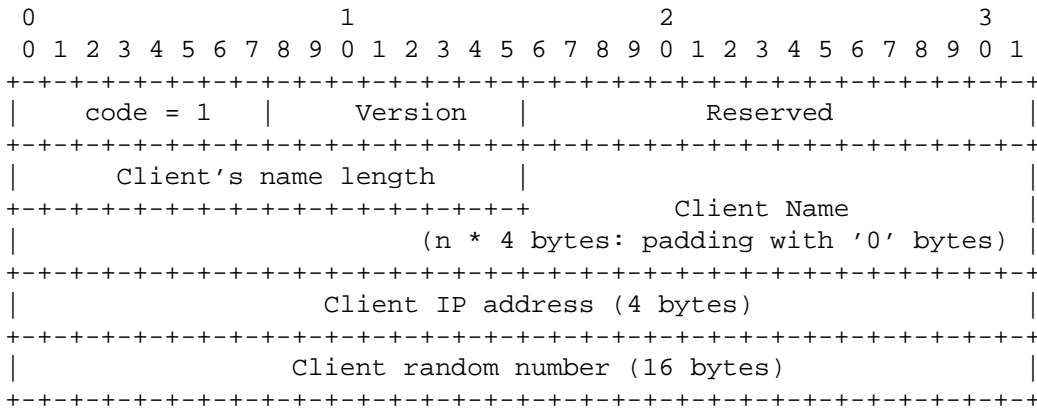


Figure 3.6: Initial request of the WLANClient to the WLANServer

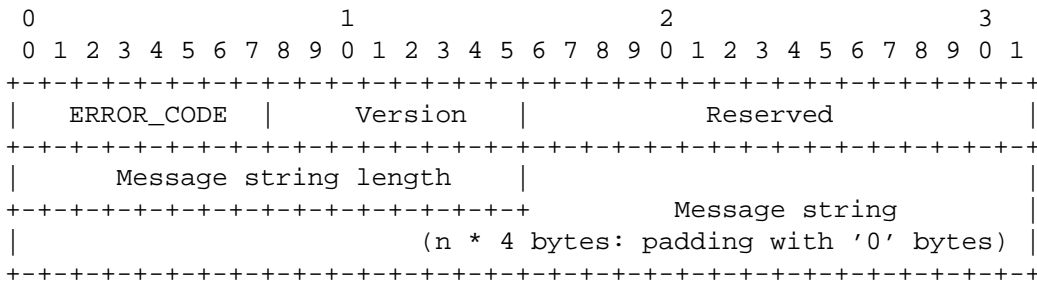


Figure 3.7: Error frame from the WLANServer to the WLANClient if it is not registered

The Security Gateway is assumed to be located in the IP address pointed by the Default Gateway IP setting of the Mobile Node. The reason is obvious: the Security Gateway is, anyway, the default gateway for every Mobile Node.

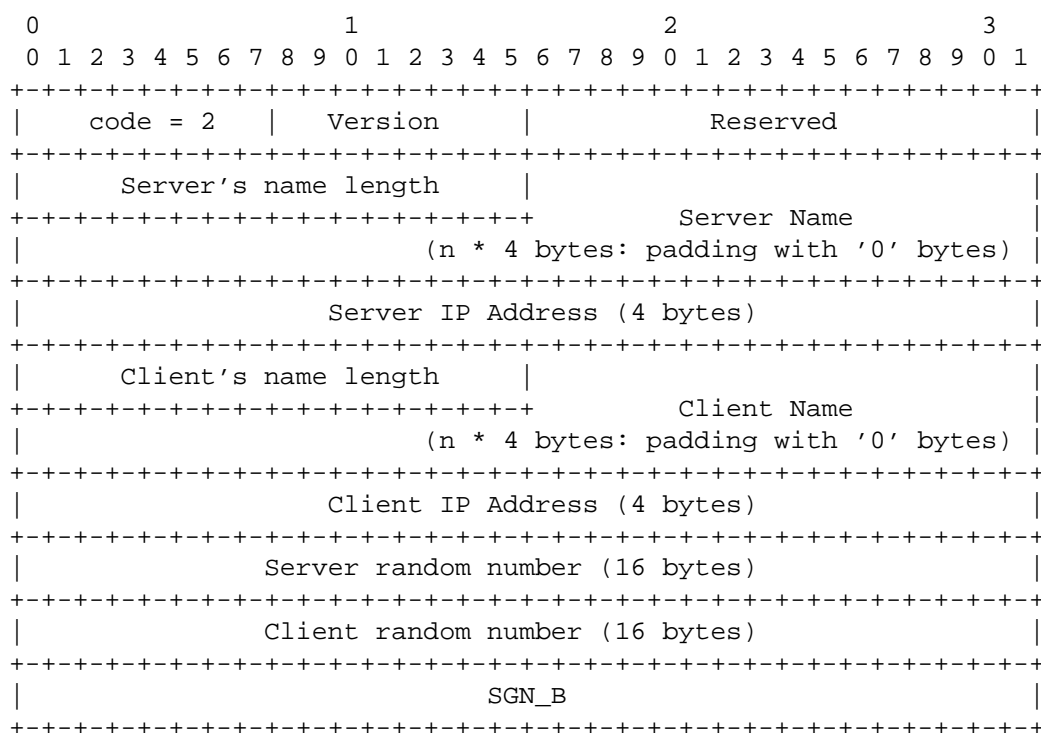


Figure 3.8: Reply of the WLANServer to the WLANClient's initial request

$$MN \rightarrow SG : (1, MN, IP_{MN}, r_{MN}) \quad (3.1)$$

- The Security Gateway's server, WLANServer (see Section 4.3) receives the request. If the Mobile Node is not registered in the Security Gateway's database, it sends back an error frame (Figure 3.7). If the Mobile Node is registered in the Security Gateway, the WLANServer goes on with the protocol. Otherwise, the WLANServer goes on with the protocol. As a reply, it sends a packet to the WLANClient with the following information: (Figure 3.8):

- The Security Gateway's name, SG (which is the Security Domain's name)
- The Security Gateway's IP address IP_{SG}
- The Mobile Node's name MN
- The Mobile Node's IP address IP_{MN}
- a random number generated by the WLANServer r_{SG}
- the random number generated by the WLANClient r_{MN}
- A HMAC signature of all this information. SGN_{SG}

This HMAC uses, among other things, the pre-shared secret between the Security Gateway and the Mobile Node. This pre-shared secret has the same function as a Mobile Node's password (machine-based authentication) in the Security Domain.

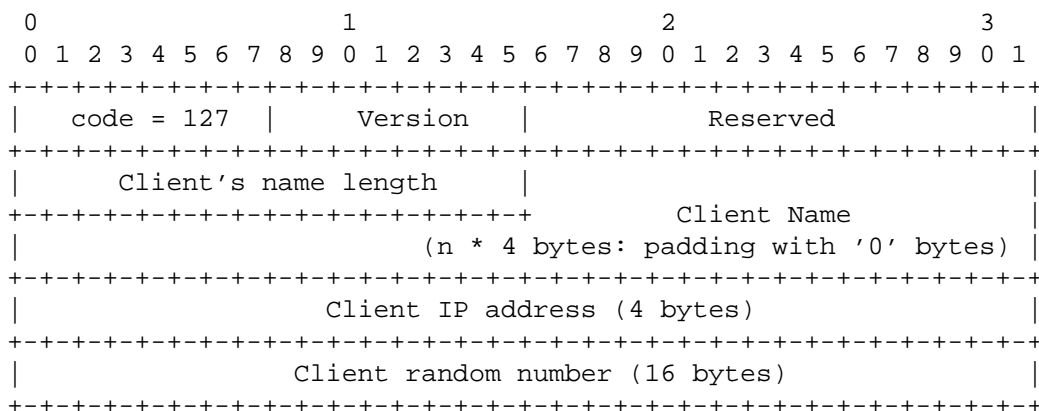


Figure 3.9: Abort frame sent by the client if it is not registered in the Security Gateway

$$SG \rightarrow MN : (2, SG, IP_{SG}, MN, IP_{MN}, r_{SG}, r_{MN}, SGN_{SG}) \quad (3.2)$$

- The WLANClient application receives the reply. If the Security Gateway is not registered in its database, it sends an “abort” message (Figure 3.9). Otherwise, it reproduces itself the HMAC signature over the packet information (it can do so because it also has the pre-shared secret that was used by the WLANServer to sign the frame). If its signature matches the signature attached in the packet, the information is assumed to be authentic (the entity that sent the reply necessarily knows the pre-shared secret between the Mobile Node and the Security Gateway). Only the Security Gateway is able to produce a correct signature since only it knows this pre-shared secret. Hence, if the signature is valid, the peer entity is assumed to be the legitimate Security Gateway.

If the signature was not authentic, the frame is dropped. Otherwise, the WLANClient also sends a confirmation message to the WLANServer. This confirmation states that the Mobile Node accepts the identity of the Security Gateway. Basically, it contains the same information as the WLANServer’s reply, but signed by the WLANClient (Figure 3.10).

$$MN \rightarrow SG : (3, MN, IP_{MN}, SG, IP_{SG}, r_{MN}, r_{SG}, SGN_{MN}) \quad (3.3)$$

- As the WLANServer receives the confirmation from the WLANClient, it does the same as its counterpart: it reproduces itself the HMAC signature over the packet information. If its signature matches the signature attached in the packet, the information is assumed to be authentic (the entity that sent the reply knows the pre-shared secret between the Mobile Node and the Security Gateway). Apart from the Security Gateway, no other entity knows the pre-shared secret. Hence, only the Mobile Node is able to produce a correct signature. If the signature is valid, the peer entity is assumed to be the Mobile Node.

