Luís Manuel Cerqueira Barreto

Segurança em Arquitecturas de Rede para Acesso sem Fios

Departamento de Ciências de Computadores
Faculdade de Ciências da Universidade do Porto
Julho / 2005

Luís Manuel Cerqueira Barreto

Segurança em Arquitecturas de Rede para Acesso sem Fios

Tese submetida à Faculdade de Ciências do Porto
Para obtenção do grau de Mestre em Informática

Departamento de Ciências de Computadores
Faculdade de Ciências da Universidade do Porto
Julho / 2005

Dedicatória

À minha filha, a Sara Maria, o meu mais precioso tesouro e à minha esposa, a Maria dos Anjos, pelo apoio e paciência que sempre demonstrou neste tempo de maior ausência.

Agradecimentos

À Prof. Doutora Susana Sargento e ao Prof. Doutor Luís Antunes queria agradecer os excelentes conselhos, a sábia orientação e o apoio constante e determinante.

Ao Mestre Pedro Brandão pelo seu apoio constante, conselhos e disponibilidade que sempre evidenciou.

À minha esposa, filha, pais, sogros e irmã pela compreensão e apoio que sempre me deram.

Aos meus colegas, da Escola Superior de Ciências Empresariais do Instituto Politécnico de Viana do Castelo, o meu muito obrigado, pela compreensão e ajuda que sempre manifestaram.

Resumo

A popularidade das redes sem fios tem aumentado significativamente nestes últimos anos. As redes sem fios têm sido instaladas em serviços e instituições do mais variado tipo, nomeadamente em instituições governamentais, académicas e militares. A sua crescente popularidade deve-se a muitos factores, mas os principais são a facilidade e flexibilidade de instalação, mobilidade e escalabilidade. Estas redes apresentam, no entanto, necessidades especiais ao nível da segurança. O meio de transmissão utilizado nas redes sem fios é o ar. Sendo este um meio inerentemente partilhado, para atacar este tipo de redes não é necessário estar fisicamente ligado. Um potencial intruso, munido das ferramentas de *software* adequadas pode visualizar a informação que circula na rede, alterar essa informação para seu proveito e até mesmo fazer-se passar por um cliente autorizado da rede sem fios. É importante implementar e definir mecanismos que melhorem a segurança das redes sem fios. Estes mecanismos deverão garantir o controlo de acesso dos utilizadores à rede, e ainda a integridade e a confidencialidade da informação.

Nesta dissertação é realizado um estudo teórico e experimental dos protocolos de segurança e mecanismos utilizados para proteger as redes sem fios: WEP, IPsec, 802.1X, WPA e IEEE802.11i. Como parte integrante do estudo experimental, são implementados em laboratório, cenários experimentais onde se utilizam o conjunto de soluções IPsec e WPA (na sua variante WPA-EAP e WPA-PSK) para proteger uma rede sem fios. Sistemas protegidos pelo IEEE802.11i não são apresentados pelo facto de não existirem ainda elementos compatíveis com esta norma. Para ser possível efectuar uma análise crítica às capacidades de segurança e aos custos introduzidos no desempenho da rede de cada uma das soluções, efectuaram-se um conjunto de testes experimentais de segurança, desempenho e complexidade das implementações. Os primeiros testes simulam possíveis ataques a uma rede sem fios. Os ataques realizados são do tipo "*man-in-the-middle*", personificação, visualização da informação, modificação da informação e controlo de sessão. Para avaliar o desempenho da rede utiliza-se um conjunto de ferramentas que permitem obter os valores de *throughput*, *jitter* e número de pacotes perdidos na presença de cada um dos mecanismos de segurança. Como complemento destes testes, efectua-se também uma análise do desempenho dos equipamentos de rede na presença dos mecanismos de segurança estudados. Para analisar o seu desempenho são efectuadas medidas relacionadas com a percentagem de CPU utilizada, a memória utilizada, o número de processos e o número de *interrupts*.

Abstract

Wireless lans (WLANs) are growing in popularity. They have been used in all kinds of services and in military, governmental and academic institutions. Their popularity is directly associated with their ease and flexibility of installation, scalability and mobility. However, these networks have special needs in security. Air is their mean of transmission. To attack these networks, as air is a shared medium, it is not required to be fiscally connected to them. A hacker, with the right software tools, can see all the information in the network, change that information for its own profit, and access to the network as an authorized user. It is of extremely importance to define and implement mechanisms that increase security in WLANs. These mechanisms should ensure information confidentiality and integrity, and also implement strong user access control methods.

In this Master Thesis it is performed a theoretical and experimental study of the mechanisms and protocols used to protect WLANs: WEP, IPsec, 802.1X, WPA and IEEE802.11i. Solutions like IPsec and WPA (in its variants WPA-EAP and WPA-PSK), are implemented in laboratory scenarios as part of the experimental study. A IEEE802.11i scenario is not presented, as there are no IEEE802.11i compliant equipments yet. A set of tests regarding security, performance and implementation complexity were performed, making it possible to analyze network performance costs and security capabilities of each solution. The first tests simulate possible attacks to WLANs. These attacks are man-in-the-middle, impersonation, sniffing and session hijacking. Network performance is measured with a set of tools. With those tools, it is possible to measure, in presence of each of the security mechanisms, network throughput, jitter and packet loss. System performance is also measured as a complement to the previous tests. System performance is related to percentage of CPU used, memory used, number of processes and number of interrupts.

Índice

Índice de Figuras

Índice de Tabelas

Acrónimos

AAA – Authentication, Accounting, Authorization

AC – Autoridade Certificadora

AES – Advanced Encryption Standard

AH – Authentication Header

AP – Access Point/Ponto de Acesso

ARP – Adress Resolution Protocol

AS – Access Server

CA – Certificate Authority

CCM CBC-MAC – Counter Mode with Cipher Block Chaining Message Authentication Code

CHAP – Challenge Handshake Protocol

CRC – Cyclic Redundancy Check

CRL – Certificate Revogation List

DEK – Data Encryption Key

DES – Data Encryption Standard

DHE – Diffie Hellman Exchange

DIK – Data Integrity Key

DMZ – Demilitarized Zone

DN – Distinguished Name

DNS – Domain Name Service

DOI – Domain of Interpretation

DoS – Denial of service

EAP – Extensible Authentication Protocol

EAPoL – Extensible Authentication Protocol over LAN

ESP – Encapsulating Security Payload

FTP – File Transfer Protocol

GEK – Group Encryption Key

GIK – Group Integrity Key

GMK – Group Master Key

GTK – Group Transient Key

HIDS – Host Intrusion Detection System

ICMP – Internet Control Message Protocol

ICV – Integrity Check Value

IDS – Intrusion Detection System

IE – Information Element

IEEE – Institute of Electrical adnd Electronic Engineers

IETF – Internet Engineering Task Force

IKE – Internet Key Exchange

IP – Internet Protocol

IPsec – Internet Security Protocol

ISAKMP – Internet Security Association Key Management Protocol

ITU – International Telecommunication Union

IV – Initialization Vector/Vector de Inicialização

KCK – Key Confirmation Key

KDC – Key Distribution Center

KEK – Key Encryption Key

KIK – Key Integrity Key

KLIPS – Kernel IPsec Support

MAC – Message Authentication Code / Medium Access Control

MD5 – Message Digest 5

MIC – Message Integrity Code

MITM – Man-In-The-Middle

MK – Master Key

MPDU – Message Authentication Code Protocol Data Unit

MS-CHAP – Microsoft CHAP

NAS – Network Access Server

NIC – Network Interface Card

NIST – National Institut of Standards

NSA – National Security Agency

OTP – One Time Passwords

PAE – Port Access Entity

PAP – Password Authentication Protocol

PDP – Ponto de Decisão da Política

PEAP – Protected EAP

PEP – Ponto de Reforço da Política

PGP – Pretty Good Privacy

PKI – Public Key Infraestructure

PMK – Pairwaise Master Key

PPP – Point-to-Point Protocol

PRNG – Pseudorandom Number Generator

PTK – Pairwaise Transient Key

RADIUS – Remote Access Dial-in User Service

RC4 – Rivest Code Four

RFC – Request For Comments

RSA – Rivest Shamir Adleman

RSN – Robust Security Network

RW – Road Warrior

SA – Security Association

SADB – Security Association Database

SCP – Secure Copy

SDT – Token de Decisão da Sessão

SET – Token de Reforço da Sessão

SHA – Secure Hash Algorithm

SO – Sistema Operativo

SPI – Security Parameter Index

SSL – Secure Sockets Layer

TCP – Transport Control Protocol

TG – Task Group

TGi – Task Group i

TKIP – Temporal Key Integrity Protocol

TLS – Transport Layer Security

TOS – Type of Service

TSC – TKIP Sequence Counter

TTLS – Tunneled TLS

UDP – User Datagram Protocol

VACL – VLAN Access Control List

VPN – Virtual Private Network

WEP – Wired Equivalent Protocol

Wi-Fi – Wireless Fidelity

WLAN – Wireless Local Area network

WPA – Wi-Fi Protected Access

WRAP – Wireless Robust Authenticated Protocol

Capítulo 1

Introdução

Uma rede de computadores é cada vez mais um factor de competitividade e de produtividade de uma organização. As redes de computadores sofreram nas duas últimas décadas um crescimento exponencial. A sua utilização permite melhorar a organização das tarefas a desenvolver por uma organização. As redes de computadores permitem também a partilha de recursos físicos, como discos e impressoras, o que origina economia e rentabilização de recursos. A maioria das redes instaladas é do tipo *ethernet* e utilizam o protocolo 802.3. Inicialmente estas redes permitiam larguras de banda de 10 *Mbits/s* utilizando como meio de transmissão cabos de pares trançados. Actualmente e devido à grande evolução tecnológica, são possíveis larguras de banda até 10 *Gbits/s*. Isto significa uma evolução de mil vezes na velocidade da transferência dos dados.

Essa constante evolução tecnológica permitiu também que se utilizassem as frequências rádio para a comunicação entre dois ou mais computadores. Surgiu assim o conceito de rede de computadores sem fios ou rede sem fios (também conhecidas por redes *wireless*). A maioria das redes sem fios utiliza o protocolo 802.11, sendo conhecidas por redes *Wireless Fidelity* (*Wi-Fi*). De uma forma muito simples, podemos comparar estas redes a uma vulgar rede do tipo *ethernet*, mas na qual as ligações entre dispositivos são substituídas por ligações sem fios utilizando radiofrequência.

A primeira norma 802.11 utilizava frequências rádio na banda 2,4 *GHz*, mas a sua largura de banda limitava-se a 2 *Mbits/s*. A primeira evolução da norma 802.11, a norma 802.11a, oferece mais canais, menos interferências, maior segurança e mais velocidade (54 *Mbits/s*), mas utiliza uma frequência rádio na gama 5.8 *GHz*. Este factor exige equipamentos muito mais caros e sem grande capacidade de interoperabilidade com os restantes equipamentos. Devido a todos estes problemas, desenvolveu-se a norma 802.11b. É a esta norma que se deve a revolução do *Wi-Fi*. Esta norma utiliza, tal como a norma inicial, a gama de frequências rádio dos 2,4 *GHz* e permite uma largura de banda de 11 *Mbits/s*. A norma 802.11b permite a implementação de dispositivos a custos relativamente baixos. Muito recentemente surgiu a norma 802.11g que permite velocidades de transferência de dados a 54 *Mbits/s*, sendo no entanto, compatível com a norma 802.11b. Os custos dos equipamentos compatíveis com esta norma são também relativamente baixos.

As redes sem fios têm sido utilizadas em muitos sectores da sociedade. Os motivos da sua popularidade são a facilidade e flexibilidade de instalação, mobilidade e escalabilidade. O número crescente de pontos de acesso público à *Internet* (denominados *Hotspots*) em locais públicos como hotéis, centros de congressos, aeroportos, repartições públicas, é apenas possível devido à utilização de redes sem fios. Um exemplo de implementação de redes sem fios de larga escala com bastante sucesso é o projecto *Campus Virtuais*, no qual Portugal é um pioneiro. Este projecto lançado pelo Governo envolve serviços, conteúdos, aplicações e rede de comunicações móveis (dentro e fora dos estabelecimentos de ensino) para estudantes e professores do Ensino Superior, permitindo um maior incentivo e facilidade na produção, no acesso e partilha do Conhecimento. A ideia dos *Campus Virtuais* é utilizar uma rede sem fios que permita transmissão de dados em banda larga possibilitando o acesso a aulas, artigos, trabalhos, notas, serviços e *Internet* com um computador portátil e a partir de qualquer ponto da rede de estabelecimentos do Ensino Superior, estando a informação ininterruptamente disponível para professores e alunos.

As redes sem fios utilizam como meio de transmissão um meio partilhado como o ar. Esta característica exige a implementação de mecanismos de segurança robustos e eficazes. Ao contrário das redes com fios em que é possível controlar o acesso de forma física aos recursos e à informação disponível na rede, numa rede sem fios é apenas necessário a um intruso (ou atacante), dentro do raio de acção da rede sem fios, estar devidamente equipado com um dispositivo com a ferramenta de *software* apropriada para poder obter acesso a recursos e a informação disponíveis. Várias são também as ferramentas de *software* disponíveis para atacar este tipo de redes. Estas ferramentas são disponibilizadas de forma gratuita numa outra rede de computadores, a *Internet*. Todas as ferramentas disponibilizadas são de utilização muito simples e eficaz. Atendendo a todas estas questões, ao nível da segurança devem ser garantidos, nas redes sem fios, mecanismos que garantam os seguintes serviços:

- **Autenticação** – serviço que procura garantir ao receptor que o utilizador que se anuncia é quem realmente afirma ser;
- **Confidencialidade** – serviço que garante a protecção dos dados transmitidos contra ataques de leitura, recorrendo a técnicas de cifra dos dados;
- **Integridade** – serviço que assegura que os dados são recebidos tal como foram enviados, isto é, sem duplicação, inserção, modificação ou reordenação;
- **Não-repúdio** – serviço que permite a uma terceira pessoa implicar o envolvimento de duas ou mais partes numa comunicação.

A primeira tentativa de implementar os serviços de segurança antes referidos foi a introdução do protocolo de segurança *Wired Equivalent Privacy* (WEP). Este mecanismo de segurança não se mostrou suficiente nem capaz; o protocolo de segurança WEP está desactualizado e apresenta fraquezas evidentes ao nível da criptografia. Estudos provaram que um intruso equipado com as ferramentas adequadas e um nível de conhecimentos técnicos moderado, facilmente conseguiria violar uma rede sem fios configurada apenas com o protocolo de segurança WEP. O WEP disponibiliza muito pouca segurança efectiva a uma rede, sendo muito vulnerável a ataques de autenticidade, confidencialidade e integridade dos dados [31][54].

A inoperância ao nível da segurança das redes sem fios levou a que se desenvolvessem novos mecanismos de segurança, nomeadamente o 802.1X, o *Wireless Fidelity Protected Access* 1 (WPA1) e o mais recente *Robust Security Network* (RSN, também denominado WPA2 ou IEEE802.11i). O 802.1X pode apenas ser considerado como um mecanismo robusto de autenticação, já que utiliza o protocolo WEP para proteger as comunicações. O WPA1 foi desenvolvido pela *Wi-Fi Alliance* [87] e tem como base o IEEE802.11i. A *Wi-Fi Alliance* desenvolveu este mecanismo de segurança, de forma a ser possível introduzir nos dispositivos *Wi-Fi* compatíveis com o WEP, um mecanismo de segurança realmente eficaz e robusto. O RSN foi desenvolvido pelo grupo de trabalho *i* do IEEE802.11, e será a norma *de facto* para as redes sem fios. Este mecanismo de segurança fornece primitivas e capacidades criptográficas novas e mais capazes, o que garante a confidencialidade e a integridade dos dados numa rede sem fios. O controlo de acesso a utilizadores é também uma das características desta norma, garantindo-se o serviço de autenticação e de não-repúdio. A principal desvantagem deste mecanismo é a necessidade de se terem de adquirir novos dispositivos para assim se poder usufruir de todas as potencialidades do mesmo. A utilização total das potencialidades do RSN não é compatível com equipamentos WEP.

Esta dissertação tem como objectivo principal efectuar um estudo criterioso ao nível da implementação, teste e avaliação dos mecanismos de segurança existentes para as redes sem fios. Os principais tópicos a abordar são o estudo dos protocolos e serviços que constituem as soluções existentes para aumentar a segurança numa rede sem fios. Incluem-se nos mecanismos de segurança soluções que não foram desenvolvidas de raiz para aplicação em redes sem fios. Como exemplo desses mecanismos de segurança pode-se referir o *Internet Protocol Security* (IPSec), cuja utilização vai muito para além das redes sem fios. Ao implementar mecanismos de segurança é também necessário considerar outros factores, tais como o custo e o nível de desempenho da rede com e sem os mecanismos de segurança.

Numa rede de média/grande dimensão, o custo de implementar um determinado mecanismo de segurança poderá ser de tal ordem que o tornará impraticável. O mesmo poderá acontecer relativamente ao desempenho da rede: deve-se ter em atenção que a degradação no desempenho de uma rede pela utilização de um determinado mecanismo de segurança poderá ser um factor de não utilização desse mecanismo. Sendo assim, é necessário estudar os diferentes mecanismos de segurança existentes para protecção de redes sem fios, e avaliá-los de acordo com a protecção implementada, o seu custo, e o seu impacto no desempenho da rede.

1.1. Objectivos

Os objectivos desta dissertação encontram-se descritos de seguida:

- Avaliar as principais ameaças à segurança nas redes sem fios;
- Analisar as tecnologias de segurança de um ponto de vista experimental;
- Analisar as dificuldades de implementação dos mecanismos de segurança baseados em sistemas em código aberto;
- Compreender e verificar o funcionamento dos diferentes mecanismos, estudar as suas vulnerabilidades e implementar possíveis ataques;
- Avaliar o impacto introduzido pelos diferentes mecanismos de segurança no desempenho das redes sem fios.

1.2. Estrutura

Esta dissertação encontra-se estruturada em 8 capítulos, organizados da seguinte forma.

No capítulo 2 é efectuada uma avaliação à segurança em redes sem fios, com uma abordagem aos tipos de intrusos e aos tipos de ataques que estes inimigos poderão realizar. É também feita uma avaliação à necessidade de segurança em redes de dados, sobretudo no tipo de redes que aqui são objecto de estudo.

No capítulo 3 introduzem-se os conceitos básicos de segurança em redes. Estudam-se os princípios da criptografia, nomeadamente a criptografia por chave-simétrica e chave-pública, os serviços de autenticação, confidencialidade, integridade e não-repudiação. São ainda apresentados os conceitos inerentes aos diversos algoritmos criptográficos, funções de *hash*, assinaturas digitais, métodos de gestão de chaves e métodos de distribuição de chaves. O capítulo apresenta ainda o conceito de autoridade certificadora.

No capítulo 4 estudam-se os protocolos de autenticação mais utilizados em redes sem fios. Este capítulo descreve o protocolo de autenticação EAP e o serviço RADIUS.

No capítulo 5 estudam-se os protocolos de segurança utilizados em redes sem fios. É feita uma abordagem ao primeiro protocolo de segurança criado de raiz para as redes sem fios, o WEP; estudam-se de seguida novos protocolos que permitem diluir ou até mesmo eliminar os defeitos apontados ao WEP, como o IPsec, o WPA e o IEEE802.11i.

No capítulo 6 apresentam-se um conjunto de implementações baseadas em alguns dos protocolos estudados no capítulo anterior. Pretende-se com este capítulo estudar as dificuldades da implementação dos protocolos de segurança com ferramentas em código aberto, bem como verificar o seu grau de complexidade ao nível de equipamentos e configuração. Neste capítulo apenas se estudaram implementações baseadas nos protocolos IPsec e WPA.

No capítulo 7 estuda-se o nível de desempenho da rede e o grau de segurança da mesma, antes e depois de se implementarem os mecanismos de segurança descritos no capítulo 6.

Esta dissertação termina no capítulo 8, onde se apresentam as principais conclusões do trabalho realizado e tópicos para trabalho futuro.

Capítulo 2

Ameaças às redes sem fios

A protecção de uma rede depende da importância dos dados a proteger, e dos prejuízos originados pela perca de informação, e pelo acesso a informação confidencial ou privada. Para se garantir a segurança numa rede é importante compreender de que forma a rede poderá ser atacada, quem a poderá atacar e quais as ferramentas que poderão ser utilizadas nos ataques. A recolha criteriosa destes aspectos permitirá conceber um plano de segurança que defina as acções a serem tomadas – quais os serviços disponíveis na rede e que acessos serão permitidos. Sendo as redes sem fios um tipo de rede com características próprias, é importante a realização de um estudo das ameaças a este tipo de redes.

A detecção e o registo de falhas de segurança nas redes sem fios possibilitarão a implementação de um plano de segurança correcto, limitado na maioria das situações por orçamentos muito restritivos. No entanto, mesmo após ser concebido um plano de segurança que satisfaça na sua maioria os requisitos necessários, deverão ser analisadas e testadas possíveis falhas de segurança. O processo de definição de uma política de segurança, em redes sem fios, é um processo dinâmico em constante desenvolvimento e actualização.

Este capítulo descreve na sua primeira secção (2.1) os tipos de intrusos que poderão atacar as redes sem fios. A secção 2.2 apresenta um conjunto de ameaças para as redes sem fios, indicando sempre que possível quais as ferramentas que poderão ser utilizadas para realizar esses ataques.

2.1. Tipos de Intrusos

Para se proteger uma rede sem fios é necessário conhecer/prever quais os seus potenciais inimigos ou intrusos. Serão os intrusos que, mediante a utilização de um conjunto de ferramentas de *software* e/ou *hardware*, tentarão atacar a rede sem fios. Um intruso não é mais do que um utilizador não autorizado, que utiliza todos os meios ao seu alcance para obter informação ou recursos apenas disponíveis aos utilizadores autorizados. Os utilizadores não autorizados podem ser divididos em quatro categorias:

- Utilizadores acidentais – São utilizadores que não tentam aceder de forma não autorizada à rede e que podem nem saber que a rede existe ou que tiveram acesso a ela. Se os seus computadores estiverem configurados para se associarem a qualquer rede que detectem, os utilizadores terão acesso aos recursos da rede. Normalmente os

computadores pessoais têm esta configuração activa, sendo a forma de aceder a *hot spot* públicos. Actualmente existem ainda muitas redes que não estão configuradas para bloquearem o acesso a utilizadores não autorizados.

- Intruso do tipo *script kiddie* – O termo *script kiddie* (normalmente utilizado de forma depreciativa) refere-se a utilizadores com intenção de ser um intruso, mas com falta de talento para tal. Este utilizador efectua o *download* de algumas ferramentas disponíveis de forma pública, sabe como utilizá-las, mas pode nem sequer saber como funcionam. Um dos maiores problemas com as redes sem fios é que mesmo um *script kiddie* poderá aceder aos recursos de uma rede mal protegida.
- Intrusos casuais – Este tipo de intrusos sabem como funcionam as ferramentas e podem até ter ajudado a desenvolver ou escrever alguma das ferramentas mais conhecidas. No entanto, o seu interesse nos recursos de uma outra rede é puramente casual. O intruso casual apenas perseguirá os objectivos mais fáceis, isto é, aqueles que são implementados com poucas ou nenhumas medidas de segurança.
- Intrusos talentosos – Este tipo de intruso pode ser pago por uma empresa para aceder aos recursos de uma rede da concorrência. Normalmente, estes intrusos são os que desenvolvem as ferramentas de ameaça à segurança de redes mais utilizadas; têm a paciência e a perspicácia necessárias para explorar as fraquezas de um sistema.

Na fase de implementação de um plano de segurança deve-se ter em atenção qual ou quais os tipos de intrusos que deverão ser impedidos de aceder à rede sem fios.

2.2. Tipos de Ameaças à Segurança das Redes sem Fios

A protecção de uma rede deverá ser assegurada, como já anteriormente mencionado, por um conjunto de serviços: autenticação, confidencialidade, integridade e não-repúdio. Existem vários tipos de ameaças, cada uma afectando um ou mais dos principais serviços de segurança de uma rede. Na prática, a maioria dessas ameaças resulta em ataques à confidencialidade, integridade, disponibilidade e ataques de personificação.

- **Confidencialidade** significa manter a informação secreta. Alguém que tente atacar a confidencialidade pretende descobrir algo que não é desejável ser do domínio público.
- **Integridade** significa ter a certeza que os dados são realmente aquilo que pretendem ser. A integridade fica comprometida quando alguém altera ou destrói os dados.
- **Disponibilidade** significa a possibilidade de aceder a um serviço quando e sempre que necessário. Os ataques do tipo negação do serviço (*Denial of Service*), que têm como objectivo eliminar um servidor ou recursos de uma rede, são ataques à disponibilidade.

- **Personificação** significa a capacidade de um atacante assumir a identidade de uma das partes legítimas de um sistema de comunicações. Podem ser considerados exemplos deste tipo de ataque o "homem no meio" e os *Rogue Access Points*.

De seguida descrevem-se algumas ameaças relacionadas com os acessos sem fios:

Reconhecimento

Nesta situação, o intruso utiliza apenas um conjunto de ferramentas para detectar possíveis redes sem fios para serem ameaçadas. Pode-se considerar este ataque como sendo passivo, já que na realidade não é realizada nenhuma acção na rede, apenas se pretende localizar e registar possíveis pontos a atacar.

Existe um conjunto vasto de ferramentas que poderão efectuar ataques de reconhecimento às redes sem fios, sendo as mais populares o *netstumbler* [13], o *kismet* [19] e *Cain e Abel* [30].

Manipulação da informação

Este tipo de ataque permite ao intruso manipular os dados que circulam na rede. Existe um conjunto de situações que possibilitam os ataques de manipulação dos dados: a manipulação de endereços IP e a repetição dos dados. Este tipo de ataque afecta a confidencialidade e a integridade dos dados.

A manipulação de endereços IP é utilizada por um intruso, quando este utiliza um endereço IP autorizado para obter acesso a recursos e serviços da rede, passando assim por um utilizador legítimo da rede. Este tipo de ataque também é conhecido por *IP Spoofing*. Para se conseguir realizar este tipo de ataque será necessária uma ferramenta que permita a criação ou configuração de pacotes IP, tendo como exemplos mais difundidos o *Airopeek* [15], *sniffer wireless* [16], *ethereal* [17], *ettercap* [20] e o *Cain e Abel* [30].

Na repetição dos dados, um atacante espera que um utilizador legítimo inicie uma sessão, capturando essa sequência de pacotes. Depois, o atacante poderá ou não manipular essa informação, e na ocasião escolhida, reenvia a sequência de pacotes tentando obter acesso não autorizado à rede. Este ataque tenta explorar as fraquezas dos mecanismos de autenticação dos sistemas. Ferramentas utilizadas para efectuar este ataque são o *ethereal* [17] e o *ettercap* [20].

Homem no meio (Man In The Middle – MITM)

Esta ameaça pode também ser considerada um ataque do tipo manipulação de informação. Neste tipo de ataque, o intruso coloca-se normalmente no meio de uma sessão a decorrer entre dois dispositivos (também conhecido por desvio de sessão).

Num ataque de desvio de sessão, um intruso tenta aproveitar-se de uma sessão já estabelecida. Após o início de uma sessão por parte de um utilizador, o intruso utiliza uma ferramenta que permita a modificação das mensagens em trânsito. Existem duas formas de modificação da mensagem: directamente durante o processo de desvio da sessão; ou guardar as mensagens, alterá-las e só depois injectá-las novamente na sessão (processo conhecido por *store and forward*).

Este tipo de ataque é possível mediante a utilização de um processo denominado de *Address Resolution Protocol (ARP) spoofing*. Este processo é efectuado da forma seguinte: o ARP identifica o endereço *Medium Access Control* (MAC) para um determinado endereço IP. Sempre que um dispositivo pretender comunicar com o seu par numa rede IP, envia um pedido de ARP por difusão na rede, solicitando o endereço MAC para o correspondente endereço IP. Um atacante poderá responder com o endereço MAC do seu dispositivo como o que identifica o endereço IP do pedido. A partir desse momento todas as comunicações entre os dispositivos são primeiro encaminhadas para o dispositivo do intruso, permitindo a manipulação da informação e/ou o registo da mesma. Exemplo de ferramentas que permitem realizar ataques deste tipo são o *ettercap* [20] e o *Cain e Abel* [30].

Rogue Access Points

A facilidade com que é possível instalar uma rede sem fios (não é necessário utilizar cablagens) pode, em determinadas situações, constituir um grande problema. Por exemplo, é muito simples instalar um ponto de acesso (*Access Point* - AP). Este "*rogue AP*" é um ponto de entrada não autorizado a uma rede e origina o risco de utilização não autorizada dos recursos de uma rede: é um ataque do tipo MITM.

Captura de informação

Este é um ataque à confidencialidade. O atacante tenta ler e obter informação que circula na rede sem fios. Este tipo de ataque é difícil de detectar. O termo que normalmente se aplica a este tipo de ataque é o de *eavesdropping*. Para que o atacante consiga obter informação que circula na rede, deverá capturar os pacotes e, através de um analisador de pacotes, descodificar a informação que eles contêm. Após a descodificação dos pacotes pelo analisador de pacotes, o atacante pode obter informação relativa à autenticação de um utilizador, tais como nomes de utilizador e palavras-chave. Depois de obtida esta informação, o atacante pode obter acesso não autorizado a recursos da rede. As sessões de *File Transfer Protocol* (FTP) e de *Telnet*, bem como as mensagens de correio electrónico quando não se utiliza mecanismos de protecção como o *Secure Sockets Laye*r (SSL) ou o *Pretty Good Privacy* (PGP), são exemplos de informação passível de ser recuperada através da captura de

pacotes. Existe um grande número de ferramentas que permitem realizar este tipo de ataques, tais como o *ethereal* [17], o *Dsniff* [18], o *sniffer wireless* [16] e o *Airopeek* [15].