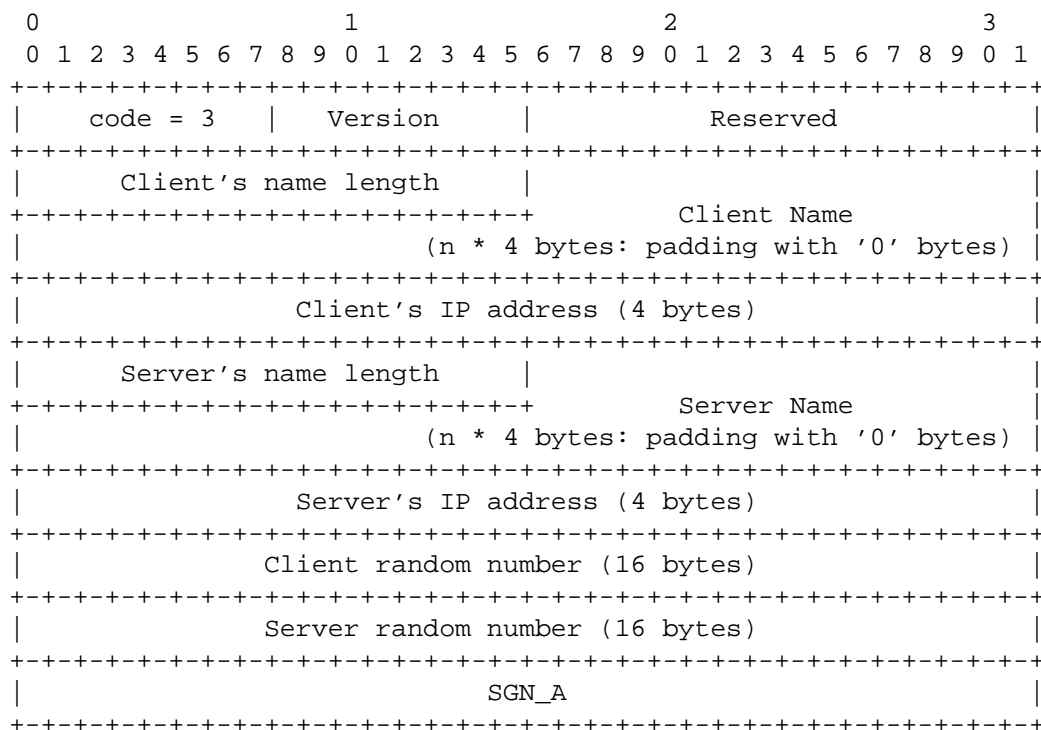


Figure 3.10: Confirmation of the WLANClient to the WLANServer

If the node's signature is not authentic, the packet is dropped (a request store cleaning is performed each time a frame is received, in order to delete old requests, see Section 4.3). Otherwise, the WLANServer updates its IPsec policy, adding rules for an encrypted tunnel allowing the normal traffic between the Mobile Node and any other IP address to flow only through the Security Gateway. This tunnel uses Preshared Key (SK) authentication.

The Preshared Key (SK) for the IPsec tunnel authentication is dynamically produced by the WLANServer. It is the result of an HMAC using the pre-shared secret between the Security Gateway and the Mobile Node (the node's password for the present Security Domain) as key and the random numbers generated by the Mobile Node and the Security Gateway and some constant value, which is fixed for every negotiation and part of the protocol specification, as data.

Finally, the WLANServer entity sends the WLANClient's confirmation (code = 3) back to it with code = 4. This packet does not need to be signed, since both parties have already been mutually authenticated. It does not provide additional information that needs to be signed in order to check the sender's identity. It just acknowledges that the WLANServer has already updated the IPsec policy in the Security Gateway.

$$SG \rightarrow MN : (4, MN, IP_{MN}, SG, IP_{SG}, r_{MN}, r_{SG}) \quad (3.4)$$

5. Upon receipt of this packet, the WLANClient updates its IPsec settings. The settings include the rules for an encrypted tunnel allowing the normal traffic between the Mobile Node and any other IP address to flow only through the Security Gateway. This rule uses Preshared Key (SK) authentication. These rules are established between the Mobile Node and the Security Gateway, so the IPsec protection of the communications only covers this segment. The Preshared Key (SK) for the IPsec authentication is dynamically produced, following the same procedure as that explained above for the Security Gateway. The routing table of the Mobile Node is updated so that it sends all its traffic through the Security Gateway.

Finally, the WLANClient entity launches a ping to the Security Gateway, in order to trigger the IPsec association setup between the Security Gateway and the Mobile Node.

This handshake protocol is suited to be run over an untrusted medium, since the preshared secret between the Mobile Node and the Security Gateway is never sent over the wireless link. However, the pre-shared secret itself (the node's password) can become a vulnerability: the simpler it is, the weaker the IPsec tunnel is. That is why users should choose long, difficult to figure out passwords¹².

3.5.1 Security analysis of the dynamic IPsec policy negotiation protocol.

Need for mutual authentication through the signed messages It could be considered to simplify the above mentioned authentication and IPsec policy negotiation protocol, by removing the mutual authentication. If only the Security Gateway and each Mobile Node know their shared secret, why to perform an authentication step? No authentication should be required since the IKE Preshared Key does not necessarily have to be dynamically negotiated (both peers have a preshared secret from which it is possible to derive the IKE Preshared Key). Only registered users would be able to build legitimate tunnels with the Security Gateway, since only they can produce the necessary IKE Preshared Key. The only information that the Security Gateway would need is the pair Mobile Node's name - present Mobile Node's IP address for each incoming Mobile Node. All the Mobile Nodes would need to know is in which Security Domain they have roamed.

This approach results in a lighter message exchange: when Mobile Nodes roamed into a new Security Domain (in which they were already registered), they would just need to send an unauthenticated request to the Security Gateway, who would update its IPsec policy considering the Mobile Node's name and IP address, and deriving the IKE Preshared Key from the preshared secret saved in the Security Domain local users' file. It would then send an unauthenticated reply to the Mobile Node stating its identity and acknowledging that it has updated its IPsec policy. From the Security Gateway's identity, the Mobile Node would retrieve the corresponding preshared secret and derive the Preshared Key from its local file, and update its IPsec policy.

In that design, the protocol is just meant to let the Security Gateway know the identity of the arriving Mobile Nodes (so that it can dynamically update its IPsec policy to build a tunnel using the adequate IKE Preshared Key with them) and to let the Mobile Nodes in which Security Domain they are in (so that they can dynamically update their IPsec policy to build a tunnel using the adequate IKE Preshared

¹²For this reason, it is recommended to use a strong random password generator. More precisely, what is needed is high-entropy passwords. Please note that passwords do not need to be easy to remember since the user will write them only once in his WLANClient configuration file, not every time he runs the WLANClient

Key with the Security Gateway). This implies a series of security weaknesses that advocate for the stronger negotiation (which performs mutual entity authentication) described before:

- **Client DoS:** If the Security Gateway reply messages are not authenticated, an attacker could impersonate the Security Gateway, declaring that Mobile Nodes have entered a different Security Domain from the actual one. Two things could happen: if the users do not know the Security Domain whose identifier is being delivered by the attacker, they would be unable to perform the IPsec configuration. If they know the Security Domain whose identifier is being delivered by the attacker, they would be able to configure their IPsec policy, but they would still be unable to establish the IPsec tunnels with the legitimate Security Gateway, since the Preshared Key they have entered in their IPsec policies corresponds to another Security Domain.
- **Security Gateway DoS 1:** a Mobile Node is identified in a Security Domain by its hostname. To configure its IPsec policy, the Security Gateway needs also the present IP address of the requesting Mobile Node in the Security Domain (which is a dynamic information) and the corresponding Preshared Key (which is derived from static data stored in a file). An attacker could issue malicious requests with arbitrary names and IP addresses. Since no authentication of the messages is performed, this would force the Security Gateway to change the correct PKE Preshared Key of “logged-in” users for other arbitrary users’ Preshared Keys, making it impossible for the legitimate users to make use of the WLAN.
- **Security Gateway DoS 2:** in Windows, the means to automatically (that is, not manually) update the IPsec policy is by using the tool “ipsecpol.exe”. This tool is very computation-demanding and the time needed to add or change rules grows with the number of rules in the active IPsec policy. In the proposed environments, the number of rules in the Security Gateway’s IPsec policy is fairly high. An attacker could flood the Security Gateway with malicious requests. Since no mutual authentication is required, these non-authenticated requests are unconditionally accepted by the Security Gateway, who must update the IPsec policy for every incoming request. This would cause the legitimate WLAN users not to be able to “log in”.

Nature of the preshared secret piece of data So, authentication in both senses has been proven to be necessary. Another question arises: although authentication is necessary, why having a preshared-secret in the participating entities instead of the IKE Preshared Key itself? In fact, it would be possible to store directly the IKE Preshared Keys of the Mobile Nodes with every Security Domain in a plain file in the Mobile Nodes and a plain file in the Security Gateway containing the Preshared Keys of the Security Gateway with every registered Mobile Node. This would avoid managing pre-shared secret information different from the IKE Preshared Keys: the Preshared Key (used for the IPsec’s IKE Phase I authentication step) would be the only used piece of information.

Of course, the possibility is valid, but there is still a need for a preshared piece of information (which can be a pre-shared secret or the IKE Preshared Key itself) that can be used to sign the messages exchanged and to derive the IKE Preshared Key that is used when updating the IPsec policy both at the Mobile Nodes and at the Security Gateway. So the only real difference is that with the approach proposed above, the IKE Preshared Key (which is also used in the IKE authentication) is not used for signing both the policy negotiation messages as well as the IKE frames, but produced dynamically

from the pre-shared secret and some other random nonces, which are different and unpredictable from one protocol run to the next. Since the IPsec policy negotiation protocol and the IKE are separate protocols, it seemed a cleaner approach to derive a different IKE Preshared Key from the pre-shared secret every time that a successful protocol run takes place. This way, two different keys are used for the two different protocols.

Mutual authentication support PKIX (entity certificates) might be used to sign the IPsec policy negotiation protocol messages instead of the proposed pre-shared secrets. Essentially, the mutual authentication would be equally valid as with the preshared secrets. The users, as well as the Security Gateway, would produce an RSA key pair. A WLAN-local Certification Authority (CA) would sign the users' Public Keys, producing WLAN-local user certificates. By using these certificates, the Mobile Nodes and the Security Gateway could also authenticate themselves. However, there is a drawback to this approach: CAs are generally stored under great security conditions. This kind of conditions cannot be expected from home and small to middle-sized environments. Operating a CA under poor security conditions is an unsafe approach.

3.5.2 Random numbers

In order to produce random numbers, a computer uses a PRNG (Pseudo Random Number Generator). A PRNG is a deterministic algorithm that produces a series of numbers whose properties are close to those of a series of purely random numbers. Unfortunately, unlike a real random sequence, a pseudo-random sequence is 100% predictable when its algorithm and seed are known. The seed is a value used by the PRNG to start the sequence.

Thus, to produce "good" (non predictable) random numbers, it is necessary to seed the PRNG with a completely unpredictable number. This number must be generated under strong random conditions.