Utilização de serviços de forma não autorizada

Nesta ameaça um utilizador sem autorização utiliza os recursos de uma rede. Pode utilizar a rede para aceder à *Internet*, para conseguir aceder a outras redes, ou como porta de saída de ataques a outras redes. Este tipo de ataque é muito semelhante ao ataque de captura de informação e pode ser considerado um ataque à confidencialidade, à integridade e à disponibilidade.

Negação de Serviço (Denial of Service - DoS)

Este ataque tem como objectivo impedir que os utilizadores acedam aos recursos de uma rede. A forma mais fácil de efectuar este ataque em redes sem fios é a de criar interferências rádio, seja através de fornos micro-ondas ou através de outros equipamentos. É também possível realizar ataques de negação de serviço através da sobrecarga de recursos. Os recursos que mais são atingidos por este tipo de ataque são a largura de banda de uma interface, as memórias e a capacidade de processamento.

O envio de um número extremamente elevado de pacotes *Internet Control Message Protocol* (ICMP) para um determinado alvo possibilita o ataque à largura de banda de uma interface. Este tipo de ataque, para além de consumir largura de banda, também exige maior capacidade de processamento ao dispositivo. As ferramentas que permitem realizar este tipo de ataques são o *pingflood* [89], o *surf* [88] e o *fraggle* [90].

Outra forma de ataque de negação de serviço é criar um número elevado de sessões TCP num determinado alvo, esgotando os recursos desse alvo. Este tipo de ataque tem como objectivo principal não permitir que utilizadores legítimos obtenham acesso aos recursos de uma rede. O *neptune* [91] e o *synk4* [92] são exemplos de ferramentas que permitem implementar este tipo de ataque.

Uma outra forma de implementar um ataque do tipo negação de serviço é através da utilização de bombas de *e-mail*. As bombas de *e-mail* inundam uma rede com um número extremamente elevado de *e-mails*, monopolizando os recursos da rede e de processamento de processador em determinados dispositivos.

Uma ferramenta como o *ettercap* [20] permite também efectuar ataques de negação de serviço. Esta ferramenta permite escolher que tipo de serviço vai ser negado numa rede ou numa sessão.

Como esta ameaça é efectuada à disponibilidade da rede, é muito difícil de prevenir em redes sem fios. Sendo assim, não é aconselhado o uso de redes sem fios quando a disponibilidade é importante.

Capítulo 3

Conceitos de segurança em redes

A utilização de mecanismos de segurança baseados em nomes de utilizador e palavras-chave mostram-se inadequados para a protecção de dados sensíveis e/ou de aplicações numa rede de computadores. A necessidade de uma protecção mais eficaz desses recursos passa pela introdução da utilização de sistemas de criptografia, de cartões inteligentes e de sistemas biométricos.

Dois sistemas que pretendam efectuar transacções seguras deverão recorrer aos serviços de autenticação, confidencialidade, integridade dos dados e não-repúdio. A definição dos mecanismos que implementam estes serviços é função da política de segurança a definir para a rede. O facto de ser desejável a implementação de todos esses serviços cria a necessidade de sistemas matemáticos e lógicos mais complexos, que exigem mais processamento nos sistemas e que, por outro lado, diminuem o desempenho de uma rede de comunicações. No entanto, para situações em que a segurança é crucial, a complexidade é diluída pelas vantagens inerentes do reforço de segurança.

Este capítulo começa por introduzir o conceito de criptografia na secção 3.1. A secção 3.1.1 apresenta uma abordagem aos sistemas de criptografia de chave simétrica, com a descrição do algoritmo criptográfico *Triple Data Encryption Standard* (3DES), do algoritmo *Rivest Code Four* (RC4), e do algoritmo *Advanced Encryption Standard* (AES). A secção 3.1.2 faz uma abordagem aos sistemas de criptografia de chave pública, com a descrição do funcionamento do algoritmo criptográfico *Rivest Shamir Addleman* (RSA) e do algoritmo de negociação de chaves *Diffie-Hellman* (DHE). Os serviços de autenticação, confidencialidade, integridade e não-repúdio são objectos de estudo da secção 3.2. Esta secção faz também uma introdução às funções de *hash*, referindo os algoritmos *Message Digest 5* (MD5) e *Secure Hash Algorithm* (SHA-1), as assinaturas digitais e as *One Time Passwords* (OTP). Na secção 3.3 estuda-se o problema da distribuição de chaves referindo-se os diversos métodos possíveis e a utilização de certificados digitais. A secção 3.4 apresenta o conceito de Autoridade Certificadora (AC). Este capítulo finaliza, na secção 3.5, com um pequeno sumário geral dos temas abordados.

3.1. Criptografia

Para que duas entidades comuniquem de forma confidencial é necessário utilizar criptografia. A palavra criptografia deriva das palavras gregas: "*kryptos*", que significa oculto, e "*graphos*", que significa escrever. Criptografia significa então escrever de forma oculta, ou seja, escrever de uma forma que as entidades não autorizadas não tenham acesso à informação. A criptografia tem evoluído significativamente: actualmente existem vários tipos de algoritmos com complexidades diferentes e que disponibilizam níveis diferentes de segurança.

A criptografia é actualmente um conceito muito importante nas redes de computadores. Como exemplo, sem criptografia não seria possível implementar um sistema de comércio electrónico: a transferência da informação relativa aos cartões de crédito, método muito utilizado para efectuar pagamentos em sistemas de comércio electrónico, deve ser efectuada de forma confidencial. Embora a criptografia não seja a condição necessária e suficiente para se implementarem mecanismos de segurança numa rede de computadores, é essencial a sua existência [2].

A criptografia é o processo de transformar a informação de forma a ser apenas legível pelo receptor da mesma. A decifra é o processo de transformar a informação cifrada em informação legível. Um algoritmo criptográfico, também denominado algoritmo de cifra, é uma função utilizada para cifrar. Existem dois tipos de cifras, as cifras de fluxo (*stream ciphers*) e as cifras de bloco (*block ciphers*). No primeiro caso, todos os *bits* de uma mensagem são sujeitos à função matemática que é utilizada pelo algoritmo criptográfico; no segundo caso, apenas os blocos com um determinado número de *bits* são sujeitos à função matemática. Na maioria dos casos são utilizadas duas funções relacionadas, uma para cifrar e outra para decifrar. A segurança dos algoritmos de cifra é baseada na dificuldade computacional de inverter uma determinada função de cifra que é pública. O utilizador legal possui um conjunto de dígitos, chave, que torna este processo (decifra) eficiente.

As próximas secções indicam a forma de utilização das chaves para cifra e decifra da informação. Os sistemas criptográficos existentes diferem apenas na forma como são utilizadas as chaves. Os sistemas que utilizam a mesma chave para cifrar e decifrar os dados são conhecidos por sistemas criptográficos de chave simétrica ou clássicos, esta chave deverá ser mantida secreta por ambas as entidades intervenientes que participam no sistema. Os sistemas que utilizam um par de chaves denominam-se sistemas criptográficos de chave pública. Nestes sistemas, uma das chaves é mantida secreta (conhecida por chave privada) por

cada uma das entidades. Esta é a chave destinada a decifrar os dados. A outra chave é tornada pública, sendo acessível a qualquer outra entidade, e será utilizada para cifrar os dados.

3.1.1. Criptografia de chave simétrica

Na criptografia de chave simétrica, a chave de cifra pode ser calculada através da chave de decifra e vice-versa. Na maioria dos algoritmos simétricos, a mesma chave é utilizada para a cifra e para a decifra. A Figura 3.1 exemplifica o processo de criptografia de chave simétrica. Como ilustra a figura, a mesma chave é utilizada pelo emissor para cifrar os dados e pelo receptor para decifrar a informação recebida e obter os dados originais.

Figura 3.1 - Criptografia de chave simétrica.

As implementações de criptografia de chave simétrica podem ser muito eficientes, de forma a que os utilizadores não experimentem nenhum atraso significativo como resultado dos processos de cifra e de decifra da informação. Este tipo de criptografia fornece também um nível de autenticação significativo, já que a informação cifrada com uma dada chave não pode ser descodificada com outra chave. Assim, enquanto as duas entidades mantiverem secreta a chave que utilizam para cifrar as comunicações, cada entidade pode ter a garantia que está a comunicar com a outra entidade. Caso uma das entidades, ou ambas, recebam informação sem sentido, poderá significar que a chave foi descoberta, e que a entidade que a descobriu está a comunicar de forma não autorizada.

A criptografia de chave simétrica só é efectiva se a chave for mantida em segredo pelas partes envolvidas na comunicação. Se alguma entidade descobre a chave secreta, a confidencialidade e a autenticidade das comunicações é comprometida. Uma entidade não autorizada com uma chave válida pode, não só decifrar as mensagens, como também cifrar novas mensagens e enviá-las como sendo uma das partes que originalmente utiliza a chave.

De seguida apresentam-se os algoritmos criptográficos de chave simétrica 3DES, RC4 e AES.

3.1.1.1. Algoritmo criptográfico 3DES

O *Data Encryption Standard* (DES) [136] é um algoritmo simétrico que cifra blocos de 64 *bits*, utilizando uma chave secreta do mesmo tamanho, onde apenas 56 *bits* dessa chave são utilizados para cifrar a mensagem; os 8 *bits* restantes servem para calcular a sua

integridade utilizando o *bit* de paridade. Este algoritmo faz parte do conjunto das cifras de blocos.

A primeira fase do algoritmo DES consiste em calcular 16 novas sub-chaves de 48 *bits* a partir da chave original de 56 *bits*. O algoritmo que efectua a codificação DES apresenta-se na Figura 3.2. O DES opera sobre blocos de dados de 64 *bits*, que após uma permutação inicial é dividido em dois blocos (mensagens) de 32 *bits*; o bloco da direita (***R***) e o bloco da esquerda (***L***). O cálculo é efectuado através de tabelas de permutações e de rotações à esquerda. Para cifrar a mensagem, são efectuadas várias permutações por tabelas em 16 iterações. Cada iteração expande a metade da mensagem (32 *bits*) para 48 *bits*, efectua uma operação de XOR com uma das sub-chaves (***k***), comprime o resultado utilizando 8 *S-boxes* (sistemas que efectuam a substituição dos *bits*) e efectua uma permutação. Ao conjunto destas quatro operações designa-se função ***f***. O resultado da função é então combinado, através de outra operação XOR, com o bloco da esquerda. O algoritmo conclui com uma permutação que é a inversa da inicial. Para efectuar a decifra é utilizado o mesmo processo do algoritmo de cifrar, invertendo-se a ordem das sub-chaves em cada iteração.

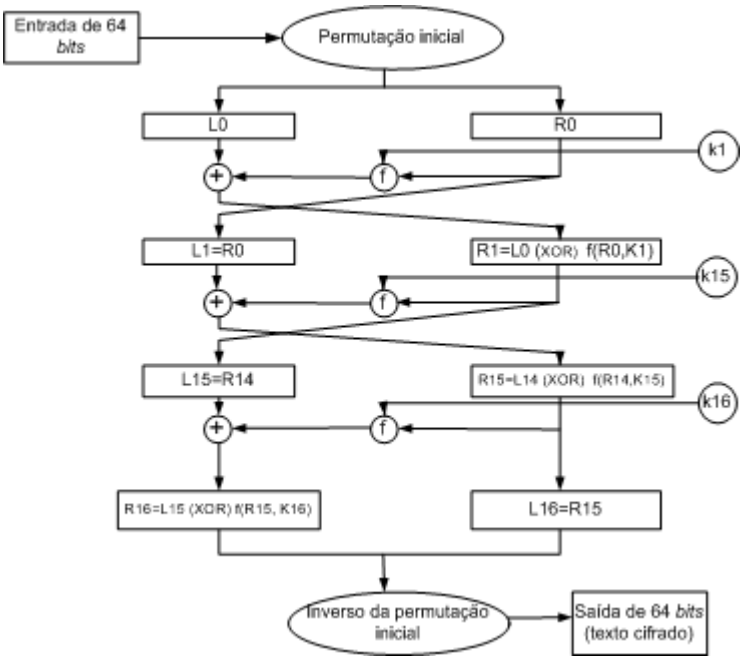


Figura 3.2 - Algoritmo criptográfico DES.

O 3DES é uma variante do DES, na qual se utilizam processos de cifra e de decifra para uma mesma mensagem (ver Figura 3.3). Como o algoritmo 3DES é também um algoritmo simétrico, tanto o receptor como o emissor devem ter na sua posse as chaves de cifra e de decifra utilizadas em todos os passos do algoritmo. O primeiro passo do 3DES consiste em cifrar os dados com uma chave k1, o segundo passo consiste em decifrar o resultado do primeiro passo com uma nova chave k2. Para finalizar o algoritmo 3DES, o

resultado do passo anterior é novamente cifrado com uma nova chave k3. O objectivo destas operações é o de reforçar a segurança dos dados, já que se torna muito mais difícil recuperar os dados originais se não se possuírem as três chaves de cifra.

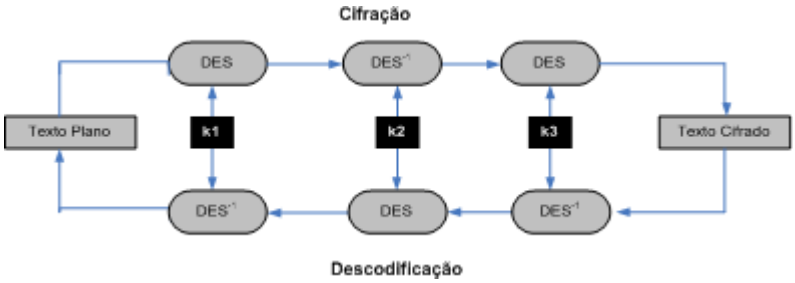


Figura 3.3 - Algoritmo criptográfico 3DES.

3.1.1.2. Algoritmo criptográfico RC4

O RC4 [137] é uma cifra de fluxo com tamanho de chave variável. Este algoritmo de chave simétrica foi desenvolvido por *Ron Rivest* da empresa de segurança *Rivest Shamir Addleman* (RSA) *Security* [137], e é utilizado em inúmeras aplicações. O RC4 é um algoritmo reversível, o que significa que o mesmo algoritmo é utilizado para a cifra dos dados e para a recuperação dos dados originais previamente cifrados. Uma das suas utilizações mais importantes é no mecanismo de segurança da maioria dos sistemas de comércio electrónico, o *Secure Sockets Layer* (SSL), ou *Transport Layer Security* (TLS) [93]. O RC4 também é utilizado no *Wired Equivalent Privacy* (WEP) e em aplicações de protecção de mensagens de correio electrónico (cifrar e assinar mensagens de correio electrónico).

O RC4 é uma cifra de fluxo aditiva binária; utiliza uma chave de tamanho variável, com valores entre 8 e 2048 *bits* em múltiplos de 8 *bits*. O núcleo do algoritmo consiste numa função geradora de fluxo de chaves. Esta função gera uma sequência de *bits* que depois são combinados com o texto através de uma operação XOR. A decifra consiste em regenerar o fluxo de chaves e efectuar uma operação de XOR com o texto cifrado. Outra função importante presente no algoritmo é a função de inicialização. Esta função aceita uma chave de tamanho variável e utiliza-a para criar o estado inicial do gerador de fluxo de chaves. Esta fase também é conhecida como fase de reserva de chaves.

O RC4 pertence a uma classe de algoritmos parametrizados para o tamanho do bloco que constitui o tamanho das palavras utilizadas. Este parâmetro $\boldsymbol{n}$ é o tamanho das palavras utilizadas no algoritmo, tomando normalmente o valor 8. O estado interno do RC4 consiste numa tabela de palavras de tamanho $\boldsymbol{2^n}$ e em contadores do tamanho das palavras ($\boldsymbol{i}$ e $\boldsymbol{j}$). A tabela é conhecida por *S-Box* e contém sempre uma permutação dos $\boldsymbol{2^n}$ valores possíveis da palavra.

A Figura 3.4 representa o algoritmo de reserva de chaves que aceita como entrada a chave de tamanho 1 *byte* armazenada na variável ***k***. A primeira parte do algoritmo corresponde à permutação de identidade em ***S***, utilizando a chave como valor de troca para produzir uma nova permutação dependente da chave. Como a única acção em ***S*** é a troca de dois valores, mantém-se o facto de ***S*** conter apenas permutações.

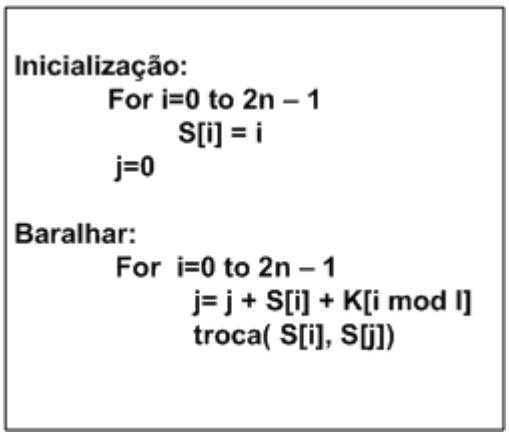


Figura 3.4 - Algoritmo de reserva de chaves do RC4.

O gerador de fluxo de chaves é apresentado na Figura 3.5. O algoritmo consiste em baralhar de forma contínua a permutação armazenada em ***S***, e escolher como saída, a cada instante, um valor diferente da permutação ***S***. Uma iteração do RC4 tem como saída uma palavra de ***n*** *bits* como fluxo de chaves, que poderá ser associada ao texto plano com uma operação de XOR, produzindo assim o texto cifrado. Para se obter novamente o texto plano é necessário regenerar as chaves e efectuar novamente todo o processo, sendo agora as tabelas constituídas pelo texto cifrado. A cifra RC4 é aproximadamente dez vezes mais rápida do que o DES. No entanto, para chaves de cifra de tamanho inferior a 128 *bits* a segurança do RC4 é muito inferior à do DES; apenas para valores de chaves de cifra superiores a 128 *bits* é que a segurança do RC4 se assemelha à do DES.

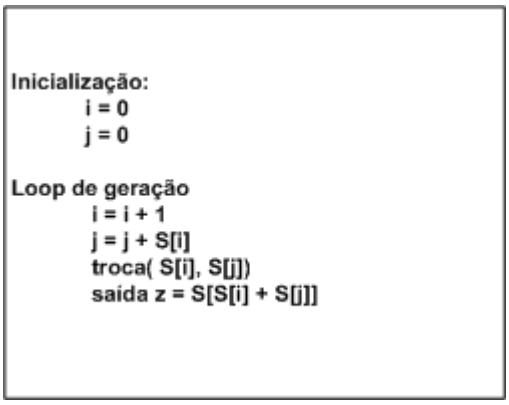


Figura 3.5 - Algoritmo de geração de chaves do RC4.

3.1.1.3. Algoritmo criptográfico AES

O *Advanced Encryption Standard* (AES) [138] é um algoritmo de cifra de blocos. O AES baseia-se no algoritmo matemático de *Rijndael*, inventado por *Joan Daeman* e *Vincent Rijmen*. Este algoritmo utiliza operações lógicas, de forma a combinar uma chave e blocos de dados não cifrados de 128 *bits* para produzir um bloco de dados cifrado. O AES, tal como o RC4, é um algoritmo reversível. Sendo assim, os blocos cifrados e não cifrados têm exactamente o mesmo tamanho.

O algoritmo AES efectua a conversão de um bloco único de 128 *bits* de uma forma segura (até hoje ainda não se verificou nenhum ataque a este algoritmo) [139]. O algoritmo de *Rijndael* possibilita a selecção do tamanho de bloco e do tamanho da chave. Cada bloco e cada chave podem ter 128, 192 ou 256 *bits*. Quando o NIST[1] adoptou este algoritmo para o AES, especificou apenas 128 *bits* para o tamanho dos blocos de dados, mas manteve a escolha dos três tamanhos para as chaves.

O AES pode ser utilizado para cifrar e decifrar blocos de dados de tamanho fixo. No entanto, as mensagens não geram blocos de tamanho fixo. Por exemplo, os dados numa rede sem fios são enviados utilizando blocos de tamanho variável, tipicamente de 512 a 1200 *bits* por cada trama. Assim sendo, para ser possível utilizar um algoritmo de cifra de blocos como o AES é necessário definir um mecanismo que converta as mensagens de comprimento variável em sequências de blocos de dados de tamanho fixo. Existem vários modos de operação que podem ser utilizados com o AES; no entanto, apenas será descrito de uma forma geral o modo *Counter Mode,* que é utilizado na nova norma de segurança 802.11i para as redes sem fios.

O AES disponibiliza também um método de verificação da autenticidade dos dados para garantir ao receptor que a mensagem não é modificada em trânsito. Para suportar este método, é introduzido um *Message Integrity Code* (MIC). O MIC não é mais que um bloco de dados cifrado, com o tamanho de 8 *bytes*, que é anexado à mensagem. O MIC é obtido pela cifra dos dados da trama a enviar em conjunto com o endereço *Medium Access Control* (MAC) do emissor e do receptor. Quando o receptor recebe a mensagem, descodifica-a e cifra-a novamente para obter um novo MIC, e compara-o com o MIC recebido; caso os MICs sejam iguais, a mensagem é legítima. Para maior eficiência é necessário que o MIC seja adicionado pelo algoritmo de cifra, fazendo todo o sentido que seja o modo de operação a definir a forma de fornecer a cifra e a autenticidade dos dados. As secções seguintes

[1] NIST – National Institute of Standards and Technology

descrevem a forma como é efectuada a cifra dos dados utilizando o modo de operação adoptado para as redes sem fios.

3.1.1.3.1. Modo de operação Counter Mode

Este modo de operação [143] não utiliza directamente a cifra de blocos AES para cifrar os dados. Pelo contrário, cifra um valor arbitrário, denominado de *counter*, e com esse resultado são efectuadas várias operações de XOR com os dados até se obter o texto cifrado. O *counter* é geralmente incrementado de um valor por cada bloco que é sucessivamente processado.

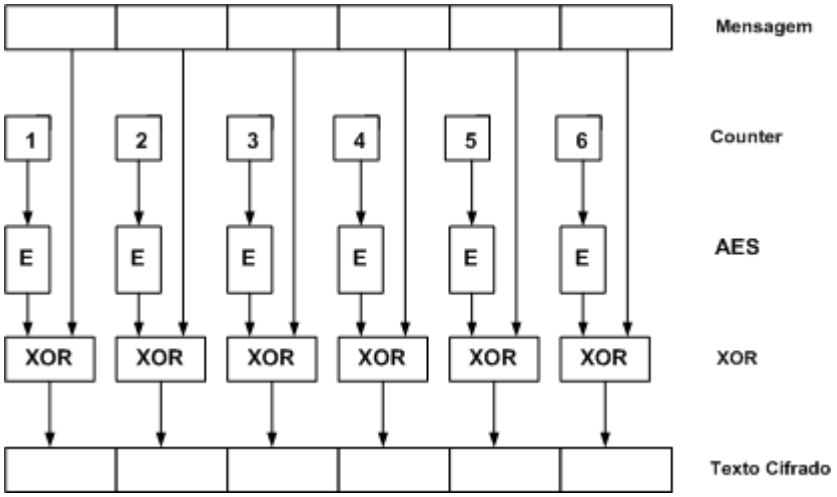


Figura 3.6 - Modo de operação *Counter Mode* do AES.

A Figura 3.6 ilustra o modo de operação do *counter mode*. A mensagem é dividida em blocos e cada bloco é associado com o resultado obtido da cifra por AES do *counter* através da operação XOR. Na Figura 3.6, o *counter* começa com o valor 1 e termina em 6 (na prática o *counter* pode começar com qualquer valor de forma arbitrária e pode também ser incrementado por qualquer outro valor). A entidade que recebe os dados cifrados e que pretende obter a informação original tem de conhecer o valor inicial do *counter* e as regras para o seu incremento. Normalmente, o *counter* é inicializado por um *nonce*[2], que é diferente para cada mensagem sucessiva. Este mecanismo evita, à partida, a repetição de blocos de dados cifrados.

Este modo oferece algumas propriedades interessantes: a decifra é efectuada utilizando o mesmo processo que o da cifra, já que, se se efectuar o XOR do mesmo valor duas vezes, obter-se-á o valor original. Deste modo, os sistemas apenas têm de implementar o processo AES de cifrar os dados. Outra propriedade importante é a possibilidade de se efectuar a cifra

[2] Nonce – conjunto de caracteres gerado aleatoriamente

dos dados em paralelo. Assim, não existe problema se a mensagem não for dividida num número exacto de blocos.

O *Counter Mode* é utilizado há bastante tempo e encontra-se num estágio de conhecimento bastante avançado no mundo da criptografia. No entanto, este modo não fornece nenhum processo de autenticação da mensagem, condição essa que é de extrema importância para as redes sem fios. De seguida apresenta-se o método *Counter Mode with cipher Block Chaining Message Authentication Code* (CCM CBC-MAC) [140] que permite colmatar esta lacuna.

3.1.1.3.2. Método Counter Mode with Cipher Block Chaining Message Authentication Code (CCM CBC-MAC)

O modo de operação CCM CBC-MAC [140] foi especialmente desenvolvido para utilização na nova norma de segurança 802.11i (este modo também é aplicável a outros sistemas). O CCM CBC-MAC foi criado por *Doug Whiting*, *Russ Housley* e *Niels Ferguson* do grupo de desenvolvimento IEEE802.11i.

O CCM CBC-MAC utiliza o *Counter Mode* em conjunto com um método de autenticação das mensagens, o CBC. O CBC é utilizado para produzir um código de integridade da mensagem, o MIC. O MIC é conhecido na comunidade criptográfica por código de autenticação da mensagem, *Message Authentication Code* (MAC).

O CBC-MAC é uma técnica normalizada. Esta técnica funciona da seguinte forma. Inicialmente utiliza-se o AES para cifrar o primeiro bloco da mensagem. De seguida, efectua-se o XOR do resultado da operação anterior com o segundo bloco da mensagem, e utiliza-se o AES para cifrar o seu resultado. Sequencialmente, efectua-se o XOR do resultado da operação anterior com o próximo bloco da mensagem e utiliza-se o AES para cifrar o seu resultado. Estes passos são efectuados sucessivamente até ao último bloco da mensagem. Como resultado final obtém-se um único bloco de 128 *bits*, que combina todos os dados da mensagem. Se um ou mais *bits* forem alterados na mensagem, o resultado sairá completamente diferente.

O CBC-MAC é um mecanismo simples mas não pode ser paralelizado: as operações de cifra deverão ser efectuadas de forma sequencial. Outro factor relevante na utilização do CBC-MAC é o facto de só poder ser utilizado em mensagens que originem um número exacto de blocos. Em situações em que tal não aconteça deverá ser utilizado o processo de enchimento (*padding*), que acrescenta à esquerda da mensagem o número suficiente de zeros que tornem a mensagem divisível num número exacto de blocos.

3.1.2. Criptografia de chave pública

A criptografia de chave pública, também conhecida por criptografia de chave assimétrica, utiliza um par de chaves – uma chave pública e uma chave privada – associadas a uma entidade que necessita de autenticar electronicamente a sua identidade, assinar ou cifrar os dados. Cada chave pública é do conhecimento público e a correspondente chave privada é mantida em segredo. Os dados cifrados com uma determinada chave pública só podem ser decifrados com a correspondente chave privada. A Figura 3.7 ilustra de forma simples o processo da criptografia de chave pública. Como indicado na figura, o emissor utiliza a chave pública do receptor para cifrar os dados. Como foi anteriormente referido, esta chave é acessível a qualquer entidade que pretenda comunicar de forma segura com o receptor, sendo disponibilizada publicamente. O receptor utiliza a sua chave privada (esta chave é apenas conhecida pelo receptor) e decifra a informação recebida obtendo os dados originais.

As implementações de criptografia de chave pública mais utilizadas baseiam-se em algoritmos patenteados pela RSA *Security*. O algoritmo de troca de chaves *Diffie-Hellman* (DHE) [134] é muito utilizado para efectuar a negociação e a troca de uma chave secreta entre dois sistemas através de um canal não seguro.

Figura 3.7 - Criptografia de chave pública.

A chave pública pode-se distribuir de forma livre, pois apenas quem possui a chave privada pode ler os dados cifrados com a chave pública; é esta característica que permite utilizar uma chave para cifrar os dados (chave pública), e uma chave totalmente diferente (chave privada) para decifrar os dados. Outra característica importante é o facto de ser computacionalmente impraticável determinar a chave privada, dada a chave pública e o algoritmo de decifra.

A utilização inversa da criptografia de chave púbica funciona da mesma forma. A informação cifrada com a chave privada pode ser unicamente descodificada com a chave pública correspondente. A utilização da criptografia de chave pública é muito útil para assinar os dados com a assinatura digital do emissor (este é um requisito importante para o comércio electrónico e outras aplicações comerciais da criptografia).