As stated by Menezes et.al. [25], to produce a random bit sequence it is possible to use the following hardware-based generators:

- Geiger counters
- Thermal noise from a semiconductor
- Frequency instability of a free running oscillator
- sound from a microphone or video input from a camera

Unfortunately, this kind of generators are out of reach for the users of WLANs at home and in small to medium-sized environments. The other possibility is to perform a software-based random bit generation using processes such as:

- The system clock
- elapsed time between keystrokes or mouse movements

- content of input / output buffers
- user input
- operating system values, such as system load and network statistics.

The random numbers used in the negotiation protocol need to be specially unpredictable, since the information derived from them is flowing through the wireless medium as clear text. If it were possible to establish a pattern on their generation, it could be a source of attacks.

Menezes et.al. [25], explain that a well designed software random bit generator should use as many good sources of randomness as available. Furthermore, each source should be sampled and the sampled sequences should be mixed using a mixing function (SHA-1 algorithm over a concatenation of the sampled sequences is specially recommended).

Therefore, the following process is performed:

1. It is requested from the users to type in 100 random keystrokes. The elapsed milliseconds between keystrokes are registered. These numbers are concatenated in an array and hashed with the SHA-1 algorithm. The hash output is saved in a file.
2. As the protocol entities are started, they retrieve the system time (seconds since UTC 1/1/70 and milliseconds), concatenate this data and hash it with SHA-1.
3. They retrieve the number of running processes in the system at that moment.
4. They also retrieve information about the total memory usage (classified in 6 different categories).
5. A variation of the SHA-1 function is used (inspired in a PRNG recommended by Menezes et.al. [25, Algorithm 9.53], and depicted in Figure 3.11):
 - As CV_0 , the keystrokes hash is used (20 bytes)
 - The memory statistics, the number of processes and the time hash are concatenated and padded with CV_0 to make Y_0 .
 - Both arrays (Y_0 as data and CV_0 as initialization vector) are introduced in the SHA-1's "f" compression function, obtaining CV_1 . From it, the first 10 bytes are used as random bytes.
 - Y_1 is obtained by rotating Y_0 20 bytes to the left and XORing it with a hash (also SHA-1) of Y_0 and the newly generated CV_1 (as chaining value) in the right-side 20 bytes. In the case of the i_{th} iteration, Y_i is obtained by rotating Y_{i-1} 20 bytes to the left and XORing it with a hash (also SHA-1) of Y_0 and the newly generated CV_{i-1} (as chaining value) in the right-side 20 bytes.
 - This procedure is repeated as many times as necessary in order to produce the desired number of random bits. Each iteration produces 10 random bytes (80 random bits).

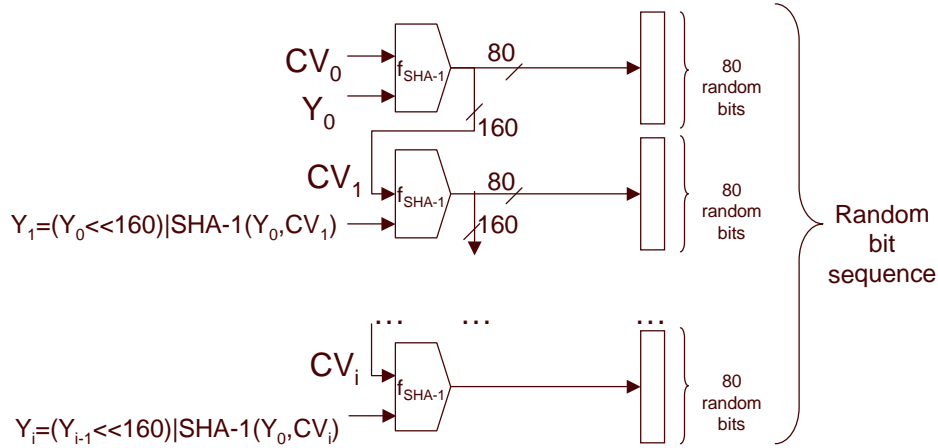


Figure 3.11: Random generation scheme.

6. The initializing arrays, CV_0 and Y_0 are produced only once: the first time a random number is generated. Afterwards, the process uses the last CV_i and Y_i produced as starting point for the random number generation explained above each time a new random number is needed.

$$CV_0 = \{SHA-1(keystrokes)\} \quad (3.5)$$

$$Y_0 = \{memory_data \mid \#processes \mid SHA-1(time) \mid padding(CV_0)\} \quad (3.6)$$

$$CV_i = \{f_{SHA-1}(CV_{i-1}, Y_{i-1})\} \quad (3.7)$$

$$Y_i = \{(Y_{i-1} * 2^{160}) \bmod 2^{512}\} \oplus f_{SHA-1}(Y_0, CV_i) \quad (3.8)$$

Through the previously described process, it is expectable to obtain relatively high entropy, although no further analysis has been performed in order to check this hypothesis' validity.

3.5.3 Security of the signatures: HMAC

The packet origin authentication and data integrity are implemented in this protocol through signatures over the frames' data by using the HMAC construction. HMAC is a hash function authentication code.

The underlying idea is to construct a MAC (Message Authentication Code) from an MDC (Modification Detection Code). First, an MDC of the frame's data is generated using a "secret key" (which in our case is the pre-shared secret between the Security Gateway and each Mobile Node). Next, a MAC is constructed out of the resulting MDC using the same "secret key" as before. The most used construction is:

$$HMAC = H(K, p1, H(K, p2, m)) \quad (3.9)$$

Table 3.3: Notation of the HMAC construct

Notation	Meaning
H	Hash function (MD5 or SHA-1)
K	Secret key used for the Hash functions
p_1	Outer padding
p_2	Inner padding
m	Plaintext to be signed

p_1 and p_2 are padding patterns used to fill up the key to one input block of the cryptographic hash function. This scheme seems to be secure [25, Note 9.67] and is documented in RFC 2104 [23].

Under normal circumstances, only the Security Gateway and each Mobile Node know the individual pre-shared secrets which serve as a “secret key” for this process. This means that only the Security Gateway and each Mobile Node are capable of producing such valid HMAC signatures. This structure guarantees both the origin authentication and the data integrity.

Chapter 4

Implementation and evaluation

The protocol described in Section 3.5 implies two software entities: a server and a client. However, the need for more applications to solve needs such as “true” random number generation or unique Security Domain identifiers brings along a number of additional programs.

The actual implementation of the whole bundle of applications will be discussed in the following sections.

4.1 Tools used in the implementation

For the implementation of the programs that will be next described, and which constitute the result of this project, some tools and technologies have been used.

C++ (Visual C++ 6.0) was chosen as programming language.

For network traffic monitoring and analysis, “windump” has been used. It is the port to Windows of the popular “tcpdump” tool. For some tests in relation with the authentication capabilities using certificates of the Windows 2000 IPsec implementation, a port of OpenSSL to Windows was utilized.

In order to implement some scripts (including test scripts) the WSH technology was chosen: it allowed to construct both simple GUI’s and wizards for the interaction with the user and powerful testing scripts.

4.2 The Mobile Nodes’ clients: WLANClient

WLANClient.exe is the client application that takes part in the IPsec policy negotiation protocol. It runs on the Mobile Nodes and starts the communication with the server entity (WLANServer) that is running on the Security Gateway.

As explained in section 3.5, it sends an initial request frame (code = 1) to the WLANServer. It assumes that the WLANServer is located in the Default Gateway IP address that has been configured in its IP settings. WLANClient identifies the Mobile Node by its host name and uses the IP address

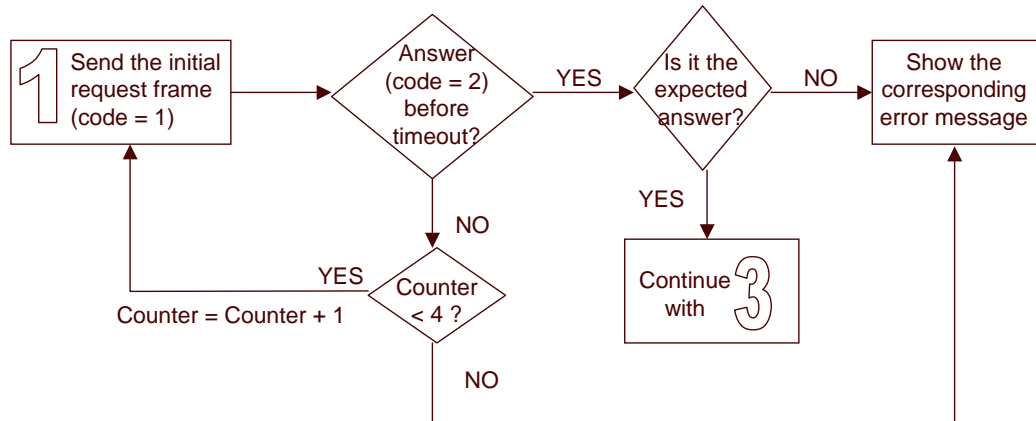


Figure 4.1: First step of the WLANClient process: sending the initial request

which has been assigned to him through DHCP or manual setting. After sending this first packet, it launches a timer while it awaits the WLANServer’s reply packet. If the timeout is exceeded, the initial request packet will be sent again. This replay procedure is repeated up to 4 times before the WLANClient gives up and assumes that the Security Gateway is not running or unreachable. This can be viewed in Figure 4.1. The replay mechanism is present whenever the client must send a frame to the server, although it has only been included in the first diagram in order to make Figure 4.2 simpler.

Normally, the reply from the WLANServer would reach the WLANClient. If the frame’s code is 2 (that is, the WLANserver’s reply to the WLANClient’s initial request) and it contains the expected information, the client goes on with the protocol. The expected information means the same Mobile Node’s name, the same Mobile Node’s IP address and the same client’s random number that the WLANClient proposed to the WLANServer in the initial request frame. If these data do not coincide, the frame is considered to be wrong, and is ignored (dropped).

The next step is to search in the Mobile Node’s database to see if the Security Gateway that has sent the reply is supported (that is, if there is a pre-shared secret established between it and the Mobile Node). If the Security Gateway is not registered in the Mobile Node’s database, the Mobile Node sends an “abort” frame (code = 127).

If the server is registered, the WLANClient extracts the information contained in the WLANServer’s reply, and verifies the integrity check (HMAC). If it is incorrect, the packet is dropped. If the HMAC is correct (the frame is authentic), the client prepares the confirmation frame (code = 3) and sends it to the WLANServer. Before it sends the confirmation frame (code = 3), it must check that its name in the present Security Domain is the same as its host name. In a normal case, no user name collision will take place and the frame will be sent. If there is a client name collision, it must send an abort frame (to let the server know that the negotiation is invalid) and a new request (code = 1) with his local name for this Security Domain, beginning the protocol once again from the beginning (this has not been included in Figure 4.2 in order to make it simpler).