3.1.2.1. Algoritmo criptográfico RSA

O algoritmo criptográfico mais utilizado em criptografia de chave pública é o RSA [141]. Este algoritmo envolve a multiplicação de números primos grandes de forma a gerar as chaves de cifra. Cada entidade presente num sistema de comunicação deverá criar uma chave pública RSA e a correspondente chave privada. Para se gerarem as duas chaves são necessários dois números primos p e q (com 100 ou mais dígitos), escolhidos de forma aleatória. A situação aqui descrita refere-se apenas à criação das chaves por uma das entidades; caso seja necessário criar mais chaves para mais entidades, todo o processo aqui descrito deverá ser repetido. Para máxima segurança, os dois números primos deverão ter o mesmo tamanho. De seguida, determinam-se o valor $n=p.q$ e a função de *Euler* $\Psi=(p-1)(q-1)$. Além destes valores, escolhe-se um número e, de forma a que e e Ψ sejam números primos relativos (eq. 3.1).

$$e.d \equiv 1 \bmod (p-1)(q-1) \qquad \text{(eq. 3.1)}$$

$$d = e^{-1} \bmod (p-1)(q-1) \qquad \text{(eq. 3.2)}$$

A chave pública da entidade será composta pelo número e conjuntamente com o número n (chave pública (e, n)); este par de valores pode ser colocado num servidor *web* ou noutro tipo de servidor de acesso público. O número d é a chave privada, sendo mantida em lugar secreto e de forma confidencial por cada entidade. É agora possível cifrar e decifrar uma mensagem. Para que uma entidade B comunique de forma segura com uma entidade A, utilizando o algoritmo, deverá primeiro obter a chave pública (n_A, e_A) da entidade A. Poderá obter a chave pública acedendo por exemplo ao servidor *web*. Depois de obtida a chave pública, para cifrar uma mensagem M é apenas necessário que a entidade B efectue a seguinte operação, em que o resultado, C, é a mensagem cifrada.

$$C = M^{e\_A} \bmod n\_A \qquad \text{(eq. 3.3)}$$

Depois de receber a mensagem cifrada C, a entidade A utiliza a sua chave privada d_A para obter a mensagem original M da seguinte forma:

$$M = C^{d\_A} \bmod n\_A \qquad \text{(eq. 3.4)}$$

O nível de segurança elevado obtido com o RSA advém do facto de não ser conhecido nenhum algoritmo eficiente para a factorização. Como os números primos são multiplicados antes de serem tornados públicos é, actualmente, muito difícil factorizá-los num espaço de tempo razoável. Assim sendo, a probabilidade de descodificar esta chave é muito pequena. As experiências que têm sido realizadas demonstram que seriam necessários milhares de anos de processamento com a utilização de números primos de 100 dígitos para factorizar o resultado (utilizando-se para tal os melhores algoritmos de factorização conhecidos) [3]; recuperar o

texto decifrado com a chave pública e o texto cifrado é o equivalente a factorizar o produto dos dois números primos.

3.1.2.2. Algoritmo de *Diffie-Hellman*

O algoritmo de *Diffie-Hellman* (DHE) [134] é um algoritmo de negociação de chaves de cifra, que permite que duas entidades negoceiem uma chave secreta através de um canal de comunicação não seguro. Este algoritmo foi desenvolvido por *Diffie* e *Hellman* e publicado em 1976.

Este algoritmo define no seu funcionamento dois parâmetros p e g. Ambos os valores são públicos e podem ser utilizados por qualquer utilizador. O parâmetro p é um número primo de valor elevado. O parâmetro g (denominado de gerador) é um número inteiro de valor inferior a p, e que satisfaz a seguinte condição: para cada número n entre 1 e p-1 *inclusivé*, existe um valor k que satisfaz a seguinte igualdade:

$$n = g^k \bmod p \qquad \text{(eq. 3.5)}$$

O processo de negociação da chave secreta entre duas entidades é o seguinte: inicialmente as duas entidades geram, de forma privada, dois números inteiros aleatórios a e b. De seguida, as duas entidades geram os seus valores públicos, utilizando os valores p e g (que também são públicos) e os seus valores privados. Como resultado obtém-se:

$$g^a \bmod p \qquad \text{(eq. 3.6)}$$

$$g^b \bmod p \qquad \text{(eq. 3.7)}$$

Depois de obtidos esses valores, as entidades trocam-nos entre si. Finalmente, a entidade emissora determina:

$$g^{ab} = (g^b)^a \bmod p \qquad \text{(eq. 3.8)}$$

E a entidade receptora determina:

$$g^{ba} = (g^a)^b \bmod p \qquad \text{(eq. 3.9)}$$

Como $g^{ab} = g^{ba} = k$, as entidades podem agora comunicar de forma segura, utilizando como chave de cifra k.

Uma variante deste protocolo, conhecido por algoritmo autenticado de *Diffie-Hellman*, foi desenvolvido por *Diffie*, *van Oorchot* e *Wiener* em 1992 [135]. Esta variante utiliza assinaturas digitais e certificados para a autenticação mútua entre as entidades. Nesta variante, antes de se efectuar o algoritmo DHE, as entidades devem obter um par chave privada/chave pública e o certificado da chave pública. Durante o processo de execução do algoritmo DHE, a entidade emissora calcula para determinadas mensagens a assinatura digital, protegendo o valor ($g^a \bmod p$). A entidade receptora efectua o mesmo procedimento. Este mecanismo

permite proteger a negociação das chaves contra ataques de *Man-In-The-Middle* (MITM), já que uma terceira entidade não pode falsificar as assinaturas digitais.

3.2. Serviços de autenticação, confidencialidade, integridade e não-repúdio

A implementação de mecanismos de segurança em redes sem fios requer a protecção de informação contra ataques de personificação, modificação de conteúdos, modificação de sequência e modificação da validade. A introdução dos serviços de autenticação, confidencialidade, integridade e não-repúdio permite resolver e/ou prevenir os ataques anteriormente mencionados. A implementação desses serviços só é possível, em redes de computadores sem fios, mediante a utilização de um conjunto de protocolos específicos. As próximas secções descrevem os conceitos adjacentes aos protocolos que permitem introduzir os serviços referidos acima.

3.2.1. Autenticação

A forma mais simples de garantir uma comunicação segura entre duas entidades é recorrer ao serviço de autenticação. A forma de autenticação mais simples limita-se apenas a autenticar mensagens entre entidades através de endereços TCP/IP, através do envio da identidade dos elementos participantes na comunicação, ou através do envio de palavras-chave. No entanto, estes mecanismos de autenticação não são fiáveis, pois qualquer intruso poderá obter a informação de autenticação. Um serviço de autenticação mais complexo, para além de permitir que duas entidades iniciem a troca de informação, encarrega-se também de verificar as identidades das entidades envolvidas no processo de comunicação. De seguida apresentam-se os métodos de autenticação com base em criptografia.

3.2.1.1. Autenticação baseada em criptografia simétrica

A melhor forma de tornar o processo de autenticação mais seguro e robusto é recorrer a técnicas de criptografia. Se ambas as entidades partilharem uma chave secreta para utilização num sistema de criptografia simétrica, também podem utilizar essa chave para se autenticarem e cifrarem a palavra-chave. A autenticação baseada em criptografia simétrica consiste no seguinte: uma das entidades envia à outra um conjunto aleatório de caracteres (*nonce*); a outra entidade cifra o *nonce* com a chave secreta e envia o resultado à entidade que iniciou a comunicação. Esta, por sua vez, decifra a mensagem recebida e compara-a com o *nonce* original; caso os valores sejam iguais a autenticação é bem sucedida. Este tipo de autenticação garante, não só a confidencialidade, mas também a integridade das

comunicações. Para reforçar a segurança, o valor do *nonce* deverá ter um tempo de vida reduzido, evitando-se assim a sua falsificação e uso indevido.

3.2.1.2. Autenticação baseada em criptografia de chave pública

A autenticação baseada em criptografia de chave pública garante, não só a autenticidade, mas também a confidencialidade quando é utilizado um algoritmo de cifra para cifrar os dados, e a integridade e o não-repúdio das comunicações através da utilização de assinaturas digitais.

A entidade que pretende ser autenticada cifra, com a sua chave privada, a palavra-chave que vai ser usada para estabelecer um canal seguro; de seguida cifra o resultado com a chave pública da entidade parceira na comunicação. Neste processo é também introduzido o conceito de assinatura digital (descrito na secção 3.2.3). Para obter a palavra-chave, a entidade receptora decifra a mensagem com a sua chave privada; de seguida decifra com a chave pública da entidade emissora (assinatura digital).

Ataques a este tipo de autenticação são praticamente impossíveis de realizar. Um atacante que pretenda obter a palavra-chave deverá obter primeiro a chave privada do receptor (se o receptor não mantiver a sua chave privada devidamente protegida, o atacante poderá obtê-la e assim conseguir obter a palavra-chave que o emissor envia ao receptor). Na situação em que o atacante se faça passar pela entidade que pretende ser autenticada, o atacante deverá cifrar a mensagem com a sua chave privada. O receptor, ao tentar decifrar a mensagem com a chave pública da entidade legítima, irá obter um conjunto de informação sem sentido, permitindo concluir que alguém de forma não legítima está a tentar obter autenticação.

3.2.2. Funções de *hash*

Uma função de *hash* é uma função que converte uma grande quantidade de dados num valor codificado dos dados de tamanho pequeno, ou mesmo reduzido; o resultado da função também é conhecido por resumo. As funções de *hash* são utilizadas em criptografia e em processamento de dados. Estas funções são essenciais na implementação do mecanismo de integridade dos dados, permitindo assim desenvolver o conceito de assinatura digital.

Uma função de *hash* tem como entrada uma mensagem $\boldsymbol{M}$ de tamanho variável, e como saída um código de *hash*, $\boldsymbol{h}(\boldsymbol{M})$, de tamanho fixo (o código de *hash* também é conhecido por resumo da mensagem). Uma função de *hash* pode ser definida muito simplesmente como uma função de compressão. As entradas da função de compressão são um bloco da mensagem e a saída do bloco de texto anterior. O resultado representa o código de

hash de todos os blocos percorridos até esse instante. O código de *hash* para um bloco $\boldsymbol{M}_i$ da mensagem pode ser definido por:

$$\boldsymbol{h}_i = \boldsymbol{f}(\boldsymbol{M}_i, \boldsymbol{h}_{i-1}) \qquad \text{(eq. 3.10)}$$

Este código de *hash*, em conjunto com o próximo bloco da mensagem, será a próxima entrada da função de *hash*. O código de *hash* da mensagem completa será o código de *hash* resultante do último bloco da mensagem. Este código final é anexado à mensagem.

Esta função baseia-se no facto de ser difícil encontrar duas mensagens diferentes que produzam o mesmo código de *hash*, ou seja, que haja colisões. As funções de *hash* são funções de um único sentido, ou seja, tendo-se $\boldsymbol{M}$ é fácil obter $\boldsymbol{h}(\boldsymbol{M})$; no entanto, se apenas se conhecer $\boldsymbol{h}(\boldsymbol{M})$, é muito difícil obter o valor de $\boldsymbol{M}$. O código de *hash* garante a integridade da mensagem já que a probabilidade de $\boldsymbol{h}(\boldsymbol{M})$ ser igual a $\boldsymbol{h}(\boldsymbol{M}')$, com $\boldsymbol{M} \neq \boldsymbol{M}'$, é muito baixa. Este método permite então detectar erros e modificações da mensagem.

Considere-se a integração das funções de *hash* em criptografia de chave simétrica. Nesta situação, a entidade emissora efectua o código de *hash* da mensagem a enviar e anexa-o à mensagem. O conjunto da mensagem com o código é depois cifrado utilizando a chave simétrica. A entidade receptora decifra a mensagem com a chave simétrica; como apenas as duas entidades conhecem a chave, garante-se autenticação. A entidade receptora calcula o código de *hash* da mensagem e compara-o com o enviado pela outra entidade; se os dois códigos forem iguais, isso significa que a mensagem não foi alterada, garantindo-se a integridade dos dados. Em situações em que os recursos computacionais disponíveis são escassos, pode-se utilizar a criptografia simétrica apenas para cifrar o código de *hash*, garantindo-se da mesma forma os serviços de autenticação e integridade.

A utilização do código de *hash* com criptografia de chave pública é um mecanismo que pode garantir a autenticação, confidencialidade, integridade e não-repúdio de uma mensagem. A entidade emissora, após calcular o código de *hash*, utiliza a sua chave privada para cifrar o código (assinatura digital) e anexa-o à mensagem original. A entidade receptora decifra o código de *hash* com a chave pública da entidade emissora e compara depois o código de *hash* obtido com o código de *hash* da mensagem; se forem iguais, significa que a mensagem não foi modificada. A integração do código de *hash* com criptografia de chave pública, efectuada desta forma, apenas garante os serviços de autenticação, não-repúdio e integridade da mensagem. Caso se pretenda garantir o serviço de confidencialidade, o conjunto formado pela mensagem e código de *hash* deverá ser cifrado pela entidade emissora com a chave pública da entidade receptora.

3.2.2.1. MD5 (*Message Digest 5*)

O MD5 [106] é um exemplo de código de *hash* de 128 *bits*. O MD5 é utilizado para garantir a integridade e autenticidade de uma mensagem.

O MD5 processa a mensagem de entrada em blocos de 512 *bits*, divididos por 16 blocos de 32 *bits*. A saída do algoritmo é um conjunto de 4 blocos de 32 *bits*, que depois de concatenados dão origem a um código de *hash* de 128 *bits*.

Suponhamos que se pretende determinar o código MD5 de uma mensagem com **b** *bits* (**b** pode ser zero, um valor arbitrariamente elevado, mas nunca poderá ser um número negativo). Para se determinar o código de *hash* utilizando o MD5 é necessário efectuar os seguintes passos. Primeiro, é efectuado o *padding* à mensagem de forma a obter uma mensagem de tamanho igual a um múltiplo de 512 menos 64 *bits* (448 **mod** 512). A adição de *bits* de *padding* é efectuada da seguinte forma: é acrescentado um único *bit* a "1" seguido de um número suficiente de zeros, de modo a tornar o seu tamanho um múltiplo de 512 menos 64. O passo seguinte consiste em acrescentar uma representação com 64 *bits* do tamanho **b** da mensagem original. Se o tamanho da mensagem **b** for superior a 2^{64}, então apenas os 64 *bits* menos significativos de **b** são utilizados. Se o tamanho da mensagem **b** for inferior a 2^{64} são utilizados todos os *bits*. No final, a mensagem resultante tem um tamanho múltiplo de 512 *bits*. De forma equivalente, esta mensagem tem um tamanho múltiplo de 16 registos com 32 *bits*. Num terceiro passo, o algoritmo inicializa um *buffer* de 128 *bits*, constituído por 4 registos (A, B, C, D) cada um com 32 *bits*. Os registos A, B, C, D são conhecidos por *chaining variables* e os seus valores são:

A=0x 01 23 45 67

B=0x 89 ab cd ef

C=0x fe de ba 98

D=0x 76 54 32 10

No quarto passo inicia-se o módulo de compressão [106]. O módulo de compressão é inicializado com os blocos de 512 *bits* da mensagem juntamente com o *buffer* das *chaining variables*. O módulo de compressão utiliza 4 ciclos com 16 operações (512 / 32 = 16) em cada ciclo. As operações resultam da utilização de funções não-lineares, de rotações e de permutações de *bits*. Estão definidas quatro funções não-lineares diferentes, sendo cada uma delas utilizada em cada um dos ciclos que compõem o módulo de compressão.

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z) \quad \text{(eq. 3.11)}$$

$$G(X,Y,Z) = (X \wedge Y) \vee (Y \wedge \neg Z) \quad \text{(eq. 3.12)}$$

$$H(X,Y,Z) = X \oplus Y \oplus Z \quad \text{(eq. 3.13)}$$

$$I(X,Y,Z) = Y \oplus (X \vee \neg Z) \qquad \text{(eq. 3.14)}$$

Onde $\oplus$, $\wedge$, $\vee$ e $\neg$ representam respectivamente as operações lógicas XOR, AND, OR e NOT. Após o algoritmo processar todos os ***L*** blocos da mensagem é produzido o código de *hash* de 128 *bits*.

O MD5 é simples de implementar e fornece uma "impressão digital" ou uma "*message digest*" de uma mensagem de tamanho arbitrário. As principais características de segurança do algoritmo são: para se encontrar duas mensagens com o mesmo código MD5, o número de operações necessárias é da ordem de 2^{64}, para descobrir uma mensagem a partir do código de *hash*, o número de operações necessárias é da ordem de 2^{128}. No entanto, o aumento da potência computacional tem permitido efectuar ataques bem sucedidos do tipo "força bruta" contra o MD5 [94].

3.2.2.2. SHA-1 (*Secure Hash Algorithm*)

O SHA [142] é muito semelhante ao MD5, sendo a sua principal diferença a obtenção de um código de *hash* de 160 *bits*. Tal como o MD5, no SHA é necessário efectuar o *padding* da mensagem, de forma a que o tamanho da mensagem seja um múltiplo de 512 menos 64 (passo 1 do MD5); da mesma forma é necessário acrescentar uma representação com 64 *bits* do tamanho da mensagem (passo 2 do MD5). Também é inicializado um *buffer*, em que este é constituído por 5 registos de 32 *bits*, que permitem produzir um código de *hash* de 160 *bits* (5x32). Esses registos têm os seguintes valores:

A=0x 67 45 23 01

B=0x ef cd ab 89

C=0x 98 ba dc fe

D=0x 10 32 54 76

E=0x c3 d2 e1 f0

O passo seguinte consiste em inicializar o módulo de compressão com os blocos de 512 *bits* da mensagem juntamente com o *buffer* anteriormente referido. O módulo de compressão utiliza 4 ciclos com 20 operações em cada ciclo, em que estas são muito semelhantes às utilizadas no MD5. Para concluir, e após o algoritmo processar todos os ***L*** blocos da mensagem, é produzido o código SHA de 160 *bits*.

3.2.2.3. MD5 vs SHA-1

Comparando o MD5 com o SHA-1, conclui-se que os seus pontos fortes se assemelham. O algoritmo SHA é mais resistente a ataques de força bruta (incluindo ataques de dicionário), já que produz um código de 160 *bits*. Ao nível do desempenho dos algoritmos,

dado o facto do código SHA processar um *buffer* de 160 *bits*, em comparação com o *buffer* de 128 *bits* do código MD5, o código SHA é mais lento, embora o grau de evolução ao nível computacional permita limitar essa diferença. Ambos os algoritmos são simples e não requerem programas de grande complexidade para os implementar.

3.2.3. Assinaturas Digitais

O processo de cifra dos dados não garante que uma mensagem seja autêntica. Num esquema de criptografia de chave pública não existe forma de saber qual a origem de uma mensagem, já que a chave é pública. Para além disso, a mensagem pode ser alterada por qualquer adversário quando em circulação. O processo de cifrar dados garante a confidencialidade das comunicações, mas não garante a origem e a integridade da mensagem. Para garantir tais condições é necessário utilizar assinaturas digitais.

O serviço de não-repúdio fornece garantias sobre a origem e o destino dos dados, de forma a proteger o destino da negação de envio por parte da origem dos dados. Assinaturas digitais através da função de *hash* protegem também a origem da negação de recepção dos dados por parte do destinatário.

A utilização das assinaturas digitais envolve dois processos, o da criação da assinatura digital e o da verificação da assinatura digital:

- Criação da assinatura digital – o emissor utiliza a sua chave privada para cifrar o resultado da função de *hash* da mensagem;
- Verificação da assinatura digital – o receptor verifica a assinatura digital, tendo como referência a mensagem original e a chave pública. O receptor determina se a assinatura digital foi criada por aquele mesmo emissor, utilizando a chave pública que corresponde à chave privada referenciada.

A Figura 3.8 exemplifica de forma simplificada os dois processos afectos às assinaturas digitais (criação e verificação).

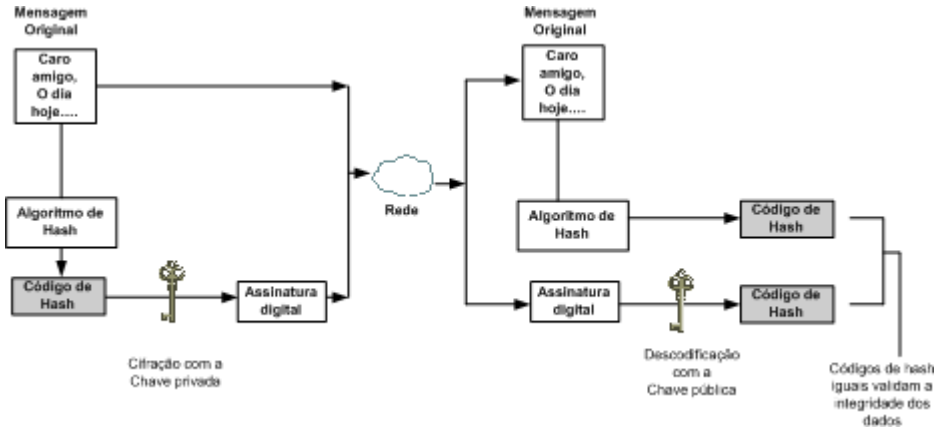


Figura 3.8 - Assinatura digital.

Como indica a Figura 3.8, o emissor cria a assinatura digital da seguinte forma. Primeiro, obtém-se um resumo da mensagem por uma função de *hash*. O resultado da função (código de *hash*) é cifrado com a chave privada do emissor, criando-se assim a assinatura digital. A assinatura digital é depois anexada à mensagem original. O envio da mensagem com a assinatura digital pode ser efectuada de duas formas: sem confidencialidade e com confidencialidade. Neste último caso, a mensagem com assinatura digital é cifrada com a chave pública do destinatário. Quando esta informação chega ao destinatário, e caso a informação chegue cifrada, o destinatário utiliza a sua chave privada para recuperar o conjunto da mensagem com assinatura digital, dando de seguida início ao processo de verificação da assinatura. Primeiro, decifra-se a assinatura digital utilizando a chave pública do emissor, obtendo o código de *hash* enviado. Depois, o receptor aplica a mesma função de *hash* utilizada pelo emissor na mensagem recebida, obtendo um novo código de *hash*. De seguida, compara os dois códigos: se os códigos forem iguais significa que o emissor que enviou a mensagem é quem diz ser quem é (a utilização de uma outra chave privada originaria um código de *hash* de valor diferente); significa também que a mensagem não foi alterada (a alteração de um único *bit* na mensagem resultaria num código de *hash* totalmente diferente).

Pode-se concluir que a utilização de uma assinatura digital garante os serviços de autenticação, integridade e não-repúdio. O único problema encontrado na utilização das assinaturas digitais é a necessidade de um mecanismo que confirme que uma determinada chave pública realmente pertence a alguém ou a determinada entidade. Este problema será discutido na secção sobre distribuição de chaves (secção 3.3).

3.2.4. *One Time Passwords* (OTP)

As OTP estão definidas no RFC 2289 [109]. O sistema OTP é um sistema de autenticação que requer uma nova palavra-chave sempre que um utilizador ou entidade se autentica. As OTP protegem as comunicações e as entidades contra ataques do tipo repetição e intercepção das palavras-chave. As palavras-chave são geradas utilizando as funções de *hash* MD5 ou MD4.

O funcionamento das OTP requer duas entidades: o gerador e o servidor. O gerador deverá produzir a OTP apropriada a partir de uma determinada palavra-chave e de informação adicional fornecida pelo servidor. O servidor deverá enviar um conjunto de informações ao gerador (os parâmetros de geração), deverá verificar a OTP recebida e deverá armazenar a sequência correspondente de numeração das OTP. Normalmente, o servidor e o gerador representam o receptor e o emissor de um sistema de comunicação, respectivamente.

O processo de geração e verificação das OTP baseia-se no seguinte conceito: a utilização de uma função de *hash* um determinado número de vezes produz uma sequência de OTPs P_i. A primeira OTP é o resultado de se processar a palavra-chave secreta ***s*** do cliente, pela função de *hash* ***N*** vezes:

$$P_0 = f^N(s) \qquad \text{(eq. 3.15)}$$

A próxima OTP é gerada processando a palavra-chave ***N-1*** vezes pela função de *hash*:

$$P_1 = f^{N-1}(s) \qquad \text{(eq. 3.16)}$$

De uma forma genérica obtém-se:

$$P_i = f^{N-i}(s) \qquad \text{(eq. 3.17)}$$

A verificação das palavras-chave segue o seguinte processo: o sistema é inicializado com P_0. Quando um cliente pretende ser autenticado, é-lhe passado o valor de *i*. O cliente fornece a próxima OTP, e o sistema guarda uma cópia desta OTP, aplicando-lhe de seguida a função de *hash*. Caso o resultado da função de *hash* não seja igual ao valor da palavra-chave armazenada no ficheiro de autenticação, o pedido de autenticação é recusado; se os dois valores forem coincidentes, o pedido de autenticação é aceite e o sistema actualiza o ficheiro de autenticação com a OTP fornecida pelo cliente. As OTP só poderão ser utilizadas *N* vezes, devendo ser geradas novas OTP assim que essas se esgotarem.

As OTP permitem melhorar o nível de segurança de um sistema, já que para um determinado número de autenticações a palavra-chave é sempre diferente, protegendo assim o sistema contra ataques de repetição e de *eavesdropping*.

3.3. Distribuição de chaves

Um dos mecanismos mais importantes ao nível da implementação de um sistema criptográfico é o processo de distribuição de chaves. Em criptografia simétrica, a chave deve ser distribuída de uma forma segura. Esta deve também ser alterada frequentemente, para limitar o seu tempo de acesso. A geração das chaves é uma função do número de entidades que as utilizam: para um grupo de *n* entidades são necessárias *n((n-1)/2)* chaves.

A distribuição das chaves pode ser simplificada se todas as entidades obtiverem as chaves de um ponto central (conhecido por *key distribution center*- KDC), sendo apenas necessário que cada entidade disponha da chave que permite comunicar de forma segura com o KDC. No entanto, as chaves utilizadas para a comunicação com o ponto central devem ser criadas de forma segura e distribuídas utilizando processos de confiança entre as duas partes.

A utilização de um KDC apresenta alguns problemas relacionados de forma directa com a própria segurança do KDC. Pode-se afirmar que um KDC é uma base de dados muito

grande com as chaves de todas as entidades/utilizadores. Um atacante que consiga ter acesso a essa base de dados pode criar novas chaves e substituí-las pelas chaves presentes no KDC. O atacante pode também substituir a chave de uma entidade pela sua, sendo assim capaz de observar as comunicações que envolvam essa entidade (ataque do tipo MITM). Sendo assim, devem ser criados mecanismos capazes de proteger de forma eficaz o KDC; devem também ser criados mecanismos que associem de forma inequívoca uma chave com a identidade de uma entidade ou utilizador.

A criptografia de chave pública é um sistema ideal para se efectuar a distribuição de chaves. Os métodos utilizados para a distribuição de chaves públicas são: anúncio público, directoria pública, autoridade, e certificados.

3.3.1. Anúncio público

Este é o método utilizado quando uma determinada entidade anuncia a sua chave pública. Qualquer método é válido para anunciar a sua chave: mensagens, página *web*, *sms*, anúncio no jornal, etc. Este método é pouco seguro, já que é possível falsificar a identidade de outra entidade.

3.3.2. Directoria pública

Neste método, uma terceira entidade é responsável pela distribuição das chaves. Essa entidade é de confiança e mantém uma directoria com um registo para cada entidade participante. Cada participante deve registar uma chave pública com a entidade responsável pela directoria. O registo deve ser efectuado de forma segura. Periodicamente, a directoria é publicada e os seus conteúdos actualizados.

Este método não é ainda totalmente seguro, pois se um atacante obtiver a chave da entidade responsável pela directoria, pode falsificar as chaves de qualquer participante e efectuar ataques de personificação.

3.3.3. Autoridade de chaves públicas

Neste método existe uma autoridade central que mantém uma base de dados com as chaves públicas de todos os participantes. Cada participante conhece a chave pública da autoridade, e apenas a Autoridade tem conhecimento da sua chave privada. Quando duas entidades ou utilizadores pretendem comunicar de forma segura, fazem-no da seguinte forma (Figura 3.9): uma das entidades envia um pedido de chave pública da outra entidade à Autoridade. O pedido contém um *nonce* (n1) gerado pela entidade que inicia a comunicação, cifrado com a chave pública da Autoridade. A Autoridade responde com uma mensagem

cifrada com a sua chave privada. A entidade que inicia a comunicação, após a recepção desta mensagem, decifra-a, utilizando a chave pública da Autoridade – autenticando assim a Autoridade. Desta mensagem, a entidade obtém a chave pública da entidade com quem pretende comunicar e o seu pedido original, verificando assim a integridade do seu pedido.

A entidade que inicia a comunicação utiliza a chave pública da outra entidade para cifrar uma mensagem com a sua identidade e um novo *nonce* (n2), que envia à outra entidade. A entidade receptora efectua, de forma equivalente, o pedido de chave pública, da entidade que inicia a comunicação. Garante-se assim, a distribuição segura das chaves públicas entre as entidades. A entidade receptora utiliza a sua chave privada para decifrar a mensagem recebida, com a identificação e o *nonce* (n2). Esta entidade envia, de seguida, uma mensagem cifrada com a chave pública da outra entidade (esta mensagem contém n2 e um novo *nonce* (n3)). Para finalizar, e após a recepção desta última mensagem, a entidade que inicia a comunicação decifra a mensagem, utilizando a sua chave privada. Envia também uma nova mensagem cifrada com a chave pública da entidade parceira contendo o *nonce* n3. A troca destas últimas mensagens permite autenticar as entidades.