If the WLANServer returns an error frame (code = ERRORCODE) instead of a normal reply, the WLANClient shows the error message contained in the frame and exits.

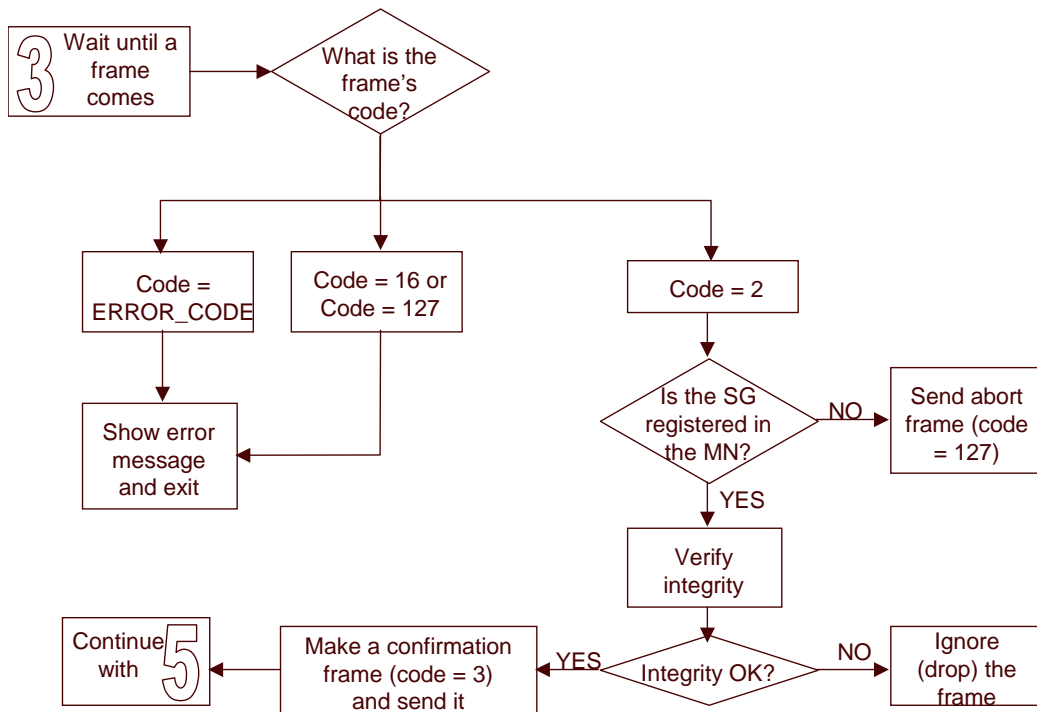


Figure 4.2: Second step of the WLANClient process: processing the WLANServer's reply

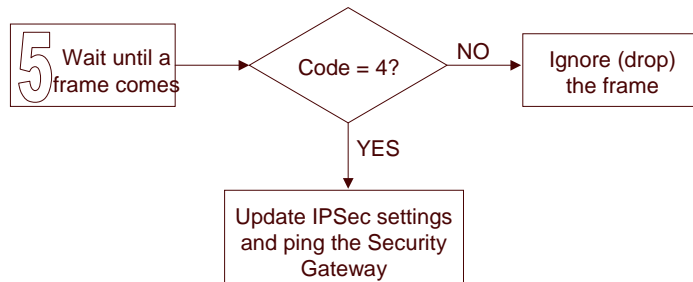


Figure 4.3: Third step of the WLANClient process: ping the Security Gateway

After this step, it just awaits the final WLANServer message, which is not authenticated. When it receives the WLANServer's final message, it updates the Mobile Node's IPsec policy and launches an initial ping over the Security Gateway, which triggers the creation of the IPsec association. Finally, it exits notifying the user that the IPsec tunnel is ready.

The application accepts a series of command line arguments:

- `-f filename`: filename is the file where some additional ipsecpol¹ commands have been introduced by the users. This commands are intended to make the users' default IPsec settings

¹ipsecpol.exe is a freely downloadable tool from the Microsoft Resource Kit aimed at the automatic configuration of IPsec policies.

compatible with the new policy (see Section 3.3.6). It is only necessary that in the ipsecpol commands of this file the IPSec policy name coincides with the default name of the policies created by the WLANServer and WLANClient programs². The whole path of the file is expected. This argument is optional.

- -p port: port is the UDP port that must be used by the WLANClient. By default, it is 58612. However, this port might be already in use in some environments by other applications. In this case, this option allows the users decide which port to use³. This argument is optional.

4.3 The Security Gateway's server: WLANServer

WLANServer is the server application that takes part in the IPSec tunnel negotiation protocol. It runs on the Security Gateway and awaits the requests of the client entities (WLANClient) running on the Mobile Nodes.

As explained in section 3.5, it is continuously waiting for frames from the WLAN clients. In order to achieve a better understanding, a diagram on the server's behaviour is presented in Figure 4.4.

The usual sequence of events is the following. First, a request frame (code = 1) arrives at the WLANServer from some WLANClient entity. The WLANServer would normally store the data contained in the request in a "request profile" object, which is stored in a vector called the "request profile pool" (read more about this in Section 4.3.1). Before doing this, it performs a cleanup of the old requests stored in the "request profile pool" (see Section 4.3.1). Afterwards, it would send a reply frame (code = 2) back. There are some exceptions to this behaviour, though.

If the request has already been received in the immediate past time (and no further protocol frames have been received from the corresponding WLANClient), the request profile should be still in the "request profile pool". If it is, the WLANServer just updates its timestamp and sends a reply frame to the WLANClient. If the node is not at all registered, the request will be not be stored either. Rather, a "user is not registered" frame (code = 16) is sent to the WLANClient.

If the WLANServer sent a reply frame to that client, it is expecting now a confirmation from the corresponding WLANClient entity. If it arrives (code = 3), it searches for the corresponding profile and extracts it from the pool vector. It checks that the information contained in the request profile stored in the pool and the information contained in the received frame (code = 3) matches. That is: both names (Security Gateway and Mobile Node), both random numbers (WLANClient's and WLANServer's) and both IP addresses (WLANClient's and WLANServer's) must be equal. If they are not, the frame is ignored (dropped).

If the data matches, it configures dynamically its IPSec policy, and eventually sends a final confirmation frame (code = 4) to the WLANClient, in order to let him know that the Security Gateway is ready to accept the Mobile Node's packets through an IPSec tunnel.

The remaining possibility is that an abort frame arrives (code = 127). This will usually happen

²The name is "SECURE_WLAN"

³It is very important to make sure that the WLANServer is also running on that port. Otherwise, it will be unreachable by the WLANClient.

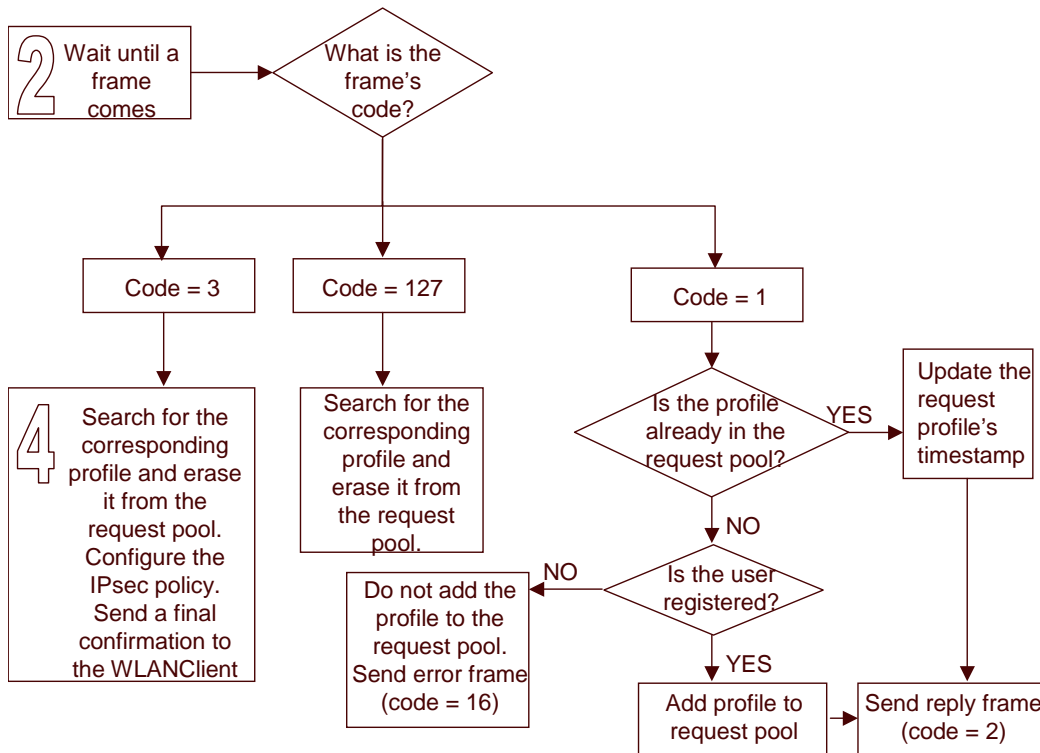


Figure 4.4: Behaviour of the WLANServer

because the WLANClient does not know the Security Gateway (in its Security Gateway database, there is no entry for this Security Domain). If this is the case, the WLANServer will look in the “request profile pool” for the previously saved request profile, and erase it. Any other frame codes are ignored (dropped).

The application accepts a series of command line arguments:

- `-ip ipaddress`: `ipaddress` is the IP address of the IP interface on which the Security Gateway is listening for the incoming clients' requests. This argument is compulsory.
- `-f filename`: `filename` is the file where some additional ipsecpol commands have been introduced by the users. These commands are intended to make the users' default IPsec settings compatible with the new policy (see Section 3.3.6). It is only necessary that in the ipsecpol commands of this file, the IPsec policy name coincides with the default name of the policies created by the WLANServer and WLANClient programs. The whole path of the file is expected. This argument is optional.
- `-p port`: `port` is the UDP port that must be used by the WLANServer. By default, it is 58612. However, this port might be already in use in some environments by other applications. In this case, this option allows the administrator to decide which port to use. This argument is optional.

4.3.1 The request profile pool

The request profile pool is a vector in which incoming requests' profiles from the WLANClient entities are stored in the WLANServer. A request profile is an object containing the following information:

- Timestamp: stores the time at which the request was received by the WLANServer.
- IP address: of the client
- Random number proposed by the WLANclient to the WLANServer
- Random number proposed by the WLANServer to the WLANClient
- Name: of the Mobile Node machine

The concepts of “request profiles” and “request profile pool” have been introduced to give the WLANServer the ability to cope with multiple concurrent protocol negotiations. It can be viewed as the “state” of the WLANServer (which, in turn, becomes a “stateful machine”). This way, the order in which frames from different clients arrive is irrelevant to the WLANServer.

Each time a frame arrives, the request profiles pool is revised by the WLANServer. Profiles whose timestamp is older than a certain number of seconds are deleted, since the communication with those clients can be considered to have been aborted. This prevents the vector from growing indefinitely large for long periods of service.

4.4 The databases on the participating entities

The Security Gateway needs to know which nodes are allowed in its Security Domain and what is the pre-shared secret with each of them. This information is saved in a file named “server.conf”. In it, there is one entry for each registered client. Each entry consists of two values: the Mobile Node's name (in fact, the node's hostname) and the pre-shared secret with that client.

Each Mobile Node also needs to know in which Security Domains (that is, Security Gateways) it is registered and what is the pre-shared secret between itself and those Security Gateways. This information is saved in a file named “client.conf”. It contains one entry for each Security Domain, and each entry consists of two or three values, depending on if there is a client's name collision or not.

If there is no client's name collision, the entries consist of two values: the Security Domain's name (Security Gateway's name) and the pre-shared secret between the Mobile Node and the individual Security Gateways.

If there is a name collision, another client is using the Mobile Node's host name. In this case, the WLANClient needs to know how it is called in this particular Security Domain (that is, the alternative client's name). This is the third value in case of a client's name collision.

4.5 Complementary applications

Up to now, only the entities that take part in the IPsec tunnel negotiation protocol have been described. However, some initial configuration steps are needed in order to prepare the machines to run both the WLANServer and the WLANClient programs. Also, some support is also needed in order to switch off the effects of these programs over the IPsec settings of the implicated workstations.

These additional applications will be described next.

4.5.1 Security Gateway IPsec initial configurator

As explained in Section 3.3.3, some previous configuration is needed in the Security Gateway. Initially, the IPsec policy in the Security Gateway would allow all non-registered entities to hold communications in the WLAN, since no rule in the policy is against it. Thus, a non-registered node could just come up and start communicating with the registered Mobile Nodes or the wired network through the Security Gateway.

The InitialIPsecConfigurator.exe program avoids exactly this problem: it “blocks” all the IP addresses of the WLAN subnet in the Security Gateway’s IPsec policy. It also makes the blocking rule compatible with the DHCP and tunnel negotiation protocols (by allowing them). It must be run only in the Security Gateway. It adds the following rules to the policy:

- A “block” rule to all IP addresses in the subnet
- A “permit” rule to all the IP addresses in the subnet ONLY for UPD traffic that goes to / from the policy negotiation protocol port (which is 58612 by default).
- A “permit” rule to all the IP addresses in the subnet ONLY for DHCP traffic.

There are some exceptions to this blocking process. The IP address of the Security Gateway will not be blocked (because a blocking tunnel between an IP address and itself makes no sense).

The Base Stations’ IP addresses may also remain unblocked (in fact, an “allow” filter may be set for them in the Security Gateway’s IPsec policy) because their addresses will, in theory, not be used by clients: they are just used for Base Station configuration through SNMP. The program offers the option to initially block them or not. Whenever it is needed, it is possible to unblock them for configuration. Afterwards, they can be blocked again.

This can be accomplished quite easily: to initially block a certain Base Station’s IP address, the administrator would run the program without entering the wished IP address in the file where such addresses are listed (view command-line arguments below). When some Base Station configuration was needed, he would enter the wished IP address in the “allowed IP addresses” file and run the program from that address to that same address (min and max values would coincide with the IP address’s least significant byte, see command-line arguments below). This way, the address would be unblocked. After the configuration of the Base Station had been completed, the Base Station IP address would be erased from the “allowed IP addresses” file, and this program would be run again

(min and max values would coincide with the IP address's least significant byte, see command-line arguments below).

This prevents unregistered clients from “hijacking” the Base Stations' IP addresses in order to perform an attack over the WLAN.

This program must be run at least once, when installing the WLANServer software. Nevertheless, as explained above, it can be further run if needed to block / unblock certain IP addresses or IP address ranges of the WLAN.

It accepts a series of command-line arguments:

- `-interface ip:port`: after the “-interface” tag, a value for the IP interface of the WLANServer must be entered. This argument is compulsory, and it consists of the IP address and the UDP port on which the Security Gateway listens for the incoming clients' requests, separated by a colon (“:”) character. By default, the UDP port used by WLANServer and WLANClient is 58612, so it is the value which should normally be entered here. However, in some cases this port may be already in use and another port needs to be used (and specified in this command line argument).
- `-f file`: after the “-f” tag, a file name can be entered. In this file, the IP addresses that do not have to be blocked (or those that must be unblocked) are listed (one per line). These addresses typically would correspond to those of the Base Stations. This argument is optional.
- `-min min`: after the “-min” tag, an integer value between 1 and 255 must be entered. This is the lowest IP address which will be blocked. For example, if the WLAN subnet is 172.12.1.*, the blocking loop will begin in 172.12.1.min. This argument is optional, but if it is entered, then the `-max` argument is compulsory.
- `-max max`: after the “-max” tag, an integer value between 1 and 255 must be entered. This is the highest IP address which will be blocked. For example, if the WLAN subnet is 172.12.1.*, the blocking loop will end in 172.12.1.max. This argument is optional, but if it is entered, then the `-min` argument is compulsory, and it must be smaller than the `-max` value.

If the Security Gateway administrator wishes to remove the IPSec policy corresponding to the WLANServer, the application accepts the command line argument “remove” (`InitialIPSecConfigurator remove`). This call deletes the corresponding IPSec policy.

4.5.2 Random initializer

This program (`RandomInit.exe`) performs the first step of the random number generation process of the system, described in Section 3.5.2. It is requested from the users to type in 100 keystrokes (random keys and times between keystrokes).

The elapsed milliseconds between one keystroke and the next are saved and concatenated in an array. Later, this array is hashed with the SHA-1 algorithm. The output bytes of the hash function are saved in a file called “`wlan.seed`”. The contents of this file constitute one of the random sources which are used to seed the PRNG of both the WLANServer and the WLANClient (and therefore is part of both

software bundles). It is strongly recommended to repeat this procedure from time to time, since it refreshes the random source of the WLANClient and WLANServer, thus enforcing the security of the protocol.

It takes no command-line arguments.

4.5.3 Security Gateway identifier configurator

In Section 3.3.5, the need for unique Security Gateway identifiers was introduced. Mobile Nodes accessing the same WLAN may have the same hostname (which is by default also the client's name). This problem is solved by allowing them to use an alternative name for that single Security Domain. But if also Security Gateways are allowed to use their plain hostname as Security Gateway identifier, a double name-collision could occur, complicating excessively the negotiation.

To prevent this from happening, a few random characters are added to the Security Gateway's hostname. These random characters are produced using the same system as the one introduced in the RandomInit.exe application (Section 4.5.2). The Security Gateway administrator is asked to enter 100 random keystrokes, and these are used to produce a series of hashed bytes. The first 4 hash bytes are appended to the Security Gateway's hostname in hexadecimal format (2 characters per byte), resulting in 8 added bytes. Between the hostname and these bytes, an underscore character is placed. The resulting string is called the Security Gateway's identifier (also Security Domain's identifier) and it is saved in the file "securitygateway.id".

When the WLANServer runs, it replies to the WLANClient entities identifying itself through this string. Additionally, this is the Security Gateway's name that must be present in the entries of the Mobile Nodes' databases containing data of the supported Security Domains (or Security Gateways). More information on these databases can be found in Section 4.4.

This procedure reduces to an acceptable minimum the possibility of a Security Gateway's name's collision. It should be performed only once, during the installation of the WLANServer software. The reason is the following: if the Security Gateway changes its identifier, every user having his node registered within it will have to change his Security Gateway database (client.conf), since it is how they recognize the different Security Domains. Depending on the number of registered users, this might become a lot of work and a nuisance for the users.

This program takes no command line arguments.

4.5.4 Client WLAN disconnecter

As the users leave a certain WLAN, they might want to switch off the IPsec policy that was activated to build a tunnel to the Security Gateway of that Security Domain.

This application, WLANDisconnect.exe, deactivates the previous IPsec policy (that is, the active policy for the last visited Security Domain) and activates a "default" IPsec policy that, essentially allowing what the user decides. In this sense, there are two possibilities:

1. "Allow everything" policy: if no command line arguments are entered, the program activates

a default IPsec policy that allows the exchange of traffic with any other entity (any other IP address) without applying security measures.

2. “Allow what the user decides” policy: of course, more conservative users would prefer to limit that default policy. This is accomplished by entering as a command line argument a file that contains additional rules to be added to the “default” IPsec policy. It must be an ipsecpol-compliant file with the additional ipsecpol commands to be executed in order to tune the “default” IPsec policy according to the user’s specific security needs. Users must just be careful to design this file bearing in mind that the “default” IPsec policy is named “DEFAULT_WLAN” (the ipsecpol commands should add the extra rules to this policy).

This program takes one command line argument:

- -f filename: where filename is the ipsecpol-compliant file containing the extra rules to be added to the “default” IPsec policy (“DEFAULT_WLAN”). This parameter is optional. If is not entered, the activated policy will be of the type “Allow everything” and if it is entered, the policy will be of the type “Allow what the user decides”.

However, the program performs no control over the contents of this file: it is the user’s responsibility to enter valid ipsecpol commands that make sense and that result in a viable IPsec policy.

4.5.5 WLAN Service installer

It is very convenient to install the WLANServer as an NT service in the Security Gateway. This allows to run the WLANServer even if no node is logged in the Security Gateway. In order to install a program as an NT service, it is necessary to use two tools called INSTSRV and SRVANY and then change the Windows registry manually. Users are prevented from using these tools and manually updating the Registry (which is an error prone practice) with the tool WLANService.exe.

It installs the WLANServer.exe as an NT service, so that when the Security Gateway is turned on, WLANServer starts running, although no user is logged in. Users must be careful to change the account on which the service is running once the WLANServer is installed as service, in order to restrict the permissions of the process. It can be found under the name “WLANServer”. The service startup must also be configured (at startup, at logon, etc.). This can be accomplished in the folder Settings/Control_Panel/Administrative_Tools/Services.