Este método, embora eficaz, sofre de alguns problemas de escalabilidade: dado que cada entidade ou utilizador tem que aceder à Autoridade para obter uma determinada chave pública, a Autoridade pode tornar-se um ponto de congestionamento.

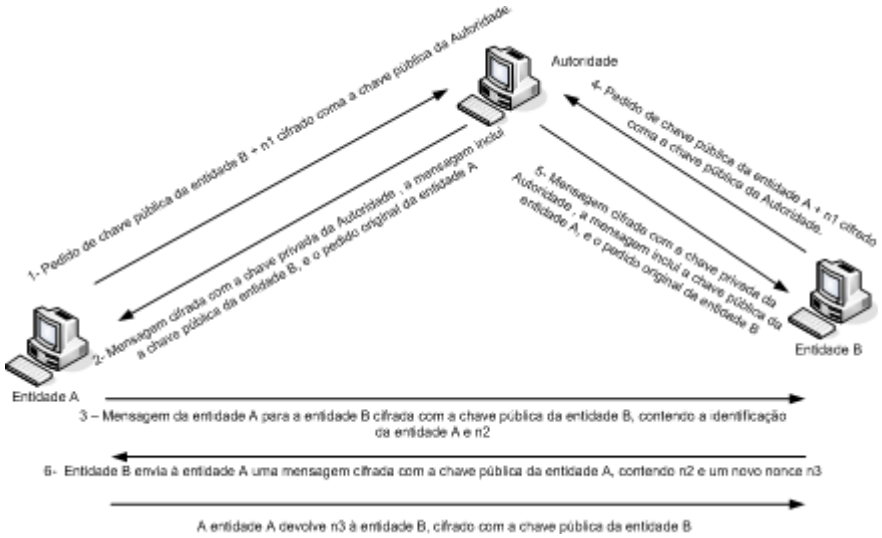


Figura 3.9 - Distribuição de chaves públicas através de uma Autoridade.

3.3.4. Certificados

Um certificado é um documento electrónico que permite identificar um indivíduo, um servidor, uma empresa ou qualquer outra entidade, e associar essa identidade com uma chave pública. Um certificado, tal como um bilhete de identidade ou um passaporte, fornece uma prova válida da identidade de alguém. A criptografia de chave pública utiliza os certificados

de forma a proteger os sistemas contra ataques de personificação. Da mesma forma que existem organismos (ou mecanismos) para emitir um bilhete de identidade ou um passaporte, também existem organismos para emitir os certificados: as Autoridades Certificadoras (AC).

Os certificados ajudam a prevenir a utilização de chaves públicas falsas em ataques de personificação. Com os certificados, apenas a chave pública certificada funciona com a correspondente chave privada possuída pela entidade identificada pelo certificado. Para além da chave pública, um certificado inclui sempre: o nome da entidade que identifica, uma data de expiração, o nome da AC que emitiu o certificado, um número de série e outras informações. Mais importante ainda, um certificado inclui sempre a assinatura digital da AC, o que permite que o certificado funcione como uma "carta de apresentação" para os utilizadores que conhecem e confiam na AC, mas não conhecem a entidade identificada pelo certificado.

Os certificados estão organizados de acordo com a especificação X.509, recomendada pelo *International Telecommunication Union* (ITU). Um certificado associa um *distinguished name* (DN) a uma chave pública. Um DN é um conjunto de pares nome-valor (tal como *uid=André*) que identifica univocamente uma entidade. O seguinte exemplo representa um funcionário de uma organização:

uid = joão, e=joão@empresa.pt, cn= João Silva, o = empresa , c = PT

em que *uid* é a identificação do utilizador, *e* é o endereço de correio electrónico, *cn* é o nome comum do funcionário (utilizador), *o* é o nome da organização ou empresa e *c* é o nome do país.

Figura 3.10 - Exemplo de um certificado.

Na Figura 3.10 ilustra-se um exemplo de um certificado criado por uma AC num formato legível para um ser humano. A Figura 3.11 representa um certificado num formato utilizado pelo *software* de interpretação dos certificados. Neste caso, o certificado encontra-se cifrado utilizando uma codificação com base 64. Esta é a forma pela qual são transmitidos os certificados entre entidades presentes numa comunicação.

Figura 3.11 - Certificado em codificação base 64.

3.4. Autoridade Certificadora (AC)

Uma AC é uma entidade que valida identidades e emite certificados. Uma AC pode ser uma entidade independente ou afecta a determinadas organizações. Qualquer entidade que utilize certificados deve manter uma colecção de certificados das AC que confia. Estes certificados determinam que outros certificados podem ser validados. No processo mais simples podem apenas ser validados certificados por uma única AC. Também é possível que um certificado de uma AC faça parte de uma cadeia de ACs, sendo cada um emitido pela AC imediatamente acima na hierarquia de certificados.

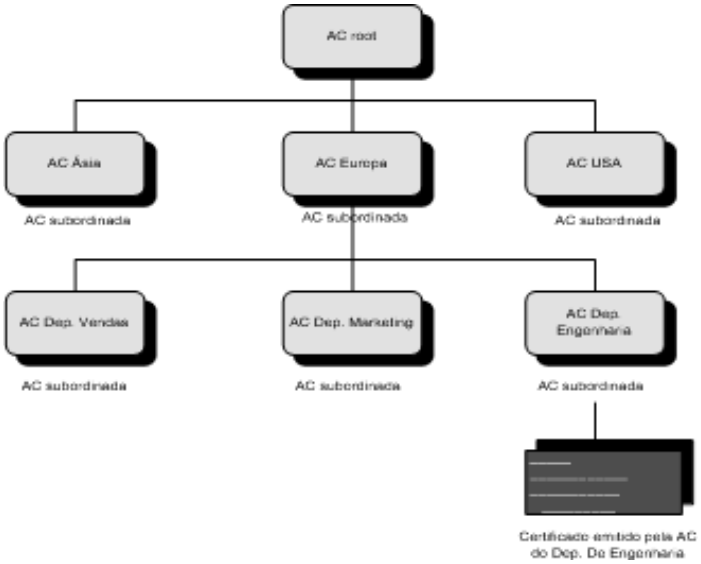


Figura 3.12 - Hierarquia de Autoridades Certificadoras.

Caso o número de certificados requeridos seja demasiado grande, ou as necessidades ao nível de definições de políticas sejam diferentes ou existam restrições a nível geográfico, é conveniente delegar a responsabilidade da emissão de certificados a diferentes autoridades. É também possível delegar responsabilidades de emissão de certificados a ACs subordinadas, criando-se assim uma hierarquia de Autoridades Certificadoras (Figura 3.12). No topo da hierarquia existe uma AC *root*. O certificado da AC *root* é auto-assinado, ou seja, o certificado é digitalmente assinado pela mesma entidade à qual o certificado identifica, a AC *root*. As Autoridades que estão directamente subordinadas à AC *root* têm os seus certificados assinados por ela. As Autoridades um nível abaixo das ACs subordinadas à AC *root*, têm os seus certificados assinados pelas ACs subordinadas de nível superior.

As hierarquias de ACs reflectem-se em cadeias de certificados. Uma cadeia de certificados é uma série de certificados emitidos por Autoridades sucessivas. A Figura 3.13 representa uma cadeia de certificados, onde se verifica que, para que um certificado emitido pela AC do Departamento de Engenharia seja válido, é necessário que exista uma cadeia de certificados que relacione os diversos certificados com a AC *root*. Esse relacionamento é

efectuado recorrendo-se a ACs subordinadas da AC *root*. Sendo assim, a AC do Departamento de Engenharia deve ter um certificado assinado por uma AC de nível superior, neste caso a AC Europa; esta por sua vez, também deve ter um certificado assinado pela AC de nível superior, ou seja a AC *root*.

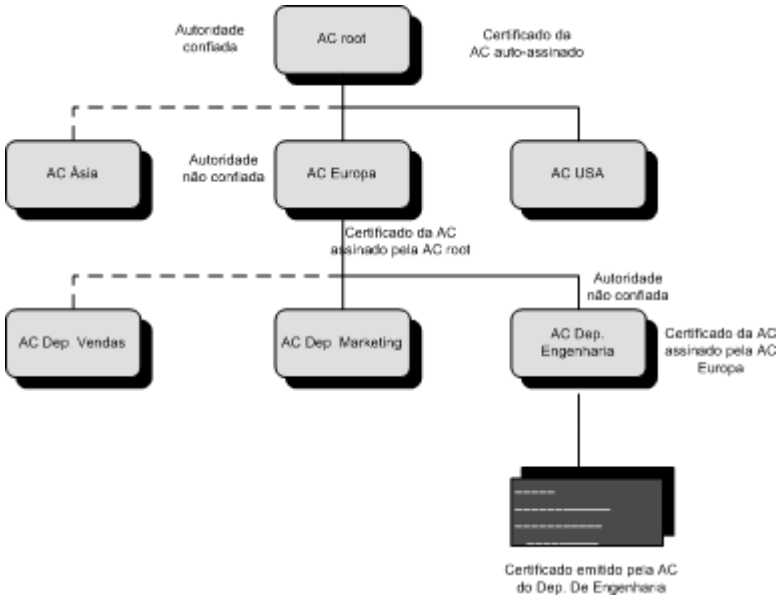


Figura 3.13 - Cadeia de certificados.

Como um certificado só é válido durante um determinado período de tempo, torna-se necessário a existência de mecanismos de renovação e de revogação dos certificados. A forma mais simples de verificar se um certificado está revogado é através da consulta da lista de certificados revogados (*certificate revogation list* – CRL), para determinar a validade ou não de um certificado, sempre que for solicitada uma autenticação baseada em certificados.

As ACs são um dos elementos mais importantes na segurança de redes e em criptografia de chave pública. Sem as ACs não é possível utilizar certificados de forma a associar inequivocamente determinada identidade com determinada chave pública.

3.5. Sumário

Vários são os mecanismos que podem ser utilizados para proteger a informação em redes de computadores. Estes mecanismos baseiam-se, normalmente, em sistemas de criptografia. A criptografia permite tornar a informação ilegível através de uma operação de codificação: a esta operação dá-se o nome de cifra da informação.

Nos sistemas de criptografia de chave simétrica, a operação de cifra dos dados é função de uma determinada chave secreta. Para se voltar a obter o texto original – processo conhecido por decifra – será necessário a mesma chave secreta. Essa chave é do conhecimento das entidades emissora e receptora envolvidas no processo de comunicação.

Nos sistemas de chave pública, o processo de cifra e decifra da informação utiliza duas chaves, a chave privada e a chave pública. A chave privada é apenas do conhecimento da entidade a que pertence; a chave pública é do domínio público, sendo conhecida por qualquer entidade. Nestes sistemas, a informação cifrada com uma das chaves só pode ser decifrada com a outra, existindo uma relação directa entre a chave pública e a chave privada.

Nos sistemas de criptografia de chave simétrica estudaram-se os algoritmos 3DES, RC4 e AES. Estes algoritmos são hoje em dia utilizados em sistemas de segurança das redes sem fios. Nos sistemas de criptografia de chave pública estudou-se o funcionamento do algoritmo RSA e do algoritmo de negociação de chaves DHE. Para cada um dos mecanismos apresentou-se o seu funcionamento, vantagens e desvantagens. Foram também introduzidos os serviços de autenticação, confidencialidade, integridade e não repúdio. Os serviços de autenticação, confidencialidade e integridade baseiam-se na utilização de sistemas criptográficos e das funções de *hash*. O serviço de não repúdio recorre ao uso das assinaturas digitais.

As funções de *hash* utilizam um algoritmo de codificação que permite obter um código de tamanho fixo (código de *hash* ou resumo da mensagem) independentemente do tamanho da mensagem a proteger. Um sistema que utiliza uma função de *hash* para proteger as mensagens, anexa a cada mensagem o seu código de *hash*. A entidade receptora calcula o código de *hash* da mensagem recebida (utilizando a mesma função de *hash* da entidade emissora) e compara-o com o recebido. Se os códigos forem iguais a mensagem não foi alterada. Este processo permitiu desenvolver o conceito de assinatura digital. As funções de *hash* apresentadas foram o MD5 e SHA, apresentando-se também uma comparação entre elas.

Abordou-se também um novo conceito de autenticação através de OTPs, em que estas definem que sempre que uma entidade for autenticada, deverá utilizar uma palavra-chave diferente, dificultando a obtenção e a utilização da mesma por um intruso. Finalmente, abordaram-se as soluções relativas ao problema da distribuição segura de chaves, onde se referiram como soluções as Autoridades Certificadoras e os certificados digitais. Os certificados são um conceito hoje em dia muito utilizado para realizar a distribuição segura das chaves.

Capítulo 4

Protocolos de autenticação em redes sem fios

Um dos principais requisitos de segurança em redes sem fios é a utilização do serviço de autenticação. Várias são as possibilidades de se criarem mecanismos de autenticação: a utilização de servidores de acesso remoto, a utilização de certificados para a autenticação, a utilização de um serviço de directorias.

Sempre que se definam mecanismos de autenticação nas redes sem fios deve-se ter em conta a especificidade destas redes, devendo a escolha dos protocolos de autenticação recair sobre aqueles que realmente garantam alguma consistência e fiabilidade aos mecanismos de autenticação.

Este capítulo faz uma abordagem aos protocolos de autenticação mais divulgados nas redes sem fios. A secção 4.1 introduz o protocolo de autenticação *Extensible Authentication Protocol* (EAP), onde são também apresentadas as variantes mais utilizadas do EAP em redes sem fios, como o EAP-*Transport Layer Security* (EAP-TLS), EAP-*Tunneled Transport Layer Security* (EAP-TTLS) e EAP-*Protected Extensible Authentication Protocol* (EAP-PEAP). A secção 4.2 descreve o protocolo de autenticação *Remote Access Dial In User Service* (RADIUS). Para finalizar a secção 4.3 apresenta um pequeno sumário do capítulo.

4.1. Extensible Authentication Protocol (EAP)

O *Extensible Authentication Protocol* (EAP) [131] é um protocolo geral de autenticação definido pelo *Internet Engineering Task Force* (IETF). Foi desenvolvido originalmente para ser utilizado com o *Point-to-Point Protocol* (PPP) [110]. É um protocolo de autenticação que fornece uma interface geral para diversos mecanismos de autenticação, onde estão incluídos o *kerberos* [132], criptografia de chave pública, *smart-cards* [133] e *One Time Passwords* (OTPs). Esta interface geral permite suportar a interoperabilidade e a compatibilidade entre métodos de autenticação de uma forma simples.

O EAP é utilizado como uma das tecnologias base que permite a autenticação na rede tanto a clientes de redes sem fios como a clientes de redes com fios. Como o protocolo EAP não necessita de um endereço TCP/IP para comunicar (utiliza a camada de ligação de dados), pode transportar mensagens entre dispositivos sem que estes tenham um endereço IP.

O EAP, só por si, não pode ser utilizado como protocolo de autenticação; este é apenas uma norma que define como trocar mensagens de autenticação entre um cliente e um servidor

de autenticação. As capacidades de segurança e a eficiência do processo criptográfico dependem do protocolo de autenticação EAP utilizado. Os protocolos de autenticação suportados pelo EAP incluem:

- EAP-MD5 (*Message Digest* 5) [131];
- EAP-TLS (*Transport Level Security*) [95];
- EAP-TTLS (*Tunneled* TLS) [97];
- EAP-PEAP (*Protected* EAP *Protocol*) [105].

Estes protocolos são descritos na secção 4.1.5.

O EAP permite resolver os problemas de segurança relacionados com ataques de dicionário às palavras-chave, ataques do tipo homem no meio (*Man-In-The-Middle* - MITM) e ataques de roubo e controlo de sessão.

4.1.1. Processo de autenticação EAP

Uma rede com suporte de EAP é composta por três entidades: cliente (*supplicant*), autenticador e servidor de autenticação.

O cliente, ou *supplicant*, é o dispositivo que precisa ser autenticado. O cliente fornece as credenciais de autenticação (um certificado ou um nome de utilizador e uma palavra-chave) ao autenticador e solicita acesso à rede. Como o EAP foi inicialmente desenvolvido para ser utilizado em ligações de rede por ligação telefónica, para ser possível utilizá-lo numa rede de área local e permitir assim ao cliente comunicar com o autenticador, foi definido o processo EAP *over Local Area Network* (EAPoL) – descrito na secção 5.3.2.

O autenticador é o dispositivo que implementa a segurança ao nível da porta de ligação e que controla o acesso à rede. O autenticador recebe as credenciais de autenticação fornecidas pelo cliente e envia-as ao servidor de autenticação. Dependendo do protocolo EAP de autenticação negociado entre o cliente e o servidor de autenticação, o autenticador retransmite as mensagens necessárias entre o cliente e o servidor de forma a facilitar o pedido de autenticação. O autenticador pode operar em dois modos diferentes: pode efectuar as funções de troca de mensagens localmente e comunicar com o servidor utilizando o protocolo RADIUS, ou pode operar como ponte EAP, permitindo que o servidor de autenticação realize as funções necessárias de troca de mensagens EAP.

O servidor de autenticação valida a informação das credenciais do utilizador fornecidas pelo cliente, e determina a aceitação ou não do pedido de acesso à rede. O servidor de autenticação especifica também o protocolo de autenticação EAP a utilizar entre o cliente e ele próprio. Pode também, assim que o pedido de acesso seja aceite, especificar parâmetros

adicionais. Estes parâmetros adicionais poderão ser utilizados para definir políticas de acesso e autorizar o acesso a áreas específicas da rede utilizando *Virtual* LANs (VLANs) dinâmicas.

Antes do cliente ser autenticado, o autenticador coloca a porta da rede no estado *uncontrolled* (sem autorização), permitindo apenas a troca de mensagens EAP/EAPoL de autenticação entre o cliente e o servidor de autenticação (Figura 4.1). Quando um cliente é autenticado e autorizado com sucesso, a porta é colocada no estado *controlled* (com autorização) e é permitido o acesso à rede.

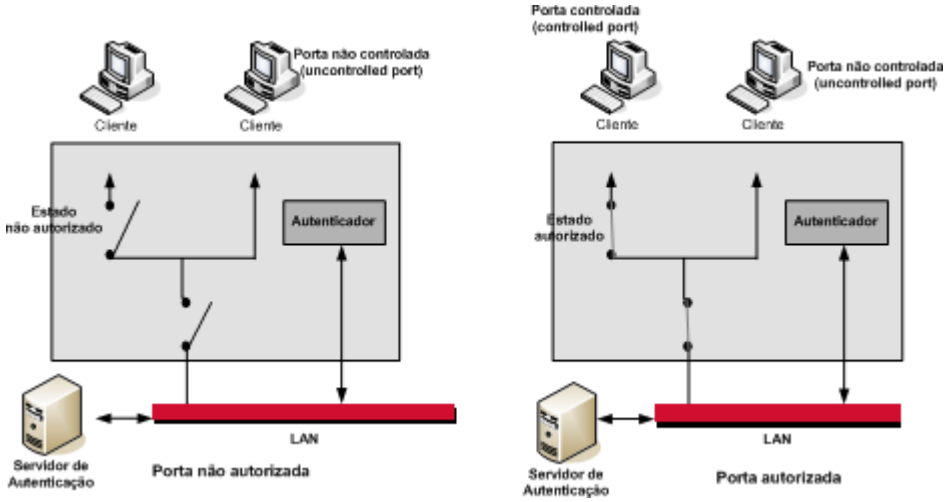


Figura 4.1 - Processo de autenticação EAP.

4.1.2. Pilha protocolar e formato das tramas EAP

A Figura 4.2 ilustra a pilha protocolar EAP. A figura mostra que o protocolo EAP situa-se entre o protocolo de ligação de dados (PPP, 802.3, 802.11) e o método EAP específico (TLS, MD5, etc.).

A Figura 4.3, por outro lado, mostra o formato do pacote EAP que compreende as seguintes partes:

1. *Código*/*Code* (1 *byte*): (1) *request*, (2) *response*, (3) *success*, (4) *failure*
2. *Identificador*/*Identifier* (1 *byte*): ajuda a corresponder pedidos com respostas.
3. *Comprimento*/*Length* (2 *bytes*): indica o comprimento do pacote EAP, incluindo os campos *Code*, *Identifier*, *Length*, *Type* e *Data*.
4. *Tipo*/*Type* (1 *byte*): este campo apenas surge se o pacote EAP for um pacote de pedido ou de resposta. Tipos superiores ou iguais a 4 indicam métodos de autenticação (por exemplo o valor 13 indica o método EAP-TLS); *Tipo* igual a 1 indica identidade, e *Tipo* igual a 2 indica notificação. Normalmente o campo *Tipo* de uma resposta é o mesmo que o de um pedido. Existe, no entanto, um *Tipo* de resposta *Non-Ackowledge* (NAK) (*Tipo*=3) para indicar que um pedido não foi aceite.

5. *Dados*/*Data* (variável): o campo *Dados* varia dependendo do método de autenticação EAP utilizado.

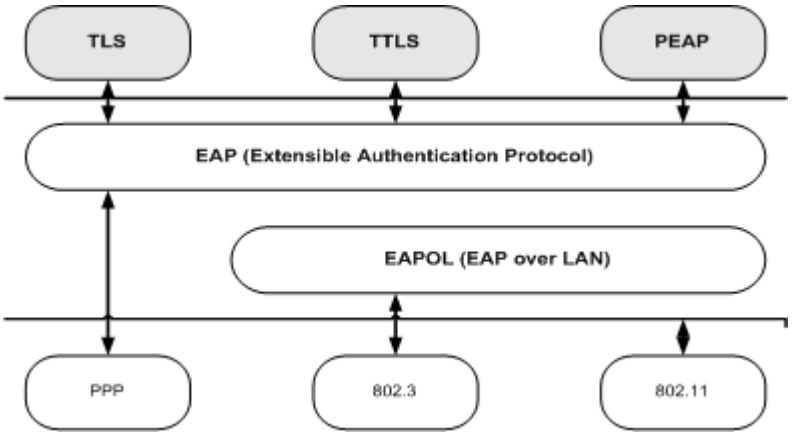


Figura 4.2 - Pilha protocolar EAP.

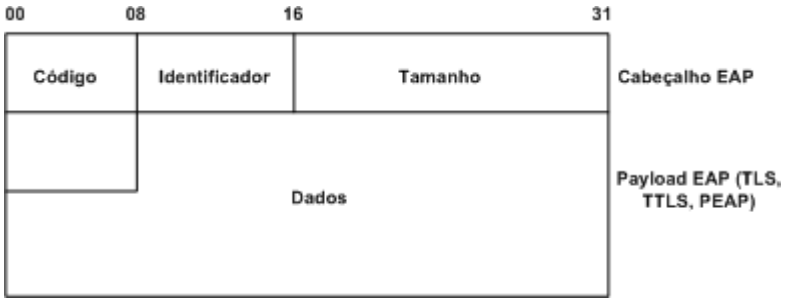


Figura 4.3 - Formato do pacote EAP.

4.1.3. Processo de troca de mensagens EAP

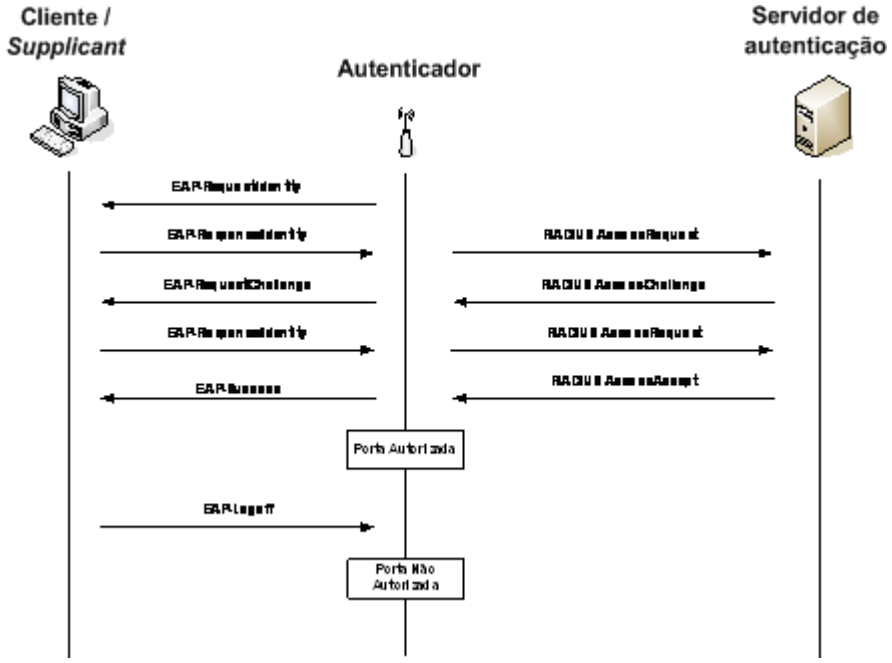


Figura 4.4 - Troca de mensagens EAP.

O EAP define uma estrutura comum para a troca de mensagens entre o cliente e a rede. No entanto, dependendo do protocolo de autenticação acordado entre o cliente e o servidor, alguns dos detalhes da troca de mensagens poderão ter algumas variações.

A Figura 4.4 ilustra o processo geral da comunicação entre o cliente EAP e o servidor de autenticação. Quando o cliente EAP efectua a ligação à rede, tenta aceder à sua informação. Nesta situação, o autenticador envia uma mensagem EAPoL ao cliente, solicitando-lhe a sua identidade. O cliente responde a este pedido enviando ao autenticador a sua identidade numa mensagem EAPoL. O autenticador encaminha a identidade do cliente para o servidor de autenticação, utilizando o protocolo acordado entre o servidor e o cliente. A norma EAP suporta múltiplos servidores de autenticação, embora o mais comum seja o RADIUS. O servidor de autenticação responde com um desafio (*challenge*) ao autenticador, especificando o tipo de autenticação EAP suportado. Esta mensagem é transmitida para o autenticador através do protocolo RADIUS. O autenticador encaminha o desafio com o tipo de autenticação solicitado pelo servidor para o cliente, utilizando EAPoL. O cliente examina o desafio e determina se pode suportar o protocolo de autenticação EAP solicitado: se não suportar o tipo de autenticação, envia uma mensagem de não confirmação (NAK) e tenta negociar um método de autenticação alternativo; se o cliente suportar o método de autenticação solicitado, responde através de uma mensagem EAPoL (mensagem EAP-*Response/Identity*), indicando as suas credenciais. O autenticador comunica as credenciais do cliente ao servidor, através de uma mensagem RADIUS de pedido de acesso. Se as credenciais forem válidas, o servidor de autenticação autentica e autoriza o cliente; caso contrário, o cliente é rejeitado. Se o cliente for autorizado, o servidor envia uma mensagem RADIUS de aceitação do pedido ao autenticador. O autenticador recebe a mensagem e configura o acesso à rede de acordo com cada uma das mensagens. Quando o cliente não pretender aceder a mais informação, ou seja quando pretender desligar-se da rede, informa o autenticador através do envio de uma mensagem de EAP-*Logoff*. Ao receber esta mensagem, o autenticador termina as ligações estabelecidas com o cliente. Para ser possível ao cliente aceder novamente aos recursos da rede, é necessário repetir todo o processo de autenticação.

4.1.4. Vantagens do EAP

As vantagens do EAP advêm do facto deste apenas definir o processo de troca de mensagens entre os participantes numa autenticação – cliente, autenticador e servidor de autenticação. Referem-se a seguir algumas das vantagens do EAP:

- Suporta múltiplos protocolos de autenticação sem a necessidade de programar o autenticador com os mecanismos específicos de autenticação. O EAP permite ao servidor de autenticação controlar que protocolos de autenticação são suportados entre ele e o cliente, sem se ter que configurar de forma total o autenticador com o protocolo de autenticação. O autenticador age como uma ponte entre o cliente e o servidor de autenticação.
- O autenticador pode autenticar clientes locais e, ao mesmo tempo, agir como ponte para clientes remotos que utilizam protocolos de autenticação que ele não suporta.
- A utilização de um autenticador e de um servidor de autenticação permite desenvolver trocas de mensagens e simplifica a gestão de credenciais. O autenticador apenas determina o resultado da autenticação a partir das mensagens fornecidas pelo servidor de autenticação. O resultado da autenticação não é afectado pelo conteúdo dos pacotes EAP – que poderão ser vulneráveis a ataques, manipulações, etc.

4.1.5. Tipos de autenticação EAP

Dos vários tipos de autenticação EAP, apenas as variantes que se baseiam no TLS, versão actual do *Secure Sockets Layer* (SSL), são aplicáveis às redes sem fios (o TLS protege a negociação entre o cliente e o servidor das credenciais dos utilizadores).

Nas secções seguintes descrevem-se os métodos de autenticação EAP-TLS, EAP-TTLS e EAP-PEAP. Não se apresenta uma descrição do método de autenticação EAP-MD5 já que é um método que apresenta várias falhas ao nível da segurança O EAP-MD5 não efectua uma autenticação mútua; utiliza apenas para autenticação dos utilizadores e clientes, a verificação de um código de *hash* das palavras-chave de cada utilizador. Este tipo de autenticação possibilita a um intruso, com as ferramentas adequadas, obter os códigos de *hash* das palavras-chave e as identidades das estações. O EAP-MD5 não protege também contra ataques de controlo de sessão. A Tabela 4-1 apresenta um pequeno resumo das características dos diversos métodos de autenticação EAP utilizados.