The application accepts a series of command line arguments:

- -ip ipaddress: ipaddress is the IP address of the IP interface on which the Security Gateway is listening for the incoming clients’ requests. This argument is compulsory.
- -f filename: filename is the file where some additional ipsecpol commands have been introduced by the users. This commands are intended to make the users’ default IPsec settings compatible with the new policy (see Section 3.3.6). It is only necessary that in the ipsecpol commands of this file, the IPsec policy name coincides with the default name of the policies created by the WLANServer and WLANClient programs. The whole path of the file is expected. This argument is optional.

- -p port: port is the UDP port that must be used by the WLANServer. By default, it is 58612. However, this port might be already in use in some environments by other applications. In this case, this option allows the administrator decide which port to use. This argument is optional.

If the Security Gateway administrator wishes to remove the WLANServer as an NT service, the application accepts the command line argument “remove” (WLANService remove). This call deletes the corresponding Windows registry’s branch, uninstalling the WLANServer as NT service.

Chapter 5

Conclusion and Outlook

Considering the project as a whole, it is possible to draw some conclusions about the WLAN security and the ideal configuration software features to cope with it. Some of them have been achieved, and some remain beyond the scope of this project.

In this chapter, the configuration solution's functionality limitations (and possible improvements) are reviewed. An analysis of the security limitations is also made.

Eventually, an outlook on the present WLAN landscape is performed, in order to classify this project's solution.

5.1 Limitations of the solution and possible extensions

There remain some open issues, which were left out of the scope of this project. The most relevant will be discussed next.

5.1.1 Adaptability

The pieces of software generated work well if the environment is set up exactly as it was proposed under our "typical environment" assumptions. However, these are just models that generalize the typical and common features of the particular cases. In some environments, the network topology might be slightly different. In others, the users might have special requirements that have not been included in this prototype's capabilities.

The adaptation of this software to the particular users' needs may become a great task depending on how much their environments differ from the ones we figured out as starting point for the project.

Although efforts have been made for the programs to be as configurable as possible, many of the options may be quite cryptic for the average user. For example, an employee in an enterprise does generally not know about adding custom IPsec rules or about editing an ipsecpol-compliant command file and entering it as command-line argument for the programs.

Table 5.1: Notation of the scalability function

Notation	Meaning
w	Number of supported C-type IP subnets by the Security Gateway
c_n	number of configured IP addresses (at least once since installation) in the n th. WLAN

He does not know about UDP ports or IP visibility either. This complexity was to be kept transparent to him. On the other hand, the simpler a program is to the users, the more complicated it becomes. So, a tradeoff between user-friendliness and complexity is always present. In this case, a reasonable position was found in between of both extremes.

5.1.2 GUI

The integration of the different software components into big applications managed through intuitive GUIs would be the next step. A management console with load and performance statistics, configuration menus and client registry would be available for the Security Gateway administrators who need not have great knowledge about networks. The same applies to the client software: a client GUI with the connection settings and the optional ipsecpol commands represented in a more user-friendly way would be nice and comfortable for the majority of users. All this functionality could be easily implemented with some knowledge of GUI programming and software development.

For both sides, an installation wizard that went through all the initial configuration steps in an automatic fashion and providing a GUI for the users would also be highly desirable.

Nevertheless, the system is quite easy to install and deploy as it is now, although its use becomes more complicated if users wish to make use of the “advanced” features.

5.1.3 Scalability

Another question is the scalability of the solution. How many Security Domains can be supported by the same Security Gateway (a Security Domain corresponds to a C-type IP subnet)? In principle as many as wireless interfaces are installed in it (although the workstation’s limitations may set a more restrictive limit). Of course, different installations would be necessary, one for each WLAN: they would need different users’ databases and configuration files, but they would share the IPSec policy. Here, a different problem arises: how many rules can be declared in an IPSec policy without coming across serious performance loss? The more rules, the slower it is to update the policy and, probably, the less optimal the IP forwarding becomes. Furthermore, it has not been investigated how the Windows IPSec implementation deals with several concurrent IPSec policy updates (which is an expectable situation when managing several WLANs).

For each supported Security Domain, a minimal number of about 750 rules and a maximal number of about 1000 rules are added (depending on how many IP addresses have been used at least once) to the IPSec policy. The exact formula is (see Table 5.1):

$$Totalnumberofrules = \sum_{i=1}^w (755 + c_i) \quad (5.1)$$

This shows that the number of rules scales quickly with the number of Security Domains.

5.1.4 Maintenance of the Security Gateway

Also, the support of users is quite rudimentary. In this project, the users' and Security Domains' databases are plain files with a couple of lines for each entry. These files are parsed when the WLANServer and WLANClient applications are started. That implies that if any changes are made in a database (to register new clients in the Security Domain or Security Domains in a Mobile Node), the applications (WLANclient and WLANServer) must be restarted. A good improvement to this drawback would be to enable the WLANServer to listen to incoming signals (similar to the *kill* command in Unix) in order to read and parse again the users' database file and insert the new data into its users' memory vector.

In the case of WLANClient it would not be a big problem, since in its normal operation, it is started each time the Mobile Node enters a Security Domain. But the case of the WLANServer is different. In principle, it is supposed to be running as a Windows service or as a continuous program. It is not very convenient to stop its service every time we want to register a new user.

5.1.5 Portability

A port to other operating systems should be a straightforward task. Only some features of the code should be kept in mind.

Firstly, functions of Windows libraries, such as "winsock" (for the socket programming) and "psapi" (for the system calls in order to get system memory statistics) have been used. All the code fragments using these libraries should be rewritten for the new platforms.

Secondly, the actual way to update the IPSec settings in a Windows is through calls to a tool called "ipsecpol", present in the Microsoft Resource Kit. This tool allows to set up an IPSec policy by calling it a series of times with the appropriate arguments. In other platforms, the way to update the IPSec policy will differ depending on the implementation used and its configuration interface.

It is not clear if all IPSec implementations have inter-platform compatibility. A hypothetical port of this program bundle to other platform might not be compatible with this version, because of IPSec implementation incompatibilities. However, these issues are beyond the scope of this project.

5.2 Remaining security issues

There are some security remaining issues in the proposed security design, which will be analyzed in this section.

5.2.1 Broadcast traffic

IPsec does not protect broadcast or multicast traffic. This could be a security problem, depending on which information flows through the WLAN in broadcast packets. If network security-sensible information is broadcasted or multicasted over the wireless medium, this could be a weak point to the security.

For example, the “arp-spoofing” technique allows attackers to gain knowledge on the adjacent sub-networks. The arp protocol’s traffic is broadcasted, and therefore outside of the IPsec’s protection.

In such networks, no broadcast or multicast real-time applications could be safely (if at all) carried out. This is, obviously, a limitation to the model. Multicast is rarely used in WLAN scenarios, but broadcast is widely used.

It requires further study in order to determine how this could affect the network security in WLAN environments. However, the issue is beyond the scope of this project.

5.2.2 Other IPsec limitations

IPsec cannot be secure if the systems involved are not. If the computers taking part in the IPsec tunnels are broken by an attacker, he would automatically break into the secured WLAN. Nevertheless, this threat is limited: although an attacker took control of a properly configured legitimate member of the Security Domain, he would not be able to impersonate other legitimate Mobile Nodes. He would not be able to eavesdrop on the traffic of the other properly configured legitimate Mobile Nodes (because the keys used for the other tunnels are different). However, he would still be able to communicate with them, with the Security Gateway or with any entity belonging to the wired infrastructure or the Internet.

IPsec’s entity authentication is host-based, and not user-based. This can become a problem if some of the machines involved in the IPsec VPN can be used by different people. In that case, all them are assumed to be trusted, since it is the system (and not the users) that is authenticated.

5.2.3 Unconfigured Mobile Nodes

Users could still enter a protected WLAN and communicate with each other, in an “authorization violation” attack. This is, generally, a minor security problem. It could though become a serious problem if non-configured users make use of too much bandwidth. WLANs have a limited bandwidth, so this non-authorized users might cause it to reach congestion states, and eventually “denial of service” states if the congestion does not allow the legitimate users to hold their communications normally.

As explained in Section 3.1.1, this happens because the access control is performed in the Security Gateway, and not in the Base Stations.

However, these non-legitimate users would be in no case able to communicate with the other legitimate Mobile Nodes, to the Security Gateway or to other entities on the wired network segment beyond the Security Gateway, since their packets would be blocked by the IPsec policies established among the legitimate entities.

5.2.4 Passwords

The security of the WLAN environment is based on IPSec tunnels. Essentially, IPSec is considered to be a secure protocol. Up to now, no serious weaknesses have been found in it. However, an implementation can contain bugs or vulnerabilities that are not present in the abstract conception of the protocol.

The only weakness could consist on choosing too simple, easy-to-guess passwords. The IKE Pre-shared Key used by the IPSec tunnels depends directly on the users' passwords. In this case, other techniques rather than brute-force or analytical methods could be used to break the IPSec security. Hence, user passwords containing high entropy should be used.

The obvious solution is to use a random bit generator in order to assign the passwords to the users. By doing this, the passwords will contain a higher level of entropy and this will make the IPSec links stronger. In fact, the RandomInit.exe application (which is contained in the software bundle, see Section 4.5.2) is a good password generator in this sense. The program provides an option that gives an output of 40 bytes without modifying the random source file (wlan.seed). The whole string, or a subset of it might be used as user passwords.

Unlike in other environments, the passwords need not be entered each time the users log in, but just once when they get registered in a certain Security Domain. This allows long, entropy-rich passwords to be used instead of user-chosen (and, hence, highly predictable) passwords.

Since the RandomInit.exe application is shipped with both installers (server and client), any of them could produce the user passwords.

5.2.5 Responsibility of the users

This solution relies on the users' responsibility to build an IPsec tunnel each time they enter a WLAN protected with this system. If they do not, their Mobile Nodes are accessible to attackers and the data they send or receive is open to eavesdropping. However, this risk is quite reduced, since users would most probably notice they have forgotten to secure their devices: they would have no access to the other Mobile Nodes, the Internet, etc.

To avoid users from forgetting to run their WLANClient applications, the WLANClient could be set in the Startup folder of every user. This would launch the WLANClient after user's login.

The correct configuration of the Security Gateway is another important item. If some of the IP addresses in the subnet are left unblocked, this would be an easy target. Also, the unprotected Base Station IP addresses could be used by attackers in order to enter the protected zone.