	EAP-MD5	EAP-TLS	EAP-TTLS	EAP-PEAP
Autenticação Do servidor	Nenhuma	Chave Pública (certificados)	Chave Pública (certificados)	Chave Pública (certificados)
Autenticação do cliente	*Hash* da Palavra-chave	Chave Pública (certificados)	*Challenge Handshake Access Protocol* (CHAP), *Password Authentication Protocol* (PAP), *Microsoft* CHAP *version* 2 (MS-CHAPv2), EAP	Qualquer tipo de EAP como EAP-MS-CHAPv2 ou chave-pública
Gestão de chaves dinâmicas	Não	Sim	Sim	Sim
Ameaças à segurança	Exposição da identidade, ataques de dicionário, ataque "homem no meio", controlo de sessão	Exposição de identidade	Ataque "homem no meio"	Ataque "homem no meio"

Tabela 4-1 - Características dos métodos de autenticação EAP.

4.1.5.1. EAP-TLS (EAP-*Transport Layer Security*)

O EAP-TLS é um método de autenticação proposto pelo IETF no RFC2716 [95], e é suportado pela maioria dos fabricantes de equipamentos. Como o nome indica, utiliza o protocolo TLS (RFC2246), originalmente criado pela *Netscape,* que é a última versão do protocolo SSL utilizado para proteger tráfego *web*. O EAP-TLS utiliza como base para o processo de autenticação um sistema baseado em criptografia de chave pública e assinaturas digitais. O EAP-TLS baseia-se em certificados X.509 [96] para resolver o processo de autenticação. O cliente deverá possuir um certificado que possa ser validado pelo servidor de autenticação; em contrapartida, o servidor deverá apresentar um certificado ao cliente, que este deverá validar.

O EAP-TLS fornece uma autenticação mútua "forte" entre o cliente e o servidor de autenticação. No entanto, em ambientes de larga escala, pode ter um custo administrativo elevado, devido à geração e à revogação de certificados: cada cliente deverá ter um certificado

digital e um certificado da AC para validar o certificado do servidor; o servidor terá que guardar um certificado digital por cada cliente, o seu próprio certificado e um certificado da AC. A Figura 4.5 ilustra o processo de autenticação EAP-TLS. Note-se que a autenticação EAP-TLS apenas tem lugar após o cliente enviar ao autenticador a mensagem EAP de resposta de Identidade (*EAP-Response Identity*). O servidor de autenticação envia como resposta uma mensagem EAP com a indicação do início da autenticação TLS; esta mensagem não tem campo de dados. Os campos dos dados das próximas mensagens encapsulam mensagens de *handshake* do protocolo TLS. Depois de o autenticador encaminhar esta mensagem para o cliente, e como é a primeira vez que o cliente estabelece uma comunicação com o servidor, o cliente responde com uma mensagem *client hello*. Esta mensagem inclui a versão ou versões do protocolo TLS suportados pelo cliente, um número aleatório que será utilizado para inicializar o processo de *handshake* TLS, e uma lista com conjuntos de primitivas criptográficas. A lista das primitivas criptográficas contém a combinação de algoritmos criptográficos suportados pelo cliente e organizados por ordem de preferência. Cada conjunto de primitivas criptográficas define um algoritmo para troca de chaves, um algoritmo para cifra dos dados e um algoritmo de autenticação de mensagens (assinatura digital). Esta mensagem e as subsequentes, enviadas pelo cliente ao servidor, são encaminhadas pelo autenticador sob a forma de mensagens EAP-*Response*. Durante o processo de autenticação EAP-TLS, o autenticador efectua apenas o encaminhamento e o encapsulamento de mensagens EAP em mensagens RADIUS (EAP-*Response*), e de mensagens RADIUS em mensagens EAP (EAP-*Request*).

O servidor, em resposta à mensagem *client hello*, envia uma mensagem *server hello*. Esta mensagem só é enviada caso o servidor aceite alguma das primitivas criptográficas indicadas pelo cliente. Esta mensagem inclui a versão do protocolo TLS a utilizar, o conjunto de primitivas criptográficas escolhidas, outro número aleatório, e o certificado (ou cadeia de certificados) X.509 do servidor. O certificado fornece ao cliente a identidade e a chave pública do servidor. Como o servidor pretende autenticar o cliente, solicita-lhe o seu certificado (*certificate request*). O servidor informa também o cliente do final da mensagem de *hello* (*server hello done*). Depois de receber esta mensagem, o cliente verifica a validade do certificado do servidor com a chave pública da AC e da lista de revogação de certificados. Se o certificado for válido, o cliente gera um novo valor aleatório. Este valor é conhecido por *pre-master secret*, e será utilizado em conjunto com os outros números aleatórios para calcular a chave de sessão. Ao contrário dos outros números aleatórios que foram enviados sem serem cifrados, o *pre-master secret* é enviado de forma cifrada ao servidor. O cliente

envia então ao servidor o seu certificado e o *pre-master secret* cifrado com a chave pública do servidor (*client key exchange*). Como o cliente envia um certificado, deve fornecer garantias ao servidor que é o proprietário legítimo do mesmo. Para tal, gera um código de *hash* de todas as suas mensagens enviadas e recebidas até ao momento e cifra-as com a sua chave privada (*certificate verify*). O cliente também informa o servidor que está pronto a utilizar as novas chaves e primitivas criptográficas negociadas (*change cipher spec*). Para finalizar, o cliente indica que a autenticação e a troca de chaves concluíram com sucesso (*finished*). Nesta fase, o cliente determina a chave de sessão através de uma função geradora de pseudoaleatórios que combina o número aleatório gerado pelo cliente, o número aleatório gerado pelo servidor e o *pre-master secret*. Esta chave não é enviada para o servidor.

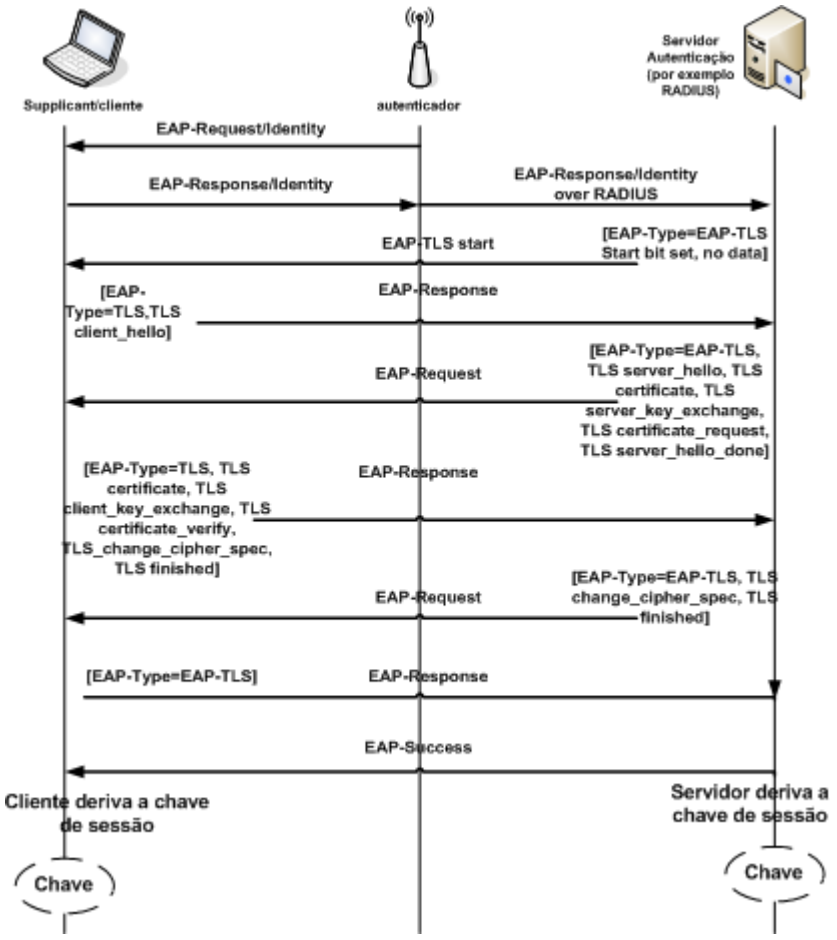


Figura 4.5 - Autenticação EAP-TLS.

O servidor, ao receber a mensagem do cliente, utiliza a chave pública deste para obter o valor de *certificate verify*. O servidor calcula o valor do código de *hash* das mensagens enviadas e recebidas do cliente, e compara-o com o valor da mensagem *certificate verify.* Se forem iguais, o cliente é autenticado. O servidor utiliza então a sua chave privada e decifra o valor da mensagem *client key exchange*, obtendo o *pre-master secret*. Da mesma forma que o cliente, o servidor determina a chave de sessão. O servidor informa então o cliente que vai

utilizar as chaves e as primitivas criptográficas para proteger os dados (*change cipher spec*), finalizando com a informação de que a autenticação e a negociação de chaves foram bem sucedidas. Na mensagem *finished*, o servidor inclui um código de *hash* de todas as mensagens trocadas e do *pre-master secret*. Esta mensagem é enviada cifrada com as primitivas criptográficas negociadas.

O cliente decifra esta mensagem, calcula o código de *hash* das mensagens trocadas e do *pre-master secret*, e compara-as com o valor da mensagem decifrada. Se forem iguais, o servidor é autenticado. Se o servidor não conhecer a chave privada correspondente com a chave pública do certificado, não pode obter o *pre-master secret*, e também não pode obter o código *hash*. Como o processo conclui com sucesso, e como o cliente não pretende obter mais informação do servidor, então envia uma mensagem vazia.

Para que todo o processo finalize, o servidor envia uma mensagem de autenticação EAP bem sucedida (*Success)*. A partir deste instante, todas as comunicações serão protegidas pelas chaves e primitivas criptográficas negociadas. Nesta última mensagem, o servidor indica ao autenticador qual a chave que permitirá cifrar os dados trocados com o cliente, possibilitando assim ao cliente e ao autenticador comunicarem de forma segura.

4.1.5.2. EAP-TTLS (EAP-*Tunneled* TLS)

O EAP-TTLS [97] é uma extensão do EAP-TLS proposto pela *Funk* [33] e pela *Certicom* [144]. Fornece as vantagens de um sistema criptográfico "forte" sem a complexidade do requisito de certificados digitais no cliente e no servidor de autenticação. Tal como o TLS, o EAP-TTLS suporta autenticação mútua, mas apenas requer que o servidor seja validado pelo cliente através de uma troca de certificados. O EAP-TTLS permite que o cliente se autentique utilizando nomes de utilizador e palavras-chave, sendo apenas necessário um certificado para o servidor de autenticação. Este mecanismo simplifica a gestão e a manutenção do sistema, mantendo a segurança e autenticação.

Um túnel TLS pode ser utilizado apenas para proteger as mensagens EAP, sendo a autenticação baseada em serviços já disponíveis na rede, tais como o RADIUS [98], *Lightweight Directory Access Protocol* (LDAP) [99] e *Active Directory* [100].

A negociação EAP-TTLS compreende duas fases: a fase da identificação e a fase do túnel TLS. Na primeira fase é utilizado o TLS para o cliente autenticar o servidor (através de um certificado digital); opcionalmente, o servidor pode pedir a autenticação do cliente. Esta fase resulta no acordo de um conjunto de primitivas criptográficas, permitindo que a fase seguinte proceda de forma segura (utilizando a camada de registo TLS). Na segunda fase é utilizada a camada de registo TLS para proteger a comunicação, criando um túnel entre o

cliente e o servidor. Nesta fase efectua-se a autenticação do cliente e a distribuição de chaves. A comunicação entre o cliente e o servidor de autenticação realiza-se através da troca de pares atributo-valor, compatível com vários protocolos, tal como o RADIUS.

A Figura 4.6 exemplifica a troca de mensagens EAP-TTLS. A primeira fase tem início quando o autenticador envia ao cliente/*supplicant* a mensagem *EAP-Request Identity*, seguida pela resposta do cliente *EAP-Response Identity*. Esta mensagem não inclui a identidade do cliente, evitando que um *eavesdropper* obtenha a identidade (esta é apenas enviada após o estabelecimento do túnel TLS). De seguida é efectuada a autenticação do servidor, estabelecendo-se um túnel de protecção da informação; qualquer informação a circular no túnel é cifrada. Ao processo de autenticação do servidor dá-se o nome de *TLS handshake*. Neste processo o servidor envia um desafio ao cliente (mensagem *Access-Challenge* EAP-TTLS), indicando-lhe que terá início o método de autenticação EAP-TTLS e que deverá ser utilizada a negociação TLS para se obterem as chaves de sessão; o pedido com a autenticação EAP-TTLS é então enviado pelo autenticador ao cliente através da mensagem *EAP-Request*. Se o cliente concordar com o método de autenticação solicitado pelo servidor, responde com a mensagem *EAP-Response* EAP-TTLS, com o parâmetro *TLS client hello*. Assim que o servidor receber a confirmação do cliente, através da mensagem enviada pelo autenticador (*Access-request* EAP-TTLS), dá-se início à negociação TLS para se obterem as chaves de sessão. Este processo é muito semelhante ao processo descrito na secção 4.1.5.1, com a particularidade de apenas ser utilizado o certificado do servidor para autenticação e obtenção das chaves de sessão. As chaves de sessão são depois utilizadas para criar o túnel de protecção da informação. Esta fase termina com o envio da mensagem *EAP-Request* EAP-TTLS, do servidor ao cliente, onde o servidor indica ao cliente o estabelecimento do túnel para proteger a informação (TLS: *finished*).

De seguida, inicia-se a segunda fase de autenticação do cliente. Nesta fase é utilizado um outro método de autenticação através do túnel (pode-se utilizar como métodos de autenticação o PAP [101], CHAP [102], MS-CHAP [103], MS-CHAPv2 [104] ou outro método EAP como o EAP-MD5). Para ser autenticado, o cliente indica a sua verdadeira identidade ao servidor (mensagem EAP- *Response Identity* (*real*)). Depois de verificada a identidade, através do envio da identidade e de uma palavra-chave pelo cliente, o servidor informa o cliente que será utilizado um outro método para autenticação. Na Figura 4.6 é utilizado o método de autenticação EAP-MD5 (mensagem *EAP- Request MD5*). Este método utiliza um código de *hash* MD5 para verificar as credenciais do cliente. Caso o código de *hash* seja validado com sucesso pelo servidor, este envia uma mensagem *EAP-Success* ao

cliente indicando-lhe que a autenticação foi bem sucedida, e enviando as novas especificações criptográficas para protecção dos dados durante a sessão.

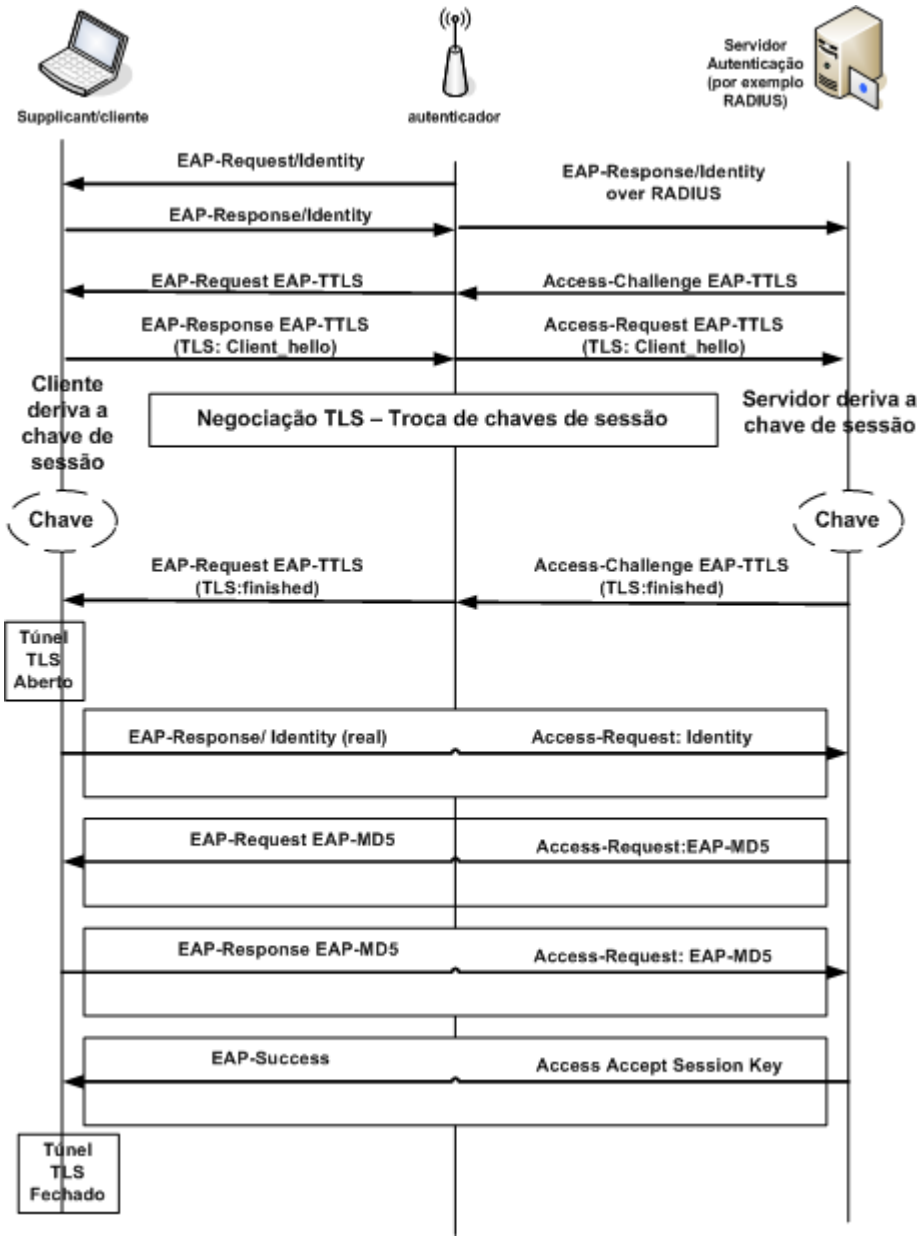


Figura 4.6 - Autenticação EAP-TTLS.

4.1.5.3. PEAP (*Protected* EAP)

O protocolo PEAP [105] é, em termos da funcionalidade de autenticação mútua, muito semelhante ao EAP-TTLS. O PEAP foi proposto pela *Cisco*, *Microsoft* e RSA *Security* como alternativa ao EAP-TTLS. O PEAP tenta contornar as fraquezas do EAP das seguintes formas: protegendo as credenciais dos utilizadores e a negociação EAP, normalizando o processo de troca de chaves, suportando fragmentação e reassemblagem e re-ligações rápidas.

O PEAP permite a utilização de outros protocolos de autenticação EAP, e protege as comunicações com um túnel TLS cifrado; a troca e criação de chaves é efectuada através do

TLS. O cliente PEAP autentica-se directamente com o servidor de autenticação. Neste caso, o autenticador funciona como uma ponte, não sendo necessário que este compreenda os protocolos específicos de autenticação EAP. Ao contrário do EAP-TTLS, o PEAP não suporta de forma nativa a autenticação baseada em serviços centralizados como o LDAP.

Tal como o EAP-TTLS, a estrutura do PEAP assenta em duas fases: numa primeira fase estabelece-se a segurança, e na segunda fase efectua-se a autenticação de forma segura.

A principal diferença entre o EAP-TTLS e o PEAP reside nos métodos de autenticação do cliente pela rede suportados no túnel TLS. O PEAP apenas suporta outros métodos EAP (EAP-MD5, EAP-TLS, etc.), enquanto que o TTLS permite muitos outros métodos (como referido anteriormente).

4.1.5.4. Comparação dos métodos de autenticação EAP

Método	Descrição	Atributos De Autenticação	Segurança *wireless*	Dificuldade de Implementação
TLS	Autenticação baseada em certificados nos dois sentidos	Autenticação mútua	A mais forte	Difícil
TTLS	Servidor autenticado através de certificado, cliente autenticado através de outro método	Autenticação mútua	Forte	Moderada
PEAP	Servidor autenticado através de certificado, cliente autenticado através de outro método EAP	Autenticação mútua	Forte	Moderada

Tabela 4-2 - Comparação dos métodos de autenticação EAP.

Na Tabela 4-2 apresenta-se uma comparação entre os métodos de autenticação EAP mais utilizados em redes sem fios. O método de autenticação mais aconselhado para utilização numa rede sem fios é o EAP-TLS. Este método permite autenticação mútua do cliente e do servidor baseada em certificados digitais, garantindo assim protecção contra ataques do tipo homem no meio, personificação e de controlo de sessão. Este método não protege contra ataques de dicionário. O principal inconveniente deste método é o de ser mais difícil de implementar comparativamente com os outros métodos, já que é necessário criar um sistema de chave pública e de assinaturas digitais. Os métodos de autenticação EAP-PEAP e EAP-TTLS são mais fáceis de implementar, sendo apenas necessário um certificado para o servidor (e não para autenticar o cliente). A autenticação do cliente baseia-se em métodos como o PAP, CHAP, MS-CHAPv2, etc. Estes métodos garantem protecção contra ataques de

dicionário e de controlo de sessão, mas não garantem protecção contra ataques do tipo "homem no meio".

4.2. RADIUS – Remote Access Dial-In User Service

O RADIUS [98] é um protocolo que permite a um servidor de acesso remoto – AS (*Access Server*) – fazer a autenticação de um utilizador através de um serviço de autenticação. Um cliente RADIUS é um tipo de servidor NAS (*Network Access Server*) – que envia pedidos de autenticação e de registo ao servidor RADIUS (servidor de autenticação) de forma a obter o acesso à rede. No caso específico das redes sem fios o NAS é representado pelo ponto de acesso (AP).

O RADIUS encontra-se especificado pelo IETF e destina-se a ser utilizado em redes TCP/IP – assume que os dispositivos utilizam uma rede IP para comunicar com o servidor RADIUS. Este protocolo define um conjunto de funcionalidades que deverão ser comuns aos servidores de autenticação, e define um protocolo de sinalização que permite aos outros dispositivos aceder a essas funcionalidades. Um servidor RADIUS consiste na parte do servidor de autenticação que suporta as capacidades RADIUS; o RADIUS consiste no protocolo utilizado para comunicar com o servidor.

Em [108] é apresentada uma especificação de funcionamento de EAP sobre RADIUS. As secções seguintes descrevem muito brevemente o protocolo RADIUS, suas mensagens e conteúdos.

4.2.1. Troca de mensagens RADIUS

O RADIUS define um conjunto de mensagens a serem trocadas entre o NAS e o AS. As mensagens definidas são muito simples e representam pedidos e desafios, bem como a aceitação ou não desses pedidos. A Figura 4.7 indica uma troca típica de mensagens RADIUS. O servidor RADIUS envia mensagens de desafio (*Access-Challenge*) que dependem do método de autenticação utilizado (PAP, CHAP, MsCHAPv2, TLS, PEAP, TTLS, etc.). As mensagens de desafio servem para que o servidor solicite ao cliente mais informação de autenticação, como certificados ou outro tipo de informação, reduzindo assim o risco de autenticação fraudulenta. O cliente RADIUS envia pedidos de acesso (*Access-Request*), de forma a enviar a sua identidade ou a responder aos desafios do servidor. Eventualmente, o servidor RADIUS autentica ou não o cliente (*Access-Accept/Access-Reject*).

A mensagem *Access-Request* é enviada do cliente para o AS, sempre que há uma tentativa de ligação de algum utilizador. O servidor RADIUS pode enviar uma mensagem *Access-Accept* de aceitação do utilizador (e enviando também atributos que indicam como

deverá ser configurado o acesso à rede). Caso o pedido de acesso à rede seja negado, o servidor envia a mensagem *Access-Reject*. Caso o servidor queira assegurar-se de que o utilizador é realmente quem diz que é, pode enviar-lhe uma mensagem de desafio *Access-Challenge*, para verificar se o utilizador lhe envia a resposta correcta. Caso o pedido de acesso à rede seja concedido, o servidor envia a mensagem *Access-Accept*.

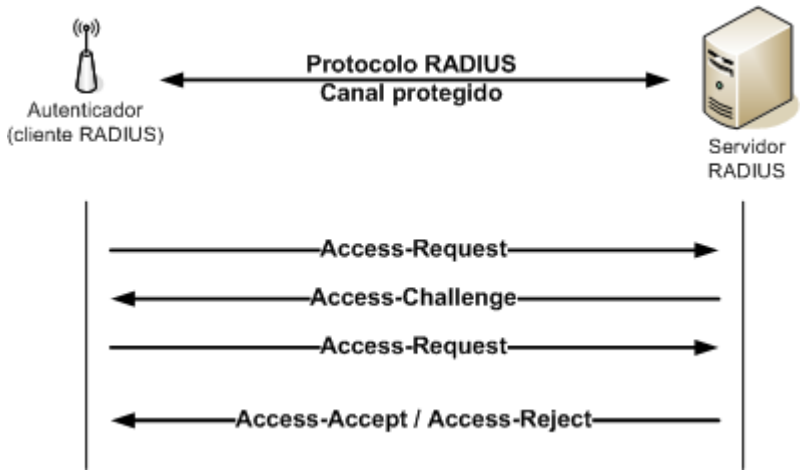


Figura 4.7 - Troca de mensagens RADIUS.

4.2.2. Formato e atributos das mensagens RADIUS

O protocolo de autenticação RADIUS apenas especifica quatro mensagens. No entanto, o significado das mensagens pode ser alterado através da utilização de atributos diferentes nessas mensagens. Dependendo do método de autenticação utilizado, também os atributos nas mensagens são diferentes. O corpo principal das mensagens RADIUS é composto por um conjunto de atributos, cada um deles contendo um conjunto de informação com significado para todas as partes na comunicação (como por exemplo, a identificação do utilizador, a palavra-chave de acesso à rede, etc.).

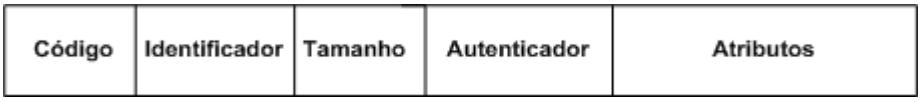


Figura 4.8 - Formato da mensagem RADIUS.

A Figura 4.8 representa o formato básico de uma mensagem RADIUS. O *byte* de *Código* indica o tipo da mensagem RADIUS, e o *Identificador* é um número arbitrário utilizado para fazer a correspondência entre pedidos e respostas. O campo *Autenticador* é destinado a reforçar a segurança das comunicações; a forma como é utilizado depende do tipo de mensagem. Numa mensagem *Access-Request*, o *Autenticador* é preenchido por um *nonce* (valor aleatório), em que este será utilizado para a cifra da palavra-chave (esta é cifrada utilizando uma combinação entre a chave secreta e o *nonce*). Nas mensagens de resposta, o *nonce* é utilizado para se obter um valor de verificação de integridade das mensagens. No

campo *Tamanho* é indicado o número total de *bytes* da mensagem. Os atributos das mensagens RADIUS contêm a informação realmente utilizada para estabelecer a autenticação. Cada mensagem pode conter um ou mais atributos, e cada um deles representa um pacote de informação. A capacidade de definir e suportar novos atributos permite extender as capacidades do RADIUS. Cada atributo das mensagens RADIUS contém um campo de *Tipo* para identificar o atributo, um campo que indica o tamanho, e um nome que define o atributo. A tabela seguinte indica alguns exemplos de atributos RADIUS.

Valor do Tipo de atributo	Nome	Descrição
1	*User-Name*	A identificação ou o nome de utilizador.
2	*User-Password*	Contém a palavra-chave que permite o acesso à rede. A palavra-chave é cifrada utilizando um segredo partilhado e o valor do *nonce* obtido do campo Autenticador da mensagem *Access-Request*.
4	*NAS-IP-Address*	O endereço IP do NAS ao qual o servidor RADIUS deverá responder.
18	*Reply Message*	Utilizado para enviar texto ao utilizador, indicando a necessidade de alguma acção ou de algum evento.

Tabela 4-3 - Exemplo de atributos RADIUS.

4.3. Sumário

O acesso a uma rede sem fios deve ser efectuado após autenticação bem sucedida. Para ser possível implementar o serviço de autenticação é necessário um conjunto de protocolos que estabeleçam esse serviço – os protocolos de autenticação. O protocolo de autenticação mais utilizado nas redes sem fios é o EAP. O EAP, embora inicialmente desenvolvido para utilização com o protocolo PPP, é suficientemente versátil e interoperável com muitos outros mecanismos, o que permite a sua utilização em redes sem fios. Na realidade, o EAP não pode ser considerado um protocolo de autenticação, mas sim uma norma de como trocar mensagens de autenticação. Vários são os protocolos de autenticação suportados pelo EAP: EAP-MD5, EAP-TLS, EAP-TTLS e EAP-PEAP. O método de autenticação EAP-MD5 não apresenta um nível de segurança muito elevado, não devendo por isso ser utilizado nas redes sem fios. Todos os outros métodos baseiam-se em sistemas de criptografia de chave pública e em assinaturas digitais.

O EAP deve ser complementado por um sistema que implemente os serviços de autenticação, autorização e registo. O serviço que normalmente permite ao EAP fornecer

esses mecanismos é o RADIUS. O RADIUS permite definir de forma centralizada quais os utilizadores que têm acesso à rede (autenticação), e que tipo de acesso lhes é fornecido (autorização), podendo-se assim definir políticas de acesso à rede.

Capítulo 5

Protocolos de segurança para redes sem fios

A necessidade de proteger a transmissão de informação numa rede sem fios é uma das consequências da utilização de um meio partilhado como o ar para a transmissão de informação. Hoje em dia existe um conjunto de protocolos de segurança em redes sem fios, com o objectivo de protecção da informação e dos intervenientes da comunicação neste tipo de redes. Estes protocolos devem garantir os serviços de autenticação, confidencialidade, integridade e não-repúdio dos dados.

A primeira tentativa de implementação de um protocolo de segurança em redes 802.11 deu origem ao protocolo *Wired Equivalent Privacy* (WEP) [116]. Este protocolo é, no entanto, pouco eficaz em garantir a segurança em redes sem fios. A implementação de redes sem fios em larga escala no mundo empresarial serviu de força motriz para a definição e desenvolvimento de novos protocolos de segurança em redes 802.11, como o *Wi-Fi Protected Access* (WPA) [46] e o 802.11i (WPA2) [117]. Além do desenvolvimento de novos protocolos, também tem existido algum esforço na adaptação de protocolos de segurança para redes sem fios, como o *Internet Protocol Security* (IPsec) [118]. Enquanto que os protocolos especificados para a protecção de redes sem fios protegem a informação ao nível da camada de ligação de dados, o IPsec protege a informação ao nível da camada de rede. Esta característica permite-lhe ser um protocolo versátil, podendo ser utilizado em qualquer tipo de rede TCP/IP, e permitindo-lhe proteger todo o tipo de tráfego independentemente da aplicação que o gere.

O WPA é um protocolo baseado no 802.11i. O WPA surge pela a necessidade de, num curto espaço de tempo, se introduzirem mecanismos de protecção realmente robustos nas redes sem fios. O WPA e o 802.11i apresentam muitas características em comum. Ambos definem um protocolo de protecção da transferência dos dados que utiliza chaves de cifra diferentes por pacote. Este protocolo utiliza o algoritmo de cifra *Rivest Code* 4 (RC4) no WPA, e *Advanced Encryption Standard* (AES) no 802.11i. Como o algoritmo AES é mais forte, este não exige a utilização de chaves de cifra por pacote. Ambos definem um mecanismo que garante de forma eficaz a integridade dos dados, e utilizam o processo 802.1X para garantir a autenticação dos clientes da rede sem fios. A principal diferença entre ambos é que o WPA é compatível com equipamentos WEP, sendo apenas necessária uma actualização de *firmware* desses equipamentos para ser possível a sua utilização com WPA. Para ser

possível usufruir de todas as potencialidades do 802.11i, nomeadamente a utilização do protocolo AES para a cifra dos dados, é necessário a substituição de todos os equipamentos (APs e adaptadores de redes sem fios).

Este capítulo apresenta na secção 5.1 o protocolo WEP, reflectindo algumas das fraquezas deste protocolo para protecção de redes. As secções seguintes fazem uma abordagem aos protocolos de segurança que têm sido desenvolvidos de forma a eliminar os problemas do WEP, descrevendo-se na secção 5.2 o protocolo IPsec aplicado às redes sem fios, na secção 5.3 o protocolo 802.1X na sua variante *Transport Layer Security* (TLS), e na secção 5.4 o protocolo WPA. Este capítulo descreve também o protocolo desenvolvido pela norma IEEE802.11i na secção 5.5. O capítulo termina na secção 5.6 com um pequeno sumário dos assuntos desenvolvidos no capítulo, bem como uma reflexão crítica dos vários protocolos.

5.1. Wired Equivalent Privacy (WEP)

O WEP [116] é um protocolo de segurança da camada de ligação de dados. O WEP baseia-se na cifra de fluxo RC4, em que a mesma chave é utilizada para a cifra e a decifra dos dados. O termo "*wired equivalent*" significa que o WEP pretende fornecer um nível de segurança semelhante ao de uma rede com fios (existente em *Local Area Networks* - LANs).

O WEP foi desenvolvido com a intenção de reforçar os seguintes serviços de segurança:

- Confidencialidade – prevenir a visualização não autorizada de informação crítica através da utilização de criptografia;
- Controlo de acesso – rejeitar pacotes incorrectamente cifrados;
- Integridade dos dados – prevenir a alteração das comunicações.

5.1.1. Funcionamento do WEP

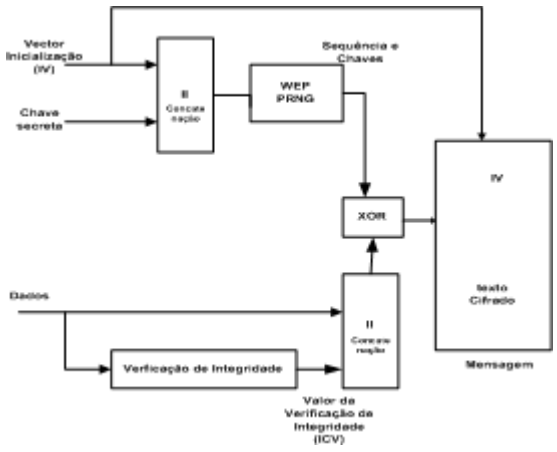


Figura 5.1 - Sistema WEP.

O WEP utiliza uma chave secreta de 40 ou 104[3] *bits*, e um vector de inicialização (*Initialization Vector* – IV) de 24 *bits*. Este vector de inicialização é adicionado à chave secreta em cada pacote de forma a garantir que cada um tem uma chave RC4 diferente (a chave secreta não é frequentemente alterada). Estes procedimentos estão ilustrados na Figura 5.1. A chave secreta é colocada como entrada de um gerador de números pseudoaleatórios (*PseudoRandom Number Generator* – PRNG), dando origem a um fluxo de chaves pseudoaleatórias com o mesmo comprimento da chave inicial. Os dados são processados por um algoritmo de verificação da integridade, resultando num *checksum*. O cálculo do *checksum* é baseado num mecanismo linear *Cyclic Redundancy Check* (CRC). O *checksum* é então concatenado com os dados para que seja possível verificar a sua integridade quando estes forem descodificados. Nos sistemas WEP o *checksum* é designado por *Integrity Check Value* (ICV). A informação a ser transmitida, ou seja o conjunto formado pelos dados com o *checksum*, é cifrada utilizando operações XOR *bit* a *bit* com o fluxo de chaves obtido anteriormente. O vector de inicialização é anexado à informação cifrada e o resultado é transmitido através da rede sem fios.

A norma 802.11 não especifica nenhuma forma de gestão de chaves. Deste modo, a gestão das chaves é manual.

5.1.2. Fraquezas do WEP

O WEP tem sido, desde a sua especificação, alvo de muitas críticas e de ataques que o tornam extremamente vulnerável. As principais falhas do WEP podem ser resumidas em três categorias [31]: não protege contra falsificações, contra ataques de repetição, e ataques de colisão ou de reutilização do vector de inicialização.

Falsificações

O WEP não garante protecção contra falsificações. Mesmo sem o conhecimento da chave de cifra, um atacante pode modificar de forma arbitrária e não detectável pacotes 802.11, pode disponibilizar dados a entidades não autorizadas, e fazer-se passar por um utilizador autorizado. Esta fraqueza está relacionada com o processo de obtenção do *checksum*. Este é obtido através de um algoritmo linear, sendo assim possível inserir ou alterar a informação dos pacotes e obter de forma muito simples e linear o novo *checksum*.

Ataques de repetição

O WEP não garante protecção contra ataques de repetição. Um atacante pode criar falsificações de pacotes sem alterar dados nesses pacotes, sendo apenas necessário registar e

[3] A norma original 802.11 estipula uma chave WEP de 40 *bits*. No entanto, diversos fabricantes disponibilizam implementações de chaves WEP de 104 *bits*, tornando o sistema criptograficamente mais forte.

guardar pacotes WEP para depois os retransmitir. Estes ataques podem também ser utilizados para obter informação relativa à chave de cifra e aos dados que protege. Esta fraqueza está relacionada com o valor de verificação de integridade RC4 (ICV). Quando os dados são cifrados, o valor do *checksum* é anexado no final da trama. Quando os dados são descodificados, o *cheksum* é utilizado para validar os dados. No entanto, como o ICV não é cifrado, é possível alterar o campo dos dados e derivar um novo ICV.

Colisões ou de reutilização de IV

Uma das falhas de segurança mais graves do WEP é a possibilidade de se efectuar o denominado ataque de colisão ao IV. Uma colisão de IV significa a reutilização de um IV num determinado momento da transmissão. Como anteriormente mencionado, um vector de inicialização é adicionado à chave secreta em cada pacote de forma a garantir que cada pacote tenha uma chave RC4 diferente. Um problema reconhecido às cifras de fluxo é que, se dois pacotes forem cifrados com o mesmo IV, facilmente se descodificam os dados cifrados. Descreve-se a seguir e de forma algébrica como se efectua este ataque (v é o IV e k é a chave secreta):

$$C_1 = P_1 \oplus RC4(v,k) \qquad \text{(eq. 5.1)}$$

$$C_2 = P_2 \oplus RC4(v,k) \qquad \text{(eq. 5.2)}$$

$$C_1 \oplus C_2 = (P_1 \oplus RC4(v,k)) \oplus (P_2 \oplus RC4(v,k)) \qquad \text{(eq. 5.3)}$$

$$C_1 \oplus C_2 = P_1 \oplus P_2 \qquad \text{(eq. 5.4)}$$

A primeira e segunda equação mostram que dois textos cifrados (C_1 e C_2) são obtidos através de uma operação XOR com o texto plano (P_1 e P_2) e o mesmo fluxo de chaves RC4(k,v). Demonstra-se mediante manipulação algébrica (eq. 5.3 e eq. 5.4) que, realizar a operação XOR *bit* a *bit* sobre dois textos cifrados que utilizam o mesmo fluxo de chaves (ou seja RC4(v,k)), permite cancelar o fluxo de chaves e tem como resultado os textos concatenados pela operação XOR ($P_1 \oplus P_2$). Deste modo, um atacante pode obter informação sobre os textos P_1 e P_2. Os ataques de colisões são possíveis porque o espaço de 24 *bits* utilizado pelo IV não é suficiente para garantir protecção contra ataques de colisão num período de tempo razoável. Um ponto de acesso que envia 1500 pacotes a 11 *Mbps*, ocupará todo o espaço de IV em menos de cinco horas. Outro factor que possibilita ataques de colisões IV é o facto de algumas placas de rede reinicializarem a zero o IV sempre que a placa é reiniciada, incrementando depois cada IV de uma unidade para cada pacote enviado a seguir. Isto significa que a transmissão começa com um IV conhecido e repetido, resultando numa excelente oportunidade para os ataques de colisão.

A segurança do WEP baseia-se também no pressuposto de que as chaves secretas são frequentemente alteradas. No entanto, na realidade não o são, já que este processo é manual. Sendo assim, verifica-se que os ataques de colisão ocorrem frequentemente.

Todos estes problemas colocam em evidência a falta de capacidade e de maturidade ao nível da segurança do WEP. Como a segurança é, sem dúvida, um dos requisitos mais importantes, devem ser implementados novos mecanismos de segurança que realmente assegurem a protecção das redes sem fios.

5.2. IPsec aplicado às redes sem fios

O IPsec [118] é uma norma para a segurança na camada de rede. Na sua essência o IPsec é um conjunto de protocolos desenvolvidos pelo *Internet Engineering Task Force* (IETF) para a implementação de redes privadas virtuais (*Virtual Private Networks* - VPNs) na *Internet*/*Intranet*.

As VPNs IPsec são um método muito comum para protecção do tráfego que circula através de redes públicas (ou não protegidas). O IPsec fornece a segurança dos dados através de um conjunto flexível de mecanismos de tunelagem e de cifra. O IPsec é uma combinação de diversas tecnologias de segurança que fornecem um sistema completo para ajudar a garantir a confidencialidade, integridade e autenticidade das comunicações dos dados através das redes públicas; fornece os componentes necessários para desenvolver políticas de segurança flexíveis.

O IPsec implementa cifra e autenticação na camada de rede, embebendo a segurança fim-a-fim na arquitectura da rede. A principal vantagem é que as aplicações individuais não necessitam de ser alteradas para obterem forte segurança. Todos os pacotes encaminhados na rede são automaticamente protegidos.

O IPsec suporta dois modos de cifra: transporte e túnel. No modo de transporte, apenas é cifrada a componente dos dados de um pacote, não se alterando o cabeçalho do mesmo. O modo túnel cifra todo o pacote, sendo mais seguro.

Para o funcionamento do IPsec, tanto o receptor como o emissor devem ter acesso às chaves públicas de cada um. A Autoridade Certificadora (AC) permite, ao receptor de uma mensagem cifrada, obter a chave pública do emissor, e autenticá-lo utilizando um certificado digital emitido pela AC. Para comunicar de forma segura, o emissor e o receptor têm inicialmente de estabelecer um acordo, uma associação de segurança (*Security Association* - SA), para determinar a forma como os pacotes trocados por eles deverão ser protegidos.

A norma IPsec inclui dois protocolos que adicionam segurança aos pacotes IP: o *Authentication Header* (AH) [119] garante a integridade dos dados e o *Encapsulating Security Payload* (ESP) [120] garante a confidencialidade. Os diversos parâmetros do IPsec são negociados entre os dispositivos através do protocolo *Internet Key Exchange* (IKE) [121], inicialmente denominado por *Internet Security Association Key Management Protocol* (ISAKMP/Oakley) [122]. O IKE utiliza, para a autenticação dos dispositivos, certificados digitais fornecidos por diversos vendedores. Tanto o ESP como o AH utilizam técnicas criptográficas para garantir a confidencialidade dos dados, e assinaturas digitais para autenticarem a fonte dos dados.

As secções seguintes descrevem os elementos e protocolos constituintes do IPsec: SAs, AH, ESP e ISAKMP.

5.2.1. Associação de segurança (*Security Association* - SA)

As SAs são a base para a implementação do IPsec. Uma SA representa o contrato entre duas entidades comunicantes que determina os protocolos IPsec (ESP, AH) utilizados para proteger os pacotes, as chaves, transformações e duração da validade das chaves. Uma implementação IPsec constrói uma base de dados de SAs (*Security Association DataBase*-SADB) que mantém as SAs utilizadas pelos protocolos IPsec. As SAs são de sentido único, ou seja, se duas entidades A e B comunicarem de forma segura utilizando, por exemplo o ESP, então a entidade A terá uma SA para processar os pacotes que envia (SA_{out}), e terá uma SA diferente para processar os pacotes que recebe (SA_{in}). Da mesma forma a entidade B criará duas SAs para processar os pacotes. A SA_{out} (e a SA_{in}) da entidade A partilha os mesmos parâmetros criptográficos que a SA_{in} (e a SA_{out}) da entidade B.

As SAs são específicas do protocolo utilizado: se duas entidades utilizam em simultâneo o AH e o ESP para comunicarem de forma segura, cada entidade deve criar separadamente SAs para cada protocolo.

Um dos elementos mais importantes da SA é o *Security Parameter Index* (SPI), de 32 *bits*. Um SPI é utilizado para identificar univocamente uma SA na entidade receptora. O emissor utiliza selectores para identificar o SA a ser utilizado. Uma implementação IPsec deve ter obrigatoriamente os seguintes selectores: endereço IP de origem e destino, o nome do protocolo da camada de transporte e os portos de origem e destino. A identificação do SA utilizado no destinatário é mais complexa, pois alguns dos campos dos selectores pertencem à camada de transporte, não tendo o destinatário acesso a esses campos. Para se resolver este problema, o SPI é enviado em cada pacote. O destinatário utiliza o valor do SPI para obter, da SADB, o SA correspondente.

As tarefas mais importantes da gestão de SAs são a criação e a eliminação de SAs. A gestão de SAs pode ser realizada de forma manual ou utilizando um protocolo de gestão de chaves como o IKE.

5.2.2. *Encapsulating Security Payload* (ESP)

O ESP é um protocolo que permite cifrar os dados de um pacote IP, fornecendo os serviços de confidencialidade, autenticação da origem dos dados, anti-repetição dos pacotes e integridade dos dados. O ESP está definido no RFC2406 [120]. Os algoritmos de cifra obrigatórios na implementação ESP são o *Data Encryption Standard* (DES) e o 3DES. O ESP pode ser utilizado em dois modos: o modo túnel e o modo de transporte. Para proteger os dados de um pacote IP, o ESP encapsula os dados na sua estrutura protocolar.

Quando o ESP é utilizado para proteger um pacote, o cabeçalho ESP é introduzido imediatamente a seguir ao cabeçalho IP (Figura 5.2).

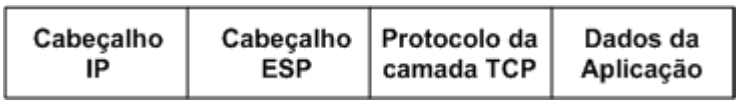


Figura 5.2 - Pacote protegido por ESP.

A Figura 5.3 representa um cabeçalho ESP que é iniciado com um campo SPI. O valor do SPI, combinado com o endereço de destino e o protocolo definido no cabeçalho IP, permitem identificar a SA apropriada para ser utilizada no processamento do pacote. O campo *número de sequência* fornece protecção para ataques de repetição ao ESP. O valor do *número de sequência* é um valor único e é sempre incrementado pelo emissor quando envia um pacote.

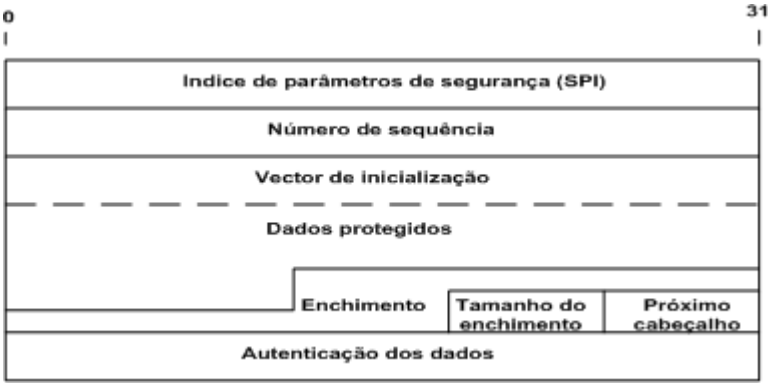


Figura 5.3 - Cabeçalho ESP.

A informação a ser protegida pelo ESP é incluída no campo de *dados* protegidos do pacote. Este campo também pode ser utilizado para armazenar um vector de inicialização necessário por determinado algoritmo criptográfico. O campo de *enchimento* é utilizado quando os modos de operação dos algoritmos criptográficos requerem que a sua entrada seja

uma cifra com um tamanho de valor múltiplo do tamanho do seu bloco. O campo de *autenticação dos dados* é utilizado para armazenar o resultado da verificação de integridade dos dados, devolvido normalmente por uma função de *hash* do pacote ESP. O tamanho deste campo depende do algoritmo de autenticação utilizado pela SA usada para processar este pacote. Caso não seja definido nenhum autenticador, este campo é eliminado. O campo *próximo cabeçalho* indica o tipo de dados que o ESP protege.

Se o ESP é utilizado em modo de transporte (Figura 5.4), o cabeçalho ESP é colocado entre o cabeçalho IP e o cabeçalho do protocolo da camada superior. Nesta situação, o campo *próximo cabeçalho* indica o tipo de protocolo da camada superior que se segue (por exemplo, o valor 6 indica que o protocolo é o TCP). Neste modo, o cálculo da integridade dos dados é obtido para o cabeçalho ESP, para o cabeçalho do protocolo da camada de transporte e para os dados; a integridade dos dados é obtida através de uma função de *hash*. Neste modo, o cabeçalho do protocolo da camada de transporte e os dados são enviados de forma protegida.

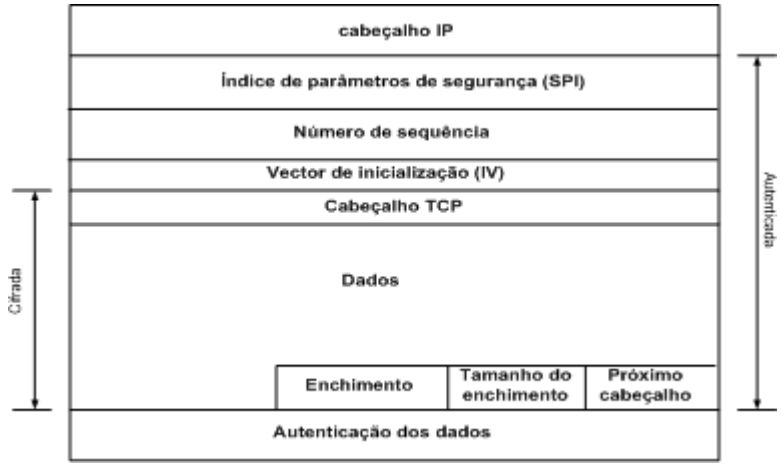


Figura 5.4 - Pacote IP protegido pelo ESP no modo transporte.

Se o ESP é utilizado em modo túnel (Figura 5.5), todo o datagrama IP é encapsulado por outro datagrama IP, e o cabeçalho ESP é colocado entre os dois datagramas. No modo túnel, o campo *próximo cabeçalho* indica encapsulamento IP sobre IP (terá o valor 4 numa implementação IPv4, ou o valor 41 numa implementação IPv6). Este modo é mais seguro, pois efectua a cifra de todo o datagrama IP original, e a integridade dos dados é obtida para o novo cabeçalho ESP e para o datagrama IP original. Tal como no modo transporte, a integridade dos dados é calculada por uma função de *hash*, e a protecção da informação é obtida por um algoritmo criptográfico simétrico.

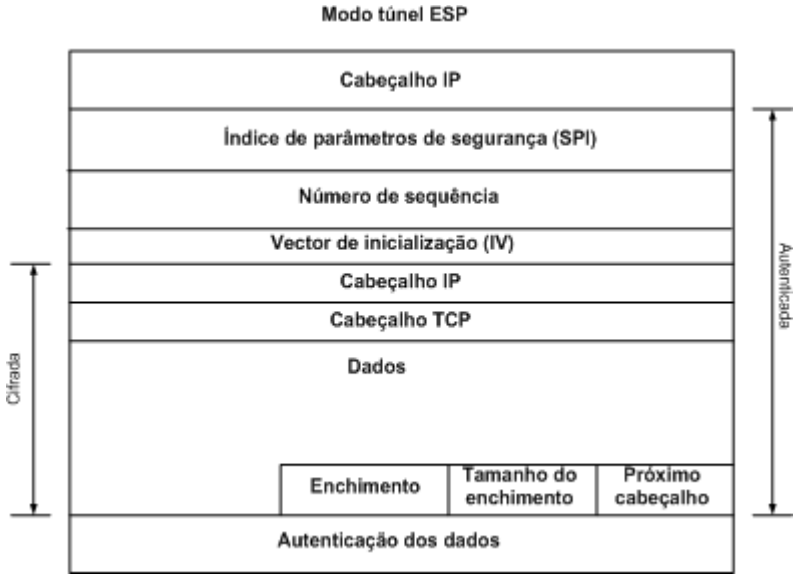


Figura 5.5 - Pacote IP protegido pelo ESP no modo túnel.

5.2.3. *Autentication Header* (AH)

O AH é um protocolo IPsec que fornece ao IP os serviços de integridade dos dados, autenticação da origem dos dados, e mecanismos limitados de anti-repetição de pacotes. O AH não fornece o serviço de confidencialidade, já que não fornece serviços de cifra de dados. O AH pode ser utilizado, tal como o ESP, para proteger um protocolo da camada superior (modo transporte) ou um datagrama IP completo (modo túnel). Em qualquer das situações, o cabeçalho AH é introduzido imediatamente a seguir ao cabeçalho IP. O campo de protocolo do cabeçalho IP tem um valor de 51. O cabeçalho AH está ilustrado na Figura 5.6.

O campo *próximo cabeçalho* indica o tipo de informação que se segue ao cabeçalho AH. Na situação do modo de transporte, indica o valor do protocolo da camada superior, por exemplo se é o TCP ou o UDP; no modo túnel indicará o tipo de encapsulamento.

O campo *SPI* permite identificar a SA usada para autenticar o pacote e o campo *número de sequência* permite impedir ataques de repetição. O campo de *autenticação dos dados* contém o resultado da função de verificação da integridade dos dados.

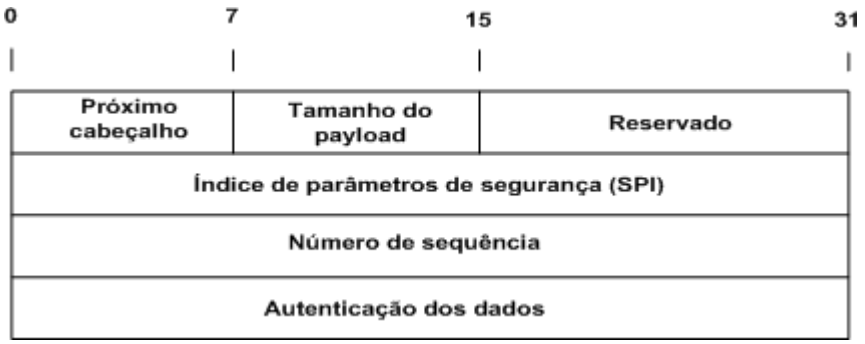


Figura 5.6 - Cabeçalho AH.

Tal como anteriormente referido, o AH pode ser utilizado no modo transporte ou no modo túnel. Quando utilizado no modo transporte, o AH protege as comunicações fim-a-fim. O cabeçalho AH é introduzido no pacote imediatamente a seguir ao cabeçalho IP e antes do cabeçalho do protocolo da camada superior. Quando utilizado no modo túnel, o AH encapsula o datagrama protegido. O cabeçalho AH é inserido entre dois cabeçalhos IP. O pacote IP interno mantém o endereçamento original da comunicação e o pacote IP externo contém os endereços dos extremos IPsec.

5.2.4. Utilização conjunta do AH e do ESP

O IPsec define também a possibilidade de um pacote IP ser protegido pelo protocolo AH juntamente com o ESP, em modo túnel e em modo transporte. Quando o AH e o ESP são utilizados de forma conjunta no modo de transporte, o ESP deve ser aplicado primeiro. Se o pacote a enviar for protegido primeiro pelo AH e depois pelo ESP, a integridade dos dados é apenas aplicável ao *payload* de transporte (TCP), já que o cabeçalho ESP é acrescentado depois de acrescentado o cabeçalho AH. Esta situação não é a desejável, já que a propriedade da integridade dos dados deve ser calculada para a maior quantidade de dados possível. Se o pacote IP é primeiro protegido com o ESP e só depois protegido pelo AH, a integridade dos dados é aplicável ao *payload* ESP que também contém o *payload* de transporte. A Figura 5.7 ilustra um pacote protegido com AH e ESP no modo de transporte. Neste modo, garante-se a o serviço de confidencialidade para o *payload* TCP, e o serviço de integridade para todo o datagrama.

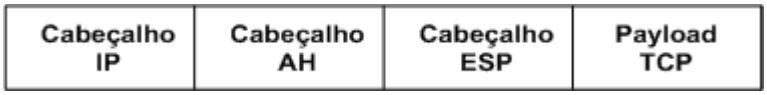


Figura 5.7 - Formato do pacote protegido pelo AH e ESP no modo transporte.

É também possível utilizar tunelamento encadeado de um pacote IP. Nesta situação não é importante a ordem de aplicação dos protocolos; apenas se deve ter em conta que os diferentes cabeçalhos IP não se devem sobrepor uns com os outros. A utilização de túneis encadeados deve garantir que um túnel exterior realmente proteja a ligação fim-a-fim de outro túnel. A Figura 5.8 ilustra um pacote protegido com o AH e o ESP no modo túnel. Neste modo, garantem-se os serviços de segurança confidencialidade, integridade para todo o datagrama IP original. A utilização de túneis encadeados é complicada e só deve ser utilizada em situações esporádicas e de fácil configuração.

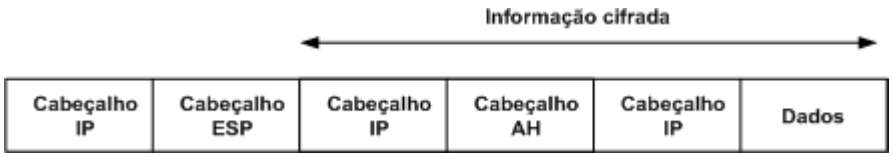


Figura 5.8 - Formato do pacote protegido pelo AH e ESP no modo túnel.

5.2.5. *Internet Security Association and Key management Protocol* (ISAKMP) e *Internet Key Echange* (IKE)

As acções a tomar relativamente ao tráfego que o sistema envia ou recebe são indicadas através das SAs. Para definir as SAs entre dois sistemas, o IPsec recorre ao ISAKMP e ao IKE. O ISAKMP [122] define de que forma dois sistemas comunicam, como são construídas as mensagens por eles trocadas; define também que transições de estado eles suportam de forma a comunicarem de forma segura. O ISAKMP fornece as formas de autenticação, o processo de troca de informação para uma troca de chaves, e os mecanismos de negociação dos serviços de segurança. No entanto, não indica como é realizada uma determinada troca de chaves, nem define os atributos necessários ao estabelecimento de SAs.