Therefore, it is recommended to block all addresses in the subnet, including those of the Base Stations. If some configuration needs to be performed on them, their addresses should be unblocked, the configuration made and again blocked.

These issues also place a high responsibility on the Security Domain's administrator.

5.3 Tests and results

The tests described below were performed in the development environment in which the programs were developed:

- Two Dell workstations with Pentium III processors and 512 M of RAM were used as Security Gateway and Mobile Node. For some tests, a Dell wireless-enabled notebook was also used as Mobile Node.
- The Base Station was an Apple Airport
- The wireless cards were Lucent Orinoco network adapters.

5.3.1 Test 1

Many applications have the problem of causing memory leaks throughout their code. Normally, these problems are not very bad if the application does not need to run for long periods of time. If they do, the accumulated memory leaks could lead to a stack overflow or unacceptable memory losses.

This test tries to find out whether this effect is present in the programs. In this sense, the most critical application is the WLANServer, which is supposed to run as a server for long periods of time without being switched off. It consists of letting successive WLANClients run the negotiation protocol continuously, one after the other, with the same server over a period of time.

This test was carried out for 64 hours and 30 minutes. A single client runs the policy negotiation protocol continuously. The system worked successfully, and no memory losses were observed in the Security Gateway.

5.3.2 Test 2

The WLANServer is supposed to be able to cope with different simultaneous negotiations from a number of WLANClient entities. This situation is quite probable. For example, when the employees arrive in the morning at the office, there may be several of them switching on their mobile terminals at the same time. The WLANServer must be able to cope with all of those requests.

This test consists of having a number of clients to negotiate IPSec tunnels continuously and at the same time with a single WLANServer. In this project's case, two and three-client tests were carried out and the system worked successfully.

Obviously, the throughput goes down as the number of concurrent clients rises.

5.3.3 Other tests

During the development period, the functioning of the protocol was also tested, in order to explore the different situations that derive from packet losses and unexpected situations.

5.4 Outlook of the project

5.4.1 Future of the WLAN network security

It is an admitted fact that, up to date, IEEE 802.11b systems are relatively easy for outside attackers to break.

Some predictions hold that *“... by the end of 2002, about 30 percent of all enterprises will risk security breaches, because they’ve deployed 802.11b WLANs without proper security. Experts advise that until next-generation WLAN security standards are defined, IPSec virtual private networks (VPNs) should be run on all WLAN connections.”*

This was stated in the article[34], which was published in January 2002. What is more: *“Systems based on 802.11b wireless networking technology, popular both in the enterprises and in the home, dominated WLAN sales in 2000. The residential market is the fastest growing segment of the WLAN business. It is projected to account for nearly half of all sales by 2005.”*

This points at the fact that this project was aimed in the right technological trend and directed to a growing market.

In the WLAN security market, two trends have appeared: some think that patching the WEP will solve the problem. The enhanced WEP solutions are an example to this trend.

Others feel that WEP has lost any credibility before the consumers, who do not seem to be willing to send their security-critical information over a WLAN. As an example, the WLAN developer cyberPIXIE decided in December to include the Advanced Encryption Standard (AES) in its suite of WLAN products [36].

The IEEE’s 802.11 Working Group is now developing a next-generation WEP but currently has no proposals for a backwards compatible encryption scheme. In fact, the only encryption scheme that is getting support from the IEEE is an AES-based proposal [17]. A unified standards-based security framework takes, however, a period of time to reach the market and it is not clear if existing installations can be upgraded to such a solution.

VPN solutions belong to those willing to leave WEP aside. Some solutions have been announced so far. However, there is also a general feeling that these VPN technologies are just a provisional patch to WLAN security. In fact, they were not intended to secure wireless environments and therefore no optimality can be expected when applying them to WLANs.

The vendor security frameworks go in another direction [14]. They combine some WLAN concepts (sometimes WEP, for example) with other security techniques. The problem with vendor solutions is their limited interoperability. With regard to the WLAN authentication, the proposal of Symbol Technologies hints at Kerberos as the authentication solution [37] for WLAN environments. Other options include RADIUS and the 802.1x technology (Section 2.3.2) as authentication schemes. The IEEE 802.1x was originally not intended for WLAN Authentication, but it has turned out to be a valid remedy for some WEP limitations, like the shared key automatic management.

Finally, the security add-ons are yet another alternative, in the form of both software and hardware systems (see Section 2.3.5).

5.4.2 Assessment of this project in the landscape of WLAN security

After the analysis of the WLAN landscape made in the previous Section 5.4.1, it is possible to evaluate this project as an alternative to the actual WLAN security options.

A priori, it is a VPN solution. VPN technologies were not originally intended to secure wireless environments. However, they have turned out to be a valid solution to the WLAN security problems. This project is a study of how VPNs can be adapted to wireless environments. It can also be considered as an analysis of the present LAN and VPN technologies and the adaptability of each technology to small and medium-size WLANs. As a VPN solution, it leaves WEP aside, relying on other security technology. IPSec is proposed to secure the traffic of the legitimate WLAN users and protecting the associated wired infrastructure. For this, some software components have been added to the WLAN entities and the architecture described in Section 3.1 has been proposed. Nomadic mobility for users roaming between multiple WLAN installations has been achieved. No special software or hardware are required to deploy the configuration solution proposed. Complicated tasks such as the configuration of IPSec policies or updating of the Windows Registry have been automated so that the dynamic configuration of Mobile Nodes in roaming conditions remains simple and fast.

It does not use a standard authentication schema, such as Kerberos, RADIUS or IEEE 802.1x. Instead, it implements an initial entity authentication based on a policy negotiation protocol (explained in Section 3.5). This protocol is intended to negotiate an IPSec Preshared Key string. This Preshared Key will be then used during the authentication step of the IKE protocol, which is responsible of the IPSec Security Association negotiation.

The security demands of the proposed WLAN scenarios (described in Section 3.1) have been fulfilled. Access control is achieved through mutual entity authentication between Mobile Node and Security Gateway with the IPSec policy negotiation protocol and, later, the IPSec's IKE authentication. Confidentiality and data integrity / origin authentication are provided by individual IPSec encrypted tunnels established between each Mobile Node and the Security Gateway that manages the Security Domain.

The solution has, however, some limitations and future work is needed. IPSec does not protect multicast or broadcast traffic. This is an open issue that requires further analysis. Furthermore, its authentication is host-based: in a multi-user Mobile Node, there is no way to limit the access of each user to the WLAN IPSec tunnel.

In terms of distribution, the prototype is a free product. Furthermore, it is "Public Domain" and can therefore be freely used or modified by anyone without asking for permission of any kind.

Bibliography

- [1] B. Aboba and D. Simon. PPP EAP TLS Authentication Protocol. RFC 2716, October 1999.
- [2] Zubair Alexander. VPN Deployment Using Windows 2000. Microsoft Corporation, February 2001. <http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/colu%msns/profwin/pw0201.asp>.
- [3] R. Atkinson and S. Kent. IP Encapsulating Security Payload (ESP). RFC 2406, 1998.
- [4] R. Atkinson and S. Kent. Security Architecture for the Internet Protocol. RFC 2401, 1998.
- [5] L. Blunk and J. Vollbrecht. PPP Extensible Authentication Protocol (EAP). RFC 2284, March 1998.
- [6] N. Borisov, I. Goldberg, and D. Wagner. Intercepting Mobile Communications: The Insecurity of 802.11. ACM Mobicom, 2001. <http://www.cs.berkeley.edu/~daw/papers/wep-mob01.ps>.
- [7] N. Boscia and D. Shaw. Wireless Firewall Gateway White Paper. Technical report, Advanced Supercomputing Division, NASA, 2001.
- [8] Microsoft Corporation. Virtual Private Networking. Microsoft Windows 2000 Resource Kit. http://www.microsoft.com/WINDOWS2000/techinfo/reskit/en/Intwork/inbe_vp%n_ibxi.htm.
- [9] Microsoft Corporation. Microsoft Privacy Protected Network Access: Virtual Private Networking and Intranet Security. Technical report, Microsoft, 1999.
- [10] Microsoft Corporation. IPSec and L2TP Implementation in Windows 2000 (Q265112), August 2000. <http://support.microsoft.com/default.aspx?scid=kb;DE;q265112>.
- [11] Microsoft Corporation. Virtual Private Networking. Windows XP Documentation, 2001. http://www.microsoft.com/windowsxp/home/using/productdoc/en/default.asp?url=/WINDOWSXP/home/using/productdoc/en/sag_IPSEctunnel.asp.
- [12] Microsoft Corporation. VPN Tunnels - GRE Protocol 47 Packet Description and Use. Microsoft Product Support Services, August 2001. <http://support.microsoft.com/default.aspx?scid=kb;DE;q241251>.

BIBLIOGRAPHY

- [13] Microsoft Corporation. DHCP Options Supported by Clients. Microsoft Windows 2000 Support, February 2002.
- [14] D.Molta. WLAN Security on the Rise. *www.networkcomputing.com*, February 2002. <http://www.networkcomputing.com/1303/1303ws22.html>.
- [15] N. Doraswamy and D. Harkins. *IPSec - The New Security Standard for the Internet, Intranets and Virtual Private Networks*. Prentice Hall, 1999.
- [16] Patel et al. Securing L2TP using IPsec. RFC 3193, November 2001.
- [17] IEEE 802.11 Working Group. Status of Project IEEE 802.11. IEEE Report, January 2002. http://grouper.ieee.org/groups/802/11/Reports/tgi_update.htm.
- [18] K. Hamzeh, G. Pall, W. Verthein, J. Taarud, W. Little, and G. Zorn. Point-to-Point Tunneling Protocol (PPTP). RFC 2637, July 1999.
- [19] D. Harkins and D. Carrel. The Internet Key Exchange (IKE). Internet RFC 2409, 1998.
- [20] IEEE. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. The Institute of Electrical and Electronics Engineers (IEEE), IEEE Std 802.11-1997, 1997.
- [21] IEEE. Standards for Local and Metropolitan Area Networks: Standard for Port Based Network Access Control. The Institute of Electrical and Electronics Engineers (IEEE), IEEE Draft P802.1X/D11, 2001, 2001.
- [22] S. Kent and R. Atkinson. IP Authentication Header. RFC 2402, November 1998.
- [23] H. Krawczyk, M. Bellare, and R. Canetti. *HMAC: Keyed-Hashing for Message Authentication*, February 1997. RFC 2104.
- [24] D. Maughan, M. Schertler, M. Schneider, and J. Turner. Internet Security Association and Key Management Protocol (ISAKMP). RFC 2408, November 1998.
- [25] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press LLC, 1997.
- [26] D. Piper. The Internet IP Security Domain of Interpretation for ISAKMP. RFC 2407, November 1998.
- [27] C. Rigney, S. Willens Livingston, A. Rubens Merit, and W. Simpson Daydreamer. Remote Authentication Dial In User Service (RADIUS). RFC 2138, June 2000.
- [28] Guenter Schaefer. IPSec. Lecture slides of Network Security, TKN, TUB, 2002. <http://www-tnk.ee.tu-berlin.de/curricula/NetworkSecurity>.
- [29] B. Schneier and Mudge. Cryptanalysis of Microsoft's Point-to-Point Tunneling Protocol (PPTP). *Proceedings of the 5th ACM Conference on Communications and Computer Security*, ACM Press, pages 132–141, 1998.