O IKE [121] define o mecanismo de troca de chaves. O IKE utiliza a forma de comunicação ISAKMP para definir uma troca de chaves e uma forma de negociar os serviços de segurança. Uma troca IKE tem como resultado final uma chave autenticada e um conjunto de serviços de segurança, ou seja uma IPsec SA.

Estes dois protocolos definem os mecanismos necessários para estabelecer, negociar, modificar e apagar SAs, definem a criação de chaves num determinado *Domain of Interpretation* (DOI). Um DOI representa o que deve ser negociado, define o esquema de nomes a utilizar por determinado protocolo, o conteúdo dos campos da ISAKMP SA, os atributos a negociar pelo IKE, e muitas outras características do IKE.

A criação da IPsec SA é efectuada da seguinte forma. Primeiro o IKE recorre ao ISAKMP para negociar uma IKE SA. De seguida, o IKE, em conjunção com o ISAKMP, protege a negociação da IPsec SA entre dois sistemas. Depois de criada a IPsec SA, os sistemas podem comunicar de forma segura.

O IKE define as seguintes fases para a troca de informação: a fase 1 ou *main mode* e a fase 2 ou *quick mode.* A fase *main mode* consiste na troca de seis mensagens que resultam na criação de uma IKE SA. Neste modo, separa-se a troca de informação relativa às chaves da troca de informação relativa à autenticação e identidade dos intervenientes. Existe também uma fase equivalente à *main mode* denominada de *aggressive mode*, que utiliza apenas três mensagens. A fase *aggressive mode* não é muito utilizada, já que não garante protecção acrescida à negociação da IKE SA. Após a negociação do IKE SA, o *quick mode* é utilizado

para negociar uma IPsec SA. Nesta negociação todos os campos, à excepção do cabeçalho ISAKMP, são cifrados. Neste modo é acrescentado o valor do código de *hash* imediatamente a seguir ao cabeçalho ISAKMP, permitindo autenticar a mensagem. A negociação *quick mode* está protegida pela IKE SA.

As mensagens trocadas pelo protocolo de gestão ISAKMP são construídas através de um cabeçalho ISAKMP comum (Figura 5.9), ao qual se acrescentam vários campos de informação diferentes (*payload*). Os campos *cookie iniciador* e *cookie receptor* são criados por cada par que intervém na comunicação, e são utilizados com o campo *mensagem ID* para identificar o estado que define a troca ISAKMP. O campo *cookie iniciador* é criado pelo emissor e o *cookie receptor* pelo receptor. Estes campos são calculados, com base no endereço IP do sistema, nos valores do protocolo e da porta, numa referência temporal e num valor secreto.

Figura 5.9 - Cabeçalho ISAKMP.

O campo *próximo payload* indica qual o tipo de *payload* que se segue ao cabeçalho ISAKMP. As versões do protocolo ISAKMP são indicadas pelos campos *versão maior* e *versão menor*. O campo *versão maior* indica a versão do protocolo ISAKMP em uso, o campo *versão menor* indica a actualização da versão do protocolo ISAKMP em uso. O *tipo de troca* ISAKMP pode ter o valor 2, caso a troca seja do tipo *main mode,* ou o valor 4, caso a troca seja do tipo *aggressive mode*. O tamanho da mensagem ISAKMP, incluindo o cabeçalho, é indicado no campo *tamanho da mensagem*. O campo *flags* fornece ao receptor informação adicional relativamente à mensagem. Estão definidas três *flags*: a *flag* E de cifra (*encryption)* que indica que os *payloads* que se seguem ao cabeçalho estão cifrados; a *flag* C de confirmação (*commit)* que indica que a negociação da SA está completa; e a *flag* A ou autenticação (*authentication)* que indica que a informação enviada pode ser autenticada mas não cifrada. O campo *mensagem ID* é um valor único que identifica a mensagem.

5.2.6. Sequência de negociação do protocolo IPsec

Antes de um pacote IP ser protegido por um dos protocolos de segurança IPsec (AH ou ESP), deve ser criada uma SA, efectuada através dos dois modos de operação do protocolo IKE, o modo *main mode* e o *quick mode*. No modo *main mode* efectua-se a autenticação e a identificação dos sistemas intervenientes. O *main mode* permite diversos métodos de autenticação: assinaturas digitais, chaves públicas (certificados) e chaves pré-partilhadas. O *quick mode* permite negociar as características e gerar as chaves de uma IPsec SA.

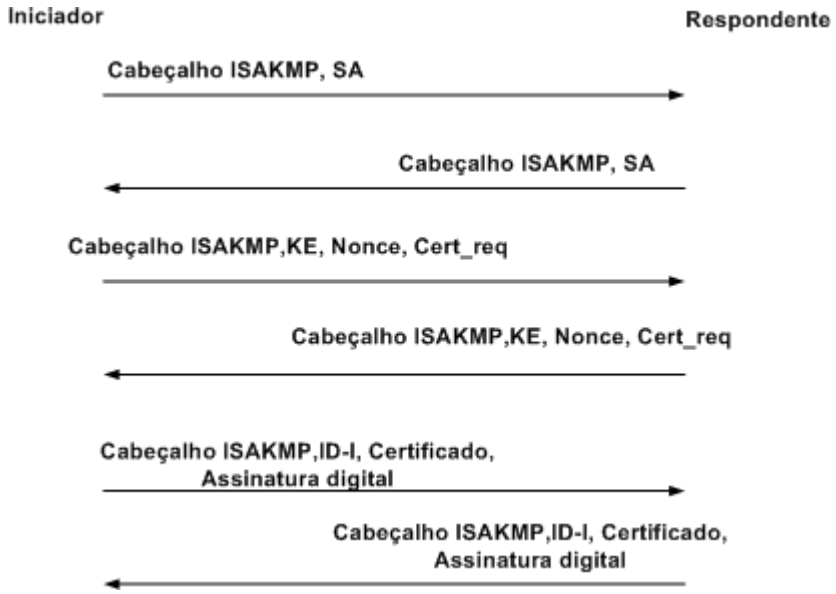


Figura 5.10 - Mensagens da fase *Main Mode.*

A Figura 5.10 representa a troca de mensagens para a fase *main mode*, em que a autenticação é baseada em chaves públicas. As chaves públicas são normalmente obtidas através de certificados; o IKE permite a troca e o pedido de certificados. Na primeira troca de mensagens, os intervenientes negoceiam os parâmetros de uma IKE SA e concordam em como serão efectuadas as restantes trocas. São também trocados, no cabeçalho ISAKMP de cada mensagem, os valores que identificam de forma unívoca cada um dos sistemas participantes na negociação IKE. Este valor é denominado de *cookie*. O iniciador escolhe o seu *cookie* na primeira mensagem e coloca-o no campo *initiator cookie* do cabeçalho ISAKMP; o receptor escolhe o seu *cookie* na segunda mensagem e coloca-o no campo *cookie responder* do cabeçalho ISAKMP. O valor *cookie* permite identificar cada um dos intervenientes na negociação; este valor é sempre igual durante toda a fase da negociação. Na primeira mensagem o iniciador indica também que *transforms* suporta. Uma *transform* representa as definições criptográficas suportadas pelo sistema; uma *transform* indica que algoritmo de cifra, que função de *hash*, que método de autenticação e que método de geração

de chaves é suportado por um sistema. Na resposta, o receptor, indica na segunda mensagem qual a *transform* a utilizar de entre as enviadas pelo iniciador.

No segundo par de mensagens os sistemas trocam as chaves públicas através de uma troca *Diffie-Hellman* e *nonces*. Nesta troca de mensagens são solicitados os certificados digitais para autenticação. O iniciador, através do campo KE (mensagem 3), indica ao receptor que será efectuada a troca de chaves e solicita o certificado do receptor (*Cert_req*). Da mesma forma, o receptor assinala o processo de troca de chaves e solicita o certificado do iniciador (mensagem 4). Estas mensagens incluem também *nonces* para permitir autenticar, identificar os intervenientes e proteger contra ataques de repetição.

Na última troca de mensagens, os sistemas identificam-se e autenticam-se. Estas mensagens são cifradas utilizando a *transform* anteriormente negociada e as chaves públicas trocadas no par de mensagens anterior. A mensagem enviada pelo iniciador ao receptor é cifrada utilizando a chave pública do receptor. Esta mensagem inclui o certificado, a identidade do iniciador, e uma assinatura digital na mensagem. Se o receptor conseguir decifrar a mensagem com a sua chave privada e obter a mesma assinatura digital, significa que o iniciador está autenticado: para conseguir calcular a mesma assinatura digital o sistema é capaz de obter o *nonce*, o que prova a sua identidade. O processo de autenticação e de identificação do receptor por parte do iniciador processa-se de forma idêntica. Conclui-se assim a fase de trocas de mensagens relativa ao *main mode*. Caso o *main mode* finalize com êxito, é criada uma IKE SA que será utilizada para proteger o *quick mode*.

A fase *quick mode* é apresentada na Figura 5.11. Todo o conteúdo das mensagens desta fase, à excepção do cabeçalho ISAKMP, é cifrado e autenticado pela IKE SA anteriormente negociada. Como é possível efectuar múltiplas trocas *quick mode* em simultâneo, introduziu-se um mecanismo para identificar cada mensagem com a correspondente troca. O campo *mensagem ID* do cabeçalho ISAKMP é utilizado para esse fim; cada troca *quick mode* deve ter um valor de *mensagem ID* único. As mensagens desta fase incluem um código de *hash* que permite verificar a integridade de cada uma delas.

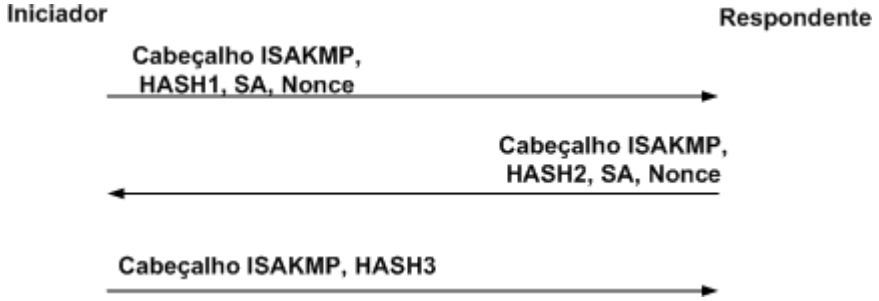


Figura 5.11 - Mensagens da fase *Quick Mode*.

O *quick mode* é essencialmente uma negociação de SAs e uma troca de *nonces* que permitem gerar novas chaves e proteger contra ataques de repetição. A fase *quick mode* termina com a negociação de uma IPsec SA. A IPsec SA é composta por duas associações de segurança – uma de entrada e outra de saída. Para cada associação de segurança são gerados SPIs diferentes, um escolhido pelo iniciador e outro pelo receptor. Este processo garante chaves de cifra diferentes para cada sentido.

5.2.7. Implementação IPsec/ VPN

Nesta secção apresenta-se a implementação de VPNs através de IPsec adaptada a uma rede sem fios. Inicialmente é apresentada a topologia da rede, os elementos constituintes e os protocolos suportados. De seguida, é descrito o funcionamento da solução de segurança e, finalmente, são apresentadas as suas vantagens e desvantagens ao nível da eficiência e complexidade. A Figura 5.12 ilustra os elementos constituintes de uma rede sem fios protegida através de IPsec.

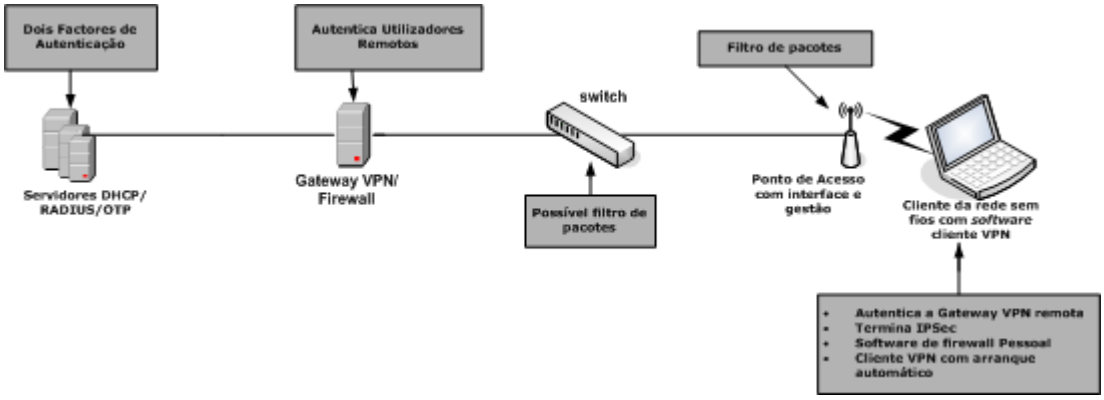


Figura 5.12 - Implementação VPN rede sem fios.

Esta rede contém utilizadores ligados a uma rede sem fios, com suporte de cliente VPN. O *software* cliente VPN deve permitir a criação de túneis cifrados entre os terminais finais sem fios e a *gateway* de acesso à rede com fios. A *firewall* garante a protecção ao nível dos dispositivos. Os sistemas operativos mais recentes da família *Windows*, o *Windows* XP e o 2003, incluem de forma nativa um cliente VPN. No entanto, este cliente apenas permite chaves de comprimento fixo de 1024 *bits*. A negociação ao nível dos protocolos inclui também muitas mensagens específicas deste tipo de sistemas, o que reduz de forma considerável a interoperabilidade com outros produtos. Ao nível dos protocolos criptográficos, a ligação nestes sistemas é normalmente estabelecida com o protocolo DES. Os terminais estão ligados aos pontos de acesso (*Access Point* - AP) que fornecem os filtros iniciais ao nível do protocolo IP, entre a rede sem fios e a rede com fios. Entre os APs e a rede com fios encontra-se um comutador de nível 2 que permite a sua conectividade. Os modelos

recentes permitem implementar a tecnologia *Virtual* LAN *Access Control Lists* (VACL), permitindo adicionar uma camada de filtros ao IPsec. Os utilizadores são autenticados através da *gateway* VPN/*firewall*. Esta *gateway* termina também os túneis e pode fornecer funcionalidades de reencaminhamento da informação DHCP para os clientes sem fios. A *gateway* encontra-se ligada a um conjunto de servidores: o servidor RADIUS autentica os utilizadores da rede sem fios (pode opcionalmente comunicar com o servidor *One Time Password* (OTP)); o servidor OTP autoriza a informação OTP enviada pelo servidor RADIUS; o servidor DHCP fornece a configuração IP aos clientes antes e depois do estabelecimento do túnel.

Os APs ligam-se aos comutadores de nível 2 numa VLAN dedicada apenas para clientes sem fios, e encaminham o tráfego IPsec dos clientes. O tráfego originado na rede sem fios é mantido separado do tráfego originado na rede com fios até ser descodificado pelo dispositivo de terminação da VPN. É importante referir que nesta implementação o WEP está desactivado. A rede sem fios é considerada não segura, sendo apenas utilizada como rede de transmissão de tráfego IPsec. Os clientes associam-se com o AP de forma a estabelecerem uma ligação com a rede na camada 2. De seguida, utilizam os serviços de DHCP e DNS para estabelecerem a conectividade ao nível da camada 3 da rede.

Após a configuração com êxito ao nível da camada 3, o túnel VPN autentica-se na *gateway* VPN. A *gateway* VPN pode utilizar certificados digitais ou chaves pré-partilhadas para obter a autenticação do dispositivo. Se a *gateway* utiliza chaves pré-partilhadas, recomendam-se as OTP para autenticação dos utilizadores. Sem as OTP, a *gateway* VPN fica vulnerável a tentativas de acesso à rede por "ataques de força bruta" de intrusos que tenham obtido a chave IPsec partilhada e utilizada pela *gateway* VPN. A *gatewa*y aproveita as vantagens dos serviços RADIUS, que pelo seu lado contactam o servidor OTP para autenticação do utilizador. A *gateway* VPN utiliza o servidor DHCP para a configuração IP de forma a que o cliente comunique através do túnel VPN.

Nas situações em que o servidor RADIUS ou a *gateway* VPN não estejam disponíveis, garante-se a segurança não permitindo o acesso à rede. Note-se que, antes de se estabelecer o túnel IPsec, quando os clientes da rede sem fios comunicam com a rede, todo o seu tráfego é considerado não seguro, aplicando-se aqui todos os problemas inerentes à falta de segurança nas redes sem fios. Existem três técnicas que permitem atenuar esses problemas. Uma delas é a introdução do *software* de *firewall* pessoal no cliente. Esta técnica permite proteger o dispositivo enquanto este está ligado à rede sem o reforço de segurança do IPsec. Outra técnica é criar a possibilidade de o cliente VPN estabelecer de forma automática o túnel

quando recebe o endereço IP correcto do servidor DHCP. Esta característica elimina a necessidade do utilizador estabelecer manualmente o túnel VPN após o arranque do dispositivo. Finalmente, podem-se implementar um conjunto de filtros no próprio AP para a utilização da rede sem fios. Recomenda-se a utilização de filtros restritivos, que apenas permitam os protocolos necessários ao estabelecimento de um túnel seguro até à *gateway* VPN. Incluem-se nesses protocolos: o DHCP, para a configuração IP inicial dos clientes; o DNS, para a resolução de nomes da *gateway* VPN; os protocolos específicos da VPN, o IKE (porta UDP 500), o ESP (protocolo IP 50) ou o AH (protocolo IP 51) e o ICMP (por questões relacionadas com detecção de falhas). Mesmo com todas estas restrições ao nível dos filtros, os servidores DHCP e DNS estão a descoberto de ataques directos nos protocolos aplicacionais; deve-se garantir que estes serviços estejam o mais seguros possível ao nível do sistema.

Existem várias possibilidades para a implementação do servidor VPN, vulgo *firewall*. Nesta dissertação referem-se apenas duas das hipóteses, uma utilizando uma única *firewall* (Figura 5.13) e a outra utilizando duas *firewalls* (Figura 5.14).

No cenário de uma única firewall, a rede interna fica separada da rede pública/rede sem fios por uma *firewall*. Os recursos acessíveis da rede pública, como servidores de correio electrónico ou *web*, estão colocados na rede interna, bem como a *firewall*. Geralmente, esta não é a aproximação mais segura já que se o servidor VPN ficar comprometido, toda a rede interna também o fica. Esta solução satisfaz as necessidades ao nível da autenticação e da cifra dos dados, mas não é a ideal, já que a segurança da rede interna depende fortemente da segurança do servidor VPN/*firewall*.

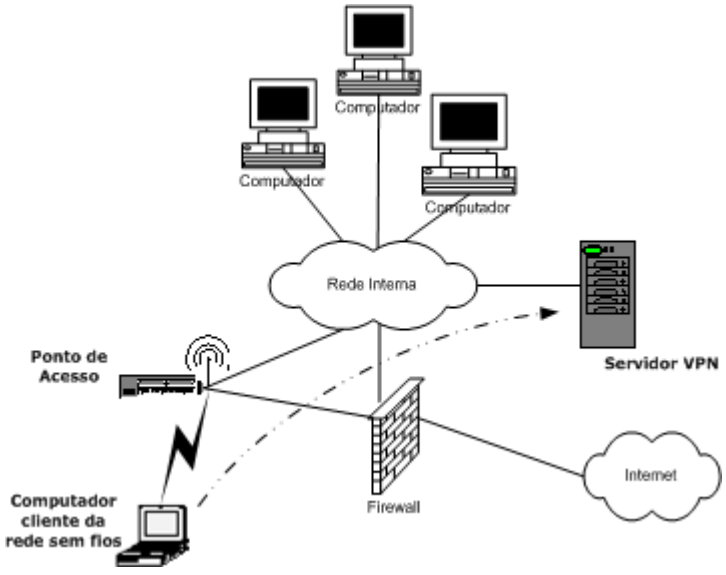


Figura 5.13 - Solução com uma única *Firewall*.

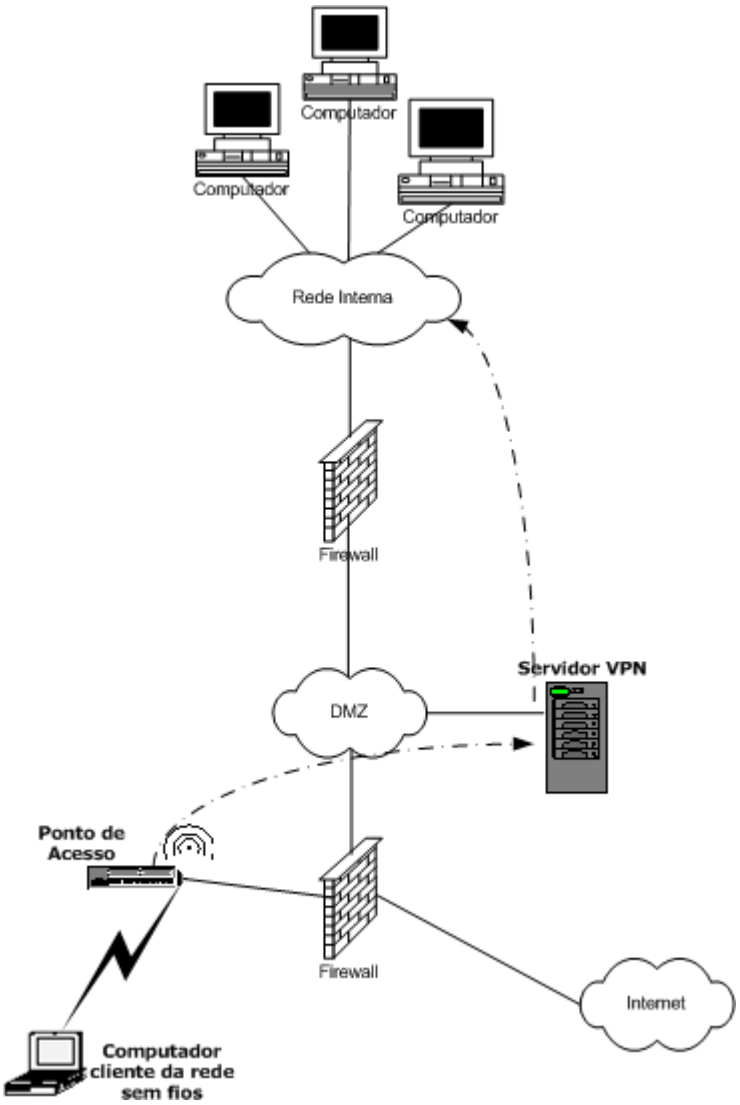


Figura 5.14 - Solução com 2 *Firewalls*, solução mais segura.

No cenário de duas *firewalls* coloca-se o servidor numa rede DMZ (*DeMilitarized Zone*). Uma DMZ é uma rede incluída entre a rede protegida e a rede externa, de forma a fornecer uma camada adicional de segurança. A DMZ é uma rede que se pretende objecto dos mais diversos ataques. Nesta situação, se o servidor VPN ficar comprometido, não dá ao atacante a vantagem de ter comprometido uma máquina interna (apenas compromete a DMZ, ou seja uma rede vazia).

5.2.7.1. Vantagens

As VPNs IPsec permitem eliminar ou atenuar os seguintes perigos: captura de informação na rede sem fios – esta ameaça é eliminada pela cifra IPsec de todo o tráfego na rede sem fios; *Man-In-The-Middle* (MITM) – esta ameaça é mitigada pela cifra IPsec e pela autenticação do cliente na rede sem fios; acesso não autorizado – os únicos protocolos permitidos entre a rede sem fios e a rede com fios são o DHCP para a configuração IP inicial,

e os de acesso à VPN (DNS, IKE e ESP); IP *spoofing* – os intrusos podem fazer *spoof* ao tráfego da rede sem fios, mas somente os pacotes IPsec válidos e autenticados é que atingirão a rede com fios; ARP *spoofing* – poderão ser realizados ataques de ARP *spoofing*; no entanto, como os dados são cifrados até à *gateway* VPN, os intrusos não podem ler os dados; ataques às palavras-chave – estas ameaças são eliminadas através de boas políticas de definição de palavras-chave, em que opcionalmente podem ser utilizadas OTP; descoberta da topologia da rede – apenas os protocolos descritos acima são permitidos entre a rede com e sem fios.

Além destas vantagens, a implementação de VPNs IPsec tem como vantagem não ser necessário adquirir novos adaptadores de redes sem fios para os computadores e/ou dispositivos móveis, podendo toda a restante estrutura de *hardware* manter-se inalterada.

5.2.7.2. Desvantagens

De uma forma geral, as VPNs são de administração e implementação complexa e difícil; devem utilizar um algoritmo de cifra forte e seguro como o 3DES, o que acrescenta processamento em toda a rede. Além disso, exigem também um período de regeneração de chaves que elimine as vulnerabilidades de comparação do texto com a sua cifra.

As vantagens das VPN podem desaparecer rapidamente se o *overhead* introduzido não for controlado e se não se criarem planos de contingência para evitar o estrangulamento da rede que daí resulta. As comunicações numa VPN implicam um acréscimo de 15 a 20% no *overhead* da rede (ver capítulo 7).

As VPNs também podem tornar-se inúteis se as definições de segurança nos sistemas ou nos dispositivos dos utilizadores ficarem comprometidas. Esta situação permitiria a um atacante recolher a informação necessária do dispositivo ou sistema e fazer uso dela para aceder aos recursos protegidos pela VPN. Para além disso, acrescenta necessidades de administração da rede, o que pode conduzir a falhas na gestão e implementação da rede.

Quando dois clientes da rede sem fios comunicam de forma directa, sem atravessar a *gateway*, é possível realizar ataques de ARP e IP *spoofing*, pois o IPsec garante apenas a protecção dos dados entre os clientes da rede sem fios e a *gateway* VPN.

No capítulo 7 são descritos alguns ataques a estas redes, e é avaliado o desempenho das mesmas ao nível da prevenção de ataques, eficiência, complexidade e *overhead* introduzido.

5.3. 802.1X/EAP-TLS – Autenticação baseada em portas

O IEEE 802.1X [123] é uma especificação de segurança em redes, inicialmente especificada para redes com fios, com os seus conceitos e utilização estendidos para redes

sem fios. O 802.1X define um controlo de acesso à rede baseado em portas: foi desenvolvido de forma a recolher informação de autenticação dos utilizadores (credenciais), e negar ou aceitar o pedido de acesso com base nessa informação.

O 802.1X baseia-se no *Extensible Authentication Protocol* (EAP), possibilitando deste modo a utilização de uma grande variedade de mecanismos de autenticação, sendo o mais aconselhado para redes sem fios o EAP-TLS. O controlo de acesso é realizado ao nível *Medium Access Control* (MAC) e é independente da camada física. No contexto do 802.1X, uma porta é qualquer tipo de acesso controlado (comutador, *router*, etc.). Dado que o 802.1X não foi inicialmente desenvolvido para as redes sem fios, foi necessário adaptar o conceito de porta a este tipo de redes. Deste modo, definiu-se que a associação entre um cliente e um AP é considerada uma porta virtual. O 802.1X garante também chaves por sessão e por cliente, e define que estas chaves devem ser alteradas regularmente, eliminando os problemas de repetição.

Uma solução 802.1X pretende fornecer as capacidades e as características necessárias à utilização de autenticação centralizada e à distribuição dinâmica de chaves. Esta aproximação baseia-se no trabalho desenvolvido pelo *Task Group "i"* (TGi) [124] do IEEE 802.11, no qual é indicada a utilização do 802.1X e do EAP para se conseguir obter essas funcionalidades adicionais.

Os três elementos principais da aproximação 802.1X/EAP são a autenticação mútua entre o cliente e o servidor de autenticação (RADIUS), as chaves de codificação/cifra derivadas dinamicamente após a autenticação, e a política de controlo centralizado, que inclui a activação de mecanismos de re-autenticação no fim de sessão e geração de novas chaves.

Quando estas características estão implementadas, um cliente que se associa a um AP, enquanto não efectuar um *login* na rede, não obtém acesso aos recursos da mesma. Depois de associado, o cliente e a rede (AP ou servidor RADIUS) trocam mensagens EAP de forma a realizarem autenticação mútua; o cliente verifica as credenciais do servidor RADIUS e vice-versa.

O 802.1X não é um protocolo de segurança, mas sim um processo de autenticação e de gestão de chaves. Numa rede sem fios, o 802.1X permite autenticação e geração de chaves de forma dinâmica. No entanto, a cifra dos dados é conseguida utilizando o protocolo WEP (utilizando o algoritmo de cifra RC4, um IV de 24 *bits* e o *checksum* para verificar a integridade dos dados). Após uma autenticação 802.1X bem sucedida, são geradas chaves de sessão iguais pelo cliente e pelo servidor de autenticação (esta chave é diferente para cada cliente autenticado com sucesso pelo 802.1X). O servidor de autenticação envia a chave de

sessão ao autenticador: é esta chave de sessão que é utilizada para protecção dos dados, entre cliente e o autenticador, através do protocolo WEP, caso este esteja activo.