- [30] B. Schneier, Mudge, and D. Wagner. Cryptanalysis of Microsoft's PPTP Authentication Extensions (MSCHAPv2). Counterpane Systems, 1999.
- [31] A. Shacham, R. Monsour, R. Pereira, and M. Thomas. IP Payload Compression Protocol (IP-Comp). RFC 2393, December 1998.
- [32] A. Shamir, I. Mantin, and S. Fluhrer. Weaknesses in the Key Scheduling Algorithm for RC4, August 2001. http://eyetap.org/~rguerra/toronto2001/rc4_ksaproc.pdf.
- [33] W. Simpson. The Point-to-Point Protocol (PPP). RFC 1661, July 1994.
- [34] Global Sources. Security, high cost remain key issues in WLAN. *Global Sources Magazine*, January 2002. <http://www.globalsources.com/MAGAZINE/CP/0201/CPISSU.HTM>.
- [35] A. Stubblefield, J. Ioannidis, and A. D. Rubin. Using the Fluhrer, Mantin and Shamir attack to break the WEP, 2001.
- [36] E. Sutherland. Examining Alternatives to Patching WEP. *www.80211-planet.com*, January 2002. www.80211-planet.com/columns/article/0,4000,1781_958331,00.html.
- [37] Inc. Symbol Wireless. Security White Paper: Evolution, Requirements, Options. *www.80211-planet.com*, January 2002. http://www.80211-planet.com/tutorials/article/0,4000,10724_965471,00.ht%ml.
- [38] W. Townsley, A. Valencia, A. Rubens, G. Pall, G. Zorn, and B. Palter. Layer Two Tunneling Protocol "L2TP". RFC 2661, August 1999.
- [39] A. Valencia. L2TP Header Compression ("L2TPHC"). Internet Draft, October 2001. <http://www.ietf.org/internet-drafts/draft-ietf-l2tpext-l2tphc-04.txt>.

BIBLIOGRAPHY

Appendix A

Overview of the implementation program files

A.1 Description of the components of the programs

There are two entities: server and client. Each of them consists of a subset of the programs analyzed in Section A.2. This is expressed in Table A.1

A.2 Files contained in the programs

A.2.1 Common files

These are files that are shared by different programs. They will be described only once. The correspondence between files and programs is described in Table A.2.

sha.h/sha.cpp provides the definitions and source code of the functions of the sha implementation

Table A.1: Correspondance between entities and programs

Programs	Server entity	Client entity
WLANServer.exe	X	
WLANClient.exe		X
RandomInit.exe	X	X
InitialIPSecConfigurator.exe	X	
SGNameConfigurator.exe	X	
WLANDisconnect.exe		X
WLANSERVICE.exe	X	

Table A.2: Relationship between programs and common files

	sha	hmac	endian	global	psapi	wlanrandom	ws_utils
WLANServer.exe	X	X	X	X	X	X	X
WLANClient.exe	X	X	X	X	X	X	X
RandomInit.exe	X		X	X			
InitialIPSecConfigurator.exe							
SGNameConfigurator.exe	X		X	X			
WLANDisconnect.exe							X
WLANService.exe							

used. It also defines a structure `SHA_CTX`, which is used within the sha operation.

hmac.h/hmac.cpp provides the definitions and source code of the functions of the hmac implementation used. It also defines an HMAC constant value.

endian.h/endian.cpp defines and implements a function in order to test the endianness of bytes.

global.h defines some types useful for the sha and hmac implementations used.

psapi.h provides the definition of the Windows API functions to access the process and memory usage information.

wlanrandom.h/wlanrandom.cpp implements the random-related functions of the server. It provides a function to generate variable-length random-byte arrays, and another function to seed the PRNG as described in Section 3.5.2.

ws_utils.h/ws_utils.cpp contains the definitions and implementation of the functions related to winsock, such as recovering the IP address of a host or the Default Gateway's IP address contained in the IP configuration of a host.

A.2.2 WLANServer

WLANServer.cpp This file contains the `main()` method of the `WLANServer.exe` application. It performs the initial set up of the server: it sets up the UDP sockets, retrieves the user information contained in the user database and seeds the PRNG according to the procedure described in Section 3.5.2. If the initial set up is completed without errors, it begins an infinite loop, accepting one incoming protocol frame and processing it conveniently in each iteration, according to the protocol specification explained in Section 3.5.

confmanager.h/confmanager.cpp `confmanager.h` provides the definitions of the functions implemented in `confmanager.cpp`, as well as the definition of the `clientFile` structure, which stores the information concerning each user, retrieved from the users database. `confmanager.cpp` implements the users database-related functions, including the database file parsing and registering the different users' `clientFile` objects in a vector.

reqprofile.h/reqprofile.cpp reqprofile.h defines the reqprofile class, which is used by the server to store the information contained in the incoming client negotiation requests. It also provides the definition of functions related to this class, and implemented in reqprofile.cpp. reqprofile.cpp implements the reqprofile class' functions, as well as the other functions associated to it, such as request frame parsing, profile analysis and reply frame production based on a reqprofile object.

confprofile.h/confprofile.cpp confprofile.h defines the confprofile class, which is used by the server when a user's confirmation has been received: the data contained in that frame is parsed into a confprofile object. It also provides the definition of functions related to this class, and implemented in confprofile.cpp. confprofile.cpp implements the confprofile class' functions, as well as the other functions associated to it, such as confirmation frame parsing, confprofile analysis and generation of session keys from the confirmation frame's data.

errorframe.h/errorframe.cpp errorframe.h defines a method to construct the error replies sent by the server to the clients, as well as the different error codes. errorframe.cpp implements the function for making error frames.

A.2.3 WLANClient

WLANClient.cpp This file contains the main() method of the WLANClient.exe application. It performs the initial set up of the client: sets up the UDP sockets, retrieves the Security Domains information contained in the Security Domain database and seeds the PRNG according to the procedure described in Section 3.5.2. If the initial set up is completed without errors, it begins the protocol explained in Section 3.5.

confmanager.h/confmanager.cpp confmanager.h provides the definitions of the functions implemented in confmanager.cpp, as well as the definition of the serverFile structure, which stores the information concerning each Security Gateway, retrieved from the Security Domains database. confmanager.cpp implements the Security Domains database-related functions, including the database file parsing and registering the different Security Gateways' serverFile objects in a vector.

frames.h defines the classes frame1 and frame2, which encapsulate the information contained in the client's request frame, and in the server's reply frame. It also defines functions associated to them, such as frame parsing or making, frame contents analysis and production of the session key from the information contained in the server's reply.

frame1.cpp implements the frame1 class functions, as well as request frame making functions.

frame2.cpp implements the frame2 class functions, as well as server's reply frames parsing, confirmation making and session calculation.

A.2.4 RandomInit

RandomInit.cpp This file contains the main() method of the RandomInit.exe application. It asks the user to enter 100 keystrokes and stores the times between keystrokes (in milliseconds). It produces a hash out of all these stored values, concatenated in an array.

A.2.5 InitialIPSecConfigurator

InitialIPSecConfigurator.cpp This file contains the main() method of the InitialIPSecConfigurator.exe application. It inserts rules into the IPSec policy in order to block all the IP addresses of the specified WLAN, preparing the environment for a secure IPSec operation.

A.2.6 SGNameConfigurator

SGNameConfigurator.cpp this file contains the main() method of the SGNameConfigurator.exe application. It asks the user to enter 100 keystrokes and stores the times between keystrokes (in milliseconds). It produces a hash out of all these stored values, concatenated in an array. From this hash, it takes a few bytes and concatenates them with the Security Gateway's hostname.

A.2.7 WLANDisconnect

WLANDisconnect.cpp this file contains the main() method of the WLANDisconnect application. It updates the client's IPSec policy so that it allows all incoming and outgoing traffic.

A.2.8 WLANService

WLANService.cpp this file contains the main() method of the WLANService application. It installs the WLANServer as an NT server and updates the Windows Registry with the program's path and command line arguments.

A.3 Folder structure and build instructions

The folder structure has been organized so that all the common files reside in a unique, shared folder. This is convenient when making changes in one of the shared components. If the shared components were not in a common directory, the changes would need to be repeated in every instance of that component.

The programs are placed in folders at the same level as that of the common components' folder, as shown in Figure A.1. The components that belong uniquely to each program (non-shared components) are placed in the program's folder. No further hierarchy is used.

In order to build each program, one Visual C++ Project has been placed in each of the program folders. All the necessary components have been added to these Visual C++ projects. In some projects, additional libraries must be added. That can be accomplished in the Properties option, in the Linker snap. The additional libraries to be linked with the projects are listed in Table A.3. The provided Visual C++ projects work properly only if the folder structure remains unchanged.

Essentially, that is all: open and build the projects with the Visual C++ compiler.

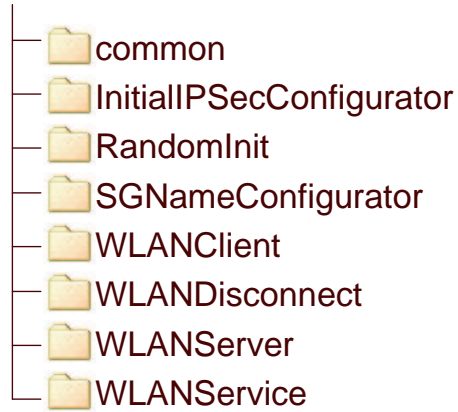


Figure A.1: Folder structure to build the programs.

Table A.3: Libraries to be linked with each program

Programs	wsock32.lib	psapi.lib
WLANServer.exe	X	X
WLANClient.exe	X	X
RandomInit.exe		
InitialIPSecConfigurator.exe	X	
SGNameConfigurator.exe		
WLANDisconnect.exe	X	
WLANService.exe		