5.3.1. Arquitectura 802.1X/EAP

O 802.1X define três tipos de entidades, ilustradas na Figura 5.15: o cliente, o controlador de acessos e o servidor de autenticação. Na terminologia 802.1X, estas entidades designam-se, respectivamente, de *supplicant*, autenticador e servidor de autenticação. O *supplicant* é a entidade que requisita serviços à rede. O autenticador (AP) é responsável por reforçar a autenticação do dispositivo que se liga à sua porta controlada, controlar o estado de autorização das suas portas, e encaminhar a informação para o servidor de autenticação. O servidor de autenticação autentica e autoriza o *supplicant*.

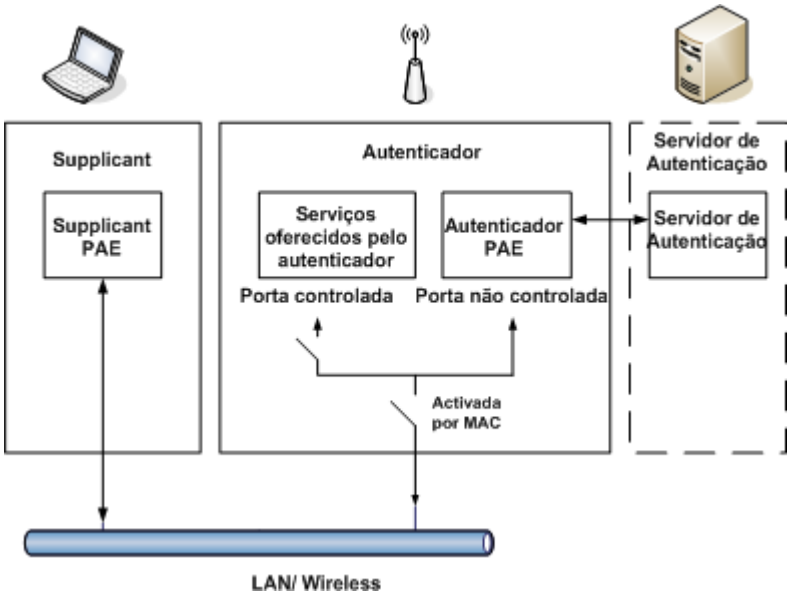


Figura 5.15 - Controlo de acesso baseado em portas.

O 802.1X define duas portas de acesso entre o cliente e o autenticador: uma porta controlada e uma porta não controlada (Figura 5.15). Por defeito, o *supplicant* é iniciado no seu estado não autorizado e apenas lhe é permitido comunicar com o autenticador através da porta não controlada. Esta porta filtra todo o tipo de tráfego à excepção das mensagens de autenticação. Os processos que executam os protocolos e algoritmos de autenticação numa porta designam-se por *Port Access Entity* (PAE). Existem PAEs do *supplicant* e PAEs do autenticador, cada um com tarefas bem definidas no processo de autenticação. Para a autenticação utiliza-se o protocolo EAP, sendo as mensagens de autenticação transportadas na parte da rede sem fios por um tipo especial de tramas EAP designadas por EAP *over* LAN (EAPoL) [123] (ver secção 5.3.2). Se o processo de autenticação é realizado com sucesso, a

porta controlada é activada permitindo o acesso a todo o tipo de tráfego; caso contrário, não é permitido o acesso à rede.

O cliente/*supplicant* é responsável por enviar as suas credenciais através de uma porta do autenticador ao servidor de autenticação. O processo de troca de informação de autenticação pode ser iniciado pelo cliente ou solicitado pelo autenticador ao cliente, dependendo de quem inicia a comunicação. O servidor de autenticação, normalmente um servidor RADIUS com suporte EAP, realiza funções de autenticação de forma a verificar as credenciais do cliente.

A Figura 5.16 descreve, de uma forma muito simplificada e resumida, a arquitecura 802.1X. Note-se que as mensagens do protocolo de autenticação são transportadas através de EAP entre o *supplicant* e o servidor de autenticação. No entanto, na rede sem fios, o EAP é transportado directamente sobre 802.1X, enquanto que na rede com fios, este é transportado através do RADIUS.

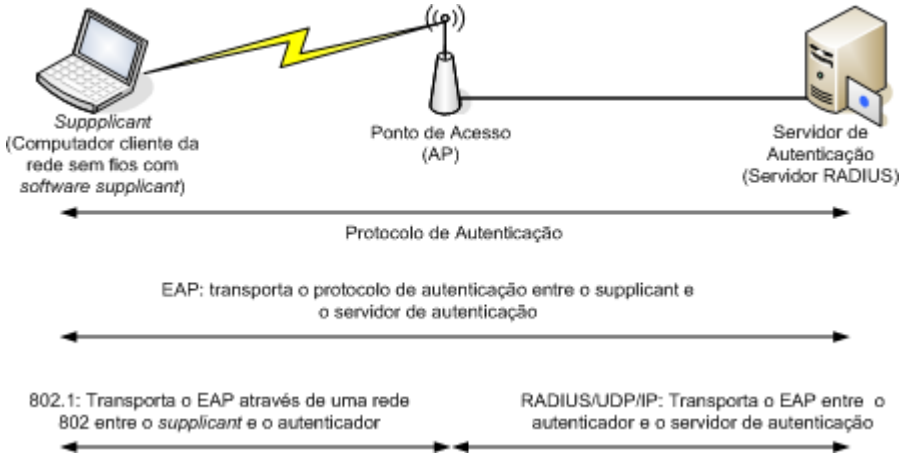


Figura 5.16 - Arquitectura 802.1X.

5.3.2. Funcionamento geral do 802.1X

Como o EAP foi originalmente desenvolvido para utilização em autenticação remota utilizando *modems*, não é um protocolo para redes locais. Sendo assim, foi necessário criar um mecanismo de encapsulamento das mensagens EAP, possibilitando assim a utilização das mesmas em redes locais. O 802.1X define um protocolo, denominado de EAP *over* LAN (EAPoL) que permite a troca de mensagens entre o *supplicant* e o autenticador.

Para se encapsular uma mensagem EAP é apenas necessário anexar no início da mensagem um cabeçalho MAC, permitindo assim o envio da mensagem na rede sem fios. O 802.1X define a forma como criar uma mensagem EAPoL (Figura 5.17), e os vários tipos de mensagens EAPoL. Existem cinco tipos de mensagens EAPoL: *EAPoL-start*, *EAPoL-key*,

EAPoL-packet, *EAPoL-logoff* e *EAPoL-encapsulated-asf-alert*. Apenas as quatro primeiras mensagens são utilizadas numa implementação 802.1X em redes sem fios.

Quando um cliente inicia o processo de autenticação não tem conhecimento do endereço MAC do autenticador. O cliente envia uma mensagem *EAPoL-start* para um endereço especial MAC de grupo *multicast* reservado para autenticadores 802.1X. Isto permite ao cliente descobrir se existe ou não um autenticador na rede, e também indica ao autenticador que o cliente está pronto a iniciar o processo de acesso à rede. Durante todo o processo de autenticação, e na comunicação entre o cliente e o autenticador são utilizadas as mensagens *EAPoL-packet;* as mensagens de pedido EAP (EAP-*Request)* e de resposta EAP (EAP-*Response*) são mensagens *EAPoL-packet*. Após se realizar o processo de autenticação com sucesso, o cliente e o servidor derivam uma chave de sessão. Esta chave de sessão é utilizada pelo WEP, se este estiver activo, para efectuar a protecção dos dados. Como o autenticador não conhece esta chave, o servidor na última mensagem do processo de autenticação (RADIUS *Access-Accept*), fornece-a ao autenticador. O autenticador fornece de seguida a chave de cifra *broadcast* (esta chave é utilizada para cifrar as mensagens de *broadcast* e de *multicast* entre vários clientes) ao cliente através da mensagem *EAPoL-key*. A chave de cifra *broadcast* é enviada de forma protegida, sendo cifrada com a chave de sessão gerada pelo cliente e pelo servidor. Depois deste processo, o cliente e o autenticador utilizam o WEP com as chaves anteriormente derivadas para comunicarem de forma segura. Quando o cliente se quer desligar da rede envia uma mensagem *EAPoL-logoff*.

A Figura 5.17 representa o formato genérico de uma trama EAPoL. O campo *tipo de pacote* indica se a mensagem é do tipo *EAPoL-start*, *EAPoL-key*, etc. O campo *corpo do pacote* assume o valor zero para as tramas *EAPoL-start* e *EAPoL-logoff*, pois estas tramas não necessitam de informação adicional. Para as tramas *EAPoL-packet* e *EAPoL-key*, o campo *corpo do pacote* assume um valor diferente de zero, pois nestas mensagens é enviada a informação de autenticação (certificados, pedidos de certificados) e as chaves de cifra.

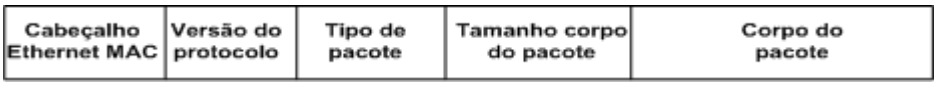


Figura 5.17 - Formato genérico da trama EAPoL.

5.3.3. Sequência de negociação do protocolo

O IEEE802.1X define vários mecanismos de autenticação baseados no protocolo de autenticação EAP. De uma forma geral, existe para cada um dos métodos de autenticação um conjunto típico de troca de mensagens 802.1X. As mensagens específicas de cada um desses

métodos de autenticação EAP estão descritas no capítulo 4; essas mensagens são utilizadas pelo processo 802.1X na fase de autenticação do cliente e do servidor.

A Figura 5.18 apresenta o conjunto de mensagens típicas necessárias para o estabelecimento de uma sessão 802.1X. A primeira mensagem é o envio, por parte do cliente, de uma mensagem *EAPoL-start* (*type*=*start* e *length*=0); esta mensagem permite também a associação entre o cliente e o AP. Depois do cliente se associar com o AP, este bloqueia todas as tentativas do cliente em obter acesso aos recursos da rede (por exemplo tráfego DHCP, HTTP e FTP). O AP força a porta de acesso do cliente para o estado não autorizada, sendo apenas possível o tráfego 802.1X. O AP envia então o pedido de identificação ao cliente (mensagem EAP *Request/Identity*). Depois de recebida esta mensagem, o cliente envia a sua identidade ao AP (mensagem EAP *Response/Identity*). O AP encaminha esta informação ao servidor de autenticação numa mensagem RADIUS *Access-Request*. Esta mensagem é apenas uma mensagem EAP encapsulada numa mensagem RADIUS; o atributo 79 representa o envio de uma mensagem EAP. De seguida, inicia-se a fase de autenticação entre o cliente e o servidor. Esta fase é constituída por um conjunto de mensagens de pedido e de resposta entre o cliente e o servidor, que dependem do protocolo de autenticação EAP utilizado (TLS, TTLS e PEAP). As secções 4.1.5.1, 4.1.5.2 e 4.1.5.3 descrevem as mensagens utilizadas para cada um desses métodos de autenticação. Nesta fase o AP funciona como encaminhador de informação entre o cliente e o servidor, encapsulando na parte da rede sem fios mensagens RADIUS em mensagens EAP, e na parte da rede com fios mensagens EAP em mensagens RADIUS.

Numa rede sem fios, e por questões de segurança, utiliza-se o mecanismo de autenticação EAP-TLS. Depois de receber e de verificar a identidade do cliente, o servidor indica ao cliente através de uma mensagem RADIUS *Access-Challenge* que vai ter início o processo de autenticação TLS (*Type*=TLS, *Flags*=*start*). O cliente, ao receber esta mensagem do autenticador, e se concordar com o método de autenticação, envia uma mensagem EAP-*Response*, onde indica o tipo de de autenticação EAP (EAP-*Type*=EAP-TLS). Esta mensagem inclui também num dos registos TLS uma mensagem *client hello.* A mensagem *client hello* inclui um número aleatório, o conjunto de primitivas criptográficas e os métodos de compressão suportados pelo cliente. O AP encapsula esta mensagem numa mensagem RADIUS e acrescenta um novo campo, o *Message Authenticator*. Este campo é um código de *hash* de toda a mensagem que permite ao servidor verificar se a mensagem está íntegra. Ao receber esta mensagem o servidor calcula o código de *hash* e compara-o com o recebido: se forem iguais aceita a mensagem. A partir deste momento e até ao final do processo de

autenticação as mensagens trocadas entre o AP e o servidor incluem sempre o campo *Message Authenticator*. O servidor responde ao cliente com uma mensagem RADIUS *Access-Challenge*. Esta mensagem, no seu atributo EAP *Message,* inclui uma mensagem *server hello,* o certificado do servidor (*certificate*), o pedido do certificado do cliente (*certificate_request*), informação adicional necessária à troca de chaves (*server_key_exchange*), e a indicação do fim da mensagem de *hello* (*server_hello_done)*. A mensagem *server hello* inclui uma identificação para a sessão, a versão do TLS suportado, o conjunto de primitivas criptográficas escolhidas de entre as enviadas pelo cliente, e um número aleatório. A mensagem *server_key_exchange* não é enviada, caso seja utilizado para troca de chaves o protocolo *Rivest Shamir Addleman* (RSA).

O cliente ao receber a mensagem obtém a identidade e a chave pública do servidor. O cliente gera um novo número aleatório: este número denomina-se de *pre-master secret*. O cliente utiliza uma função geradora de pseudoaleatórios que combina os números aleatórios obtidos pelo cliente e pelo servidor com o *pre-master secret* para obter a chave de sessão. O cliente envia ao servidor uma mensagem EAP *Response*. Esta mensagem contém o certificado do cliente (*certificate*); no campo *client_key_exchange* é enviado o *pre-master secret* cifrado com a chave pública do servidor; no campo *certificate_verify* o cliente envia um código de *hash* da mensagem, cifrada com a sua chave privada. Esta mensagem informa também o servidor que o cliente está preparado para utilizar as primitivas criptográficas anteriormente negociadas e a nova chave de sessão (*change_cipher_spec*); o cliente inclui também a indicação da troca de chaves bem sucedidas (*finished*).

O servidor utiliza a chave pública do cliente e decifra o calor do campo *certificate verify*. Para autenticar o cliente, o servidor calcula o código de *hash* da mensagem e compara-o com o valor obtido do campo *certificate verify*: se os valores forem iguais, o cliente é autenticado. O servidor utiliza a sua chave privada e decifra o valor do campo *client_key_exchange*, obtendo o valor do *pre-master secret*. Da mesma forma que o cliente, o servidor obtém a chave de sessão. O servidor envia então uma mensagem RADIUS *Access-Challenge* indicando que também está pronto a utilizar as primitivas criptográficas e a nova chave de sessão (*change cipher spec*); o servidor informa também o final do processo de autenticação (*finished*). O cliente, como não pretende mais informação do servidor, envia-lhe uma mensagem EAP *Response* sem dados e com a indicação do tipo de autenticação TLS *(*EAP *type*=EAP-TLS*).*

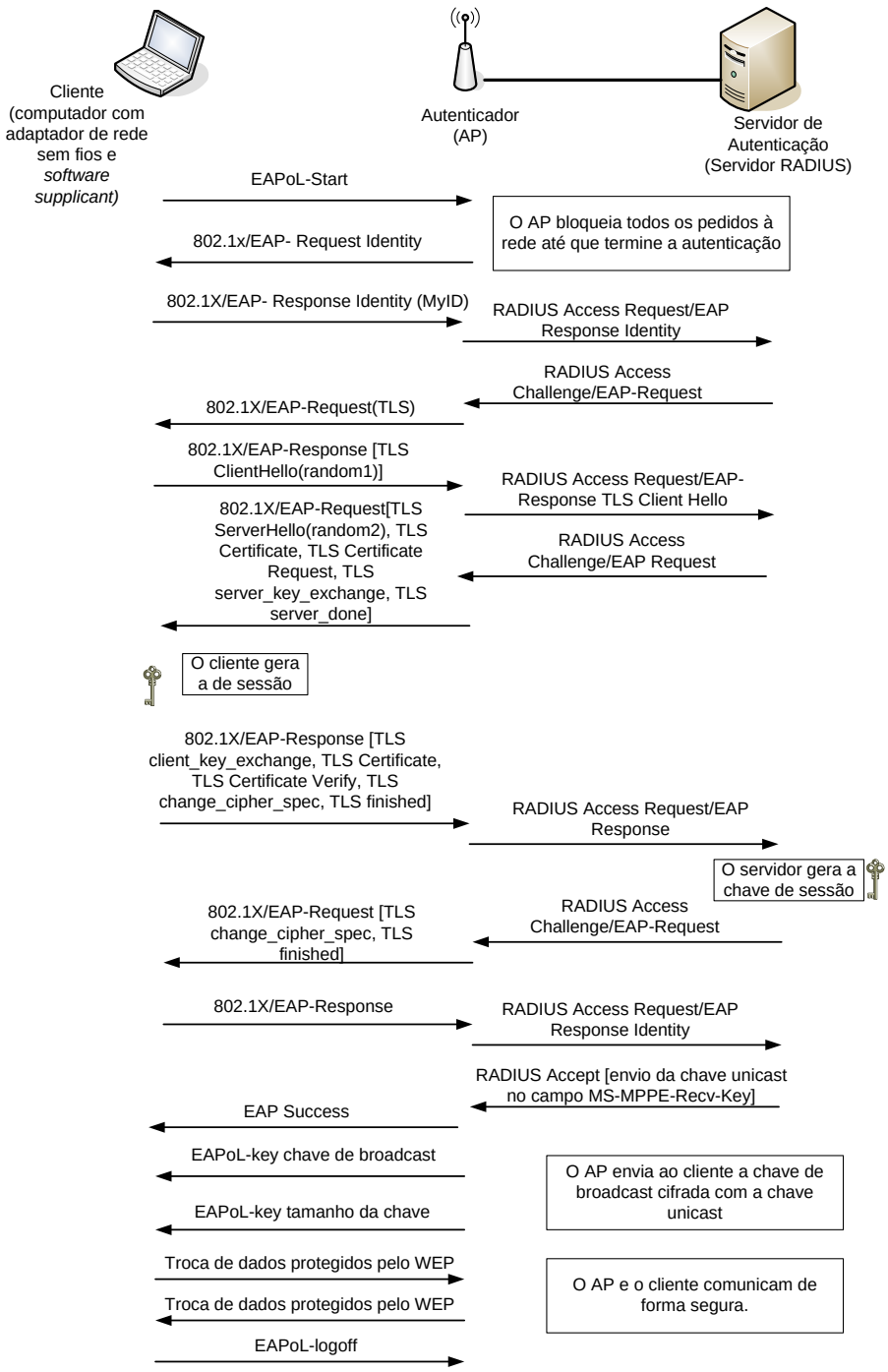


Figura 5.18 - Troca típtica de mensagens 802.1X.

Como a autenticação entre o cliente e o servidor foi bem sucedida, o servidor envia uma mensagem RADIUS *Access-Accept* ao AP. Nesta mensagem, o servidor no campo MPPE *Recv key* envia a chave de sessão ao AP: é esta a chave que será utilizada pelo WEP para proteger as comunicações entre o AP e o cliente. O AP, ao receber a mensagem RADIUS *Access-Accept*, altera o estado da porta para autorizada e envia ao cliente uma mensagem EAP *Success;* esta mensagem indica ao cliente que a partir desse instante todas as comunicações com o AP são realizadas de forma protegida utilizando o WEP e a chave de sessão. O AP activa o WEP com a nova chave e permite todo o tipo de tráfego entre o cliente e a rede. Para

permitir ao cliente enviar mensagens de *broadcast* e de *multicast*, o AP envia ao cliente uma mensagem *EAPoL-key* onde indica a chave de *broadcast*. Esta mensagem é cifrada utilizando a chave de sessão.

As chaves de protecção dos dados são utilizadas durante o tempo da sessão, ou até que seja atingido um determinado tempo limite. Por questões de segurança, sempre que é estabelecida uma nova sessão, é gerada uma nova chave de sessão; sempre que é atingido um tempo limite devem ser geradas novas chaves *broadcast* e *multicast*. Para terminar a comunicação o cliente envia apenas ao AP uma mensagem *EAPoL-logoff.*

5.3.4. Implementação 802.1X/EAP

Nesta secção apresenta-se a implementação de um sistema de autenticação 802.1X adaptado a uma rede sem fios. Inicialmente é apresentada a topologia da rede, os elementos constituintes e os protocolos suportados. De seguida é descrito o funcionamento da solução de segurança e, finalmente, são apresentadas as suas vantagens e desvantagens ao nível da eficiência e complexidade. A Figura 5.19 apresenta os elementos principais da rede. Esta rede tem um servidor RADIUS (existem implementações comerciais e em código aberto); os APs necessitam de suporte ao EAP/802.1X, o que normalmente se consegue por actualizações de *firmware*. Nas máquinas clientes é instalado o *software* cliente EAP, denominado de "*supplicant*", para suportar o tipo de autenticação EAP apropriada (também para os clientes existem implementações comerciais e em código aberto). Para ligar o AP e os servidores é também necessário um comutador de nível 2; para fornecer os endereços IPs ao cliente é necessário instalar um servidor de DHCP.

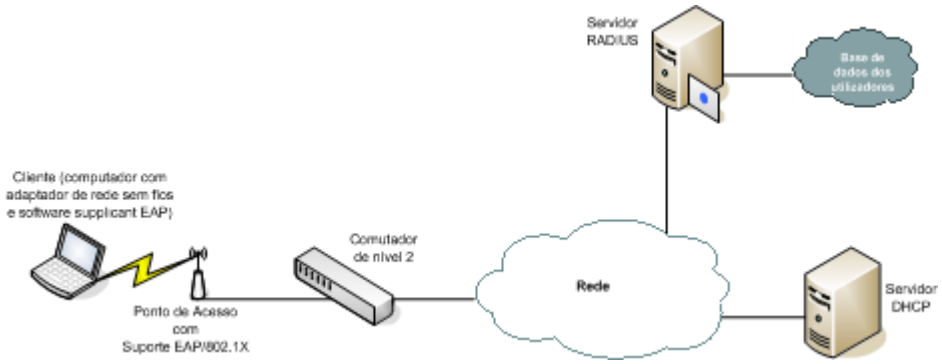


Figura 5.19 - Implementação 802.1X/EAP.

Na maioria dos casos, os APs estão ligados aos comutadores de nível 2 já existentes. Os servidores RADIUS e DHCP devem estar localizados no módulo relativo aos serviços da rede. Garante-se a segurança prevenindo o acesso à rede de clientes não autorizados, incluindo a eventualidade de falha de serviço do servidor RADIUS; deve-se garantir esta opção, pois grande parte da capacidade de segurança do sistema assenta no serviço RADIUS.

Os clientes e os APs utilizam EAP para autenticarem os dispositivos da rede sem fios e os utilizadores com o servidor RADIUS. Por questões de escalabilidade e de gestão, os dispositivos são configurados para utilizar o protocolo DHCP para obterem a configuração IP. O estabelecimento das comunicações com o servidor DHCP só acontece após o dispositivo e os utilizadores serem autenticados com sucesso através do protocolo EAP. Após a configuração DHCP estar completa, o utilizador tem acesso à rede corporativa e, se estiverem implementados filtros, estes são aplicados. De forma a prevenir os ataques devido às colisões dos vectores de inicialização, recomenda-se a regeneração das chaves de cifra, tanto para mensagens *unicast* como de *broadcast*.

5.3.4.1. Vantagens

Com este tipo de implementação fica em evidência um conjunto de vantagens. Por um lado, elimina-se o perigo dos *Sniffers* de pacotes devido à codificação da informação, e elimina-se o perigo de acesso não autorizado à rede, pois apenas os utilizadores autenticados é que lhe têm acesso. Além disso, a natureza da autenticação mútua de diversos tipos de autenticação EAP combinados com o ICV permitem prevenir que intrusos se coloquem no caminho das comunicações, eliminando o perigo de ataques MITM. Com estes mecanismos, os intrusos não podem realizar IP *Spoofing* sem primeiro se autenticarem na rede (deve-se, no entanto, garantir a implementação de processos de filtragem na camada 3). Uma outra vantagem consiste no facto de, se um intruso não se conseguir autenticar na rede sem fios, não pode realizar ARP *Spoofing*. Finalmente, se não for possível a um intruso realizar autenticação, não pode realizar descoberta da rede, pois fica limitado aos protocolos que não são filtrados.

5.3.4.2. Desvantagens

Uma das desvantagens desta solução é a possibilidade de realização de ataques às palavras-chave, pois um atacante pode monitorar as trocas de mensagens EAP/802.1X entre o cliente da rede sem fios e o AP. Para garantir uma segurança forte a este nível devem ser implementados métodos de autenticação bastante seguros, tais como o OTP. Esta solução também não permite a utilização de chaves de cifra por pacote. A chave de cifra, embora diferente para cada sessão, não é diferente para cada pacote enviado na sessão. Como é utilizado o WEP (algoritmo RC4 + IV de 24 *bits*), é possível que após algum tempo de utilização da cifra um atacante possa descobri-la.

A inexistência de autenticação mútua entre o cliente e o AP (apenas existe autenticação mútua entre o servidor e o cliente) permite, mediante a utilização de um AP não

autorizado, expor o cliente a ataques do tipo MITM. A utilização de um AP não autorizado permite também realizar ataques de desvio de sessão e de *Denial of Service* (DoS).

Devido a todas estas desvantagens existe a necessidade de desenvolver mecanismos de segurança mais robustos e eficazes.

5.4. WPA (Wi-Fi Protected Access)

A *Wireless-Fidelity Alliance*[4] [87] tem promovido a criação de uma nova norma designada *Wi-Fi Protected Access* (WPA) [46]. A norma 802.11 que está na base do desenvolvimento da solução WPA é a norma 802.11i [117]. No entanto, o WPA é compatível e pode ser utilizado como solução de segurança em todas as variantes 802.11, incluindo 802.11a, b e g.

Comparativamente ao 802.1X, o WPA introduz a autenticação mútua entre o cliente e o AP, um mecanismo robusto de verificação da integridade das mensagens e a geração efectiva de chaves por pacote.

O WPA garante a segurança em redes *Wi-Fi* com a introdução de um algoritmo de cifra robusto, bem como a introdução de autenticação por utilizador, uma das características não disponíveis no WEP. Quando instalado de forma correcta, garante-se que os dados de cada utilizador permanecem protegidos, e que apenas utilizadores autorizados podem aceder aos recursos da rede.

A finalidade do WPA é ser um forte substituto do WEP, interoperacional com o WEP, actualizável por *software* nos produtos *Wi-Fi* já existentes, aplicável a nível empresarial e doméstico e disponível de forma imediata. A actualização de *software* para suporte do WPA inclui a actualização dos APs e das placas de rede dos terminais, e possivelmente do seu sistema operativo. O WPA mantém ainda compatibilidade futura com equipamentos 802.11i.

Para melhorar a codificação dos dados, o WPA utiliza o *Temporal Key Integrity Protocol* (TKIP) [117] que fornece melhorias consideráveis ao nível da cifra dos dados com chaves temporárias. O WPA inclui ainda uma função de mistura de chaves por pacote de dados, uma mensagem para verificação da integridade dos dados (MIC), vectores de inicialização (IV) estendidos com regras de sequenciação, e um mecanismo de renovação de chaves. O MIC, ao contrário do ICV do WEP, não é obtido recorrendo a um método linear, o que realmente garante a protecção de integridade das mensagens. O WPA utiliza IVs de 48 *bits*, o que permite eliminar muitos dos defeitos apontados aos IVs utilizados no WEP.

[4] Organização industrial que certifica a interoperabilidade dos dispositivos baseados na norma 802.11. Inclui um grupo de fornecedores de tecnologia: *Cisco Systems, Microsoft, Proxim/Agere e Symbol Technologies*

O TKIP aumenta o tamanho da chave de cifra de 40 para 128 *bits* e substitui a chave única e estática do WEP por chaves que são geradas dinamicamente e distribuídas pelo servidor de autenticação. O TKIP utiliza uma metodologia de hierarquias e de gestão de chaves que elimina a previsibilidade das mesmas.

O MIC destina-se a prevenir que um intruso obtenha pacotes de dados, os altere e volte a reenviá-los. O MIC fornece uma função matemática pela qual o receptor e o emissor efectuam o cálculo e comparação da MIC: se o resultado das funções no emissor e receptor for diferente, assume-se que os dados foram violados e o pacote é eliminado.

A utilização de IV estendidos elimina o problema de re-utilização de IV, tornando mais difícil para um atacante descobrir a chave de cifra. As regras de sequenciação dos IVs eliminam o perigo de ataques de repetição; sempre que uma mensagem contiver um IV com um número de sequenciação de uma mensagem mais antiga é eliminada.

O WPA inclui autenticação do utilizador através do 802.1X, e assim como na solução anterior, pode ser utilizado qualquer um dos métodos EAP disponíveis. Com estas tecnologias garante-se uma autenticação robusta do utilizador: para tal é necessário um servidor central de autenticação, que garanta autenticação mútua de forma a que um utilizador não adicione à rede um AP não reconhecido pela mesma.

Nas secções seguintes apresenta-se o princípio de funcionamento do WPA e dos seus módulos associados, a descrição breve da implementação de uma rede com WPA, suas vantagens e desvantagens.

5.4.1. *Temporal Key Integrity Protocol* (TKIP)

O TKIP elimina, utilizando *hardware* já existente e mantendo o protocolo criptográfico RC4, algumas das falhas do WEP. Para aumentar a segurança, altera a forma como as chaves são determinadas, e efectua uma maior rotação de chaves. Ao contrário do WEP, o TKIP fornece mecanismos de gestão dinâmica de chaves. Apresentam-se a seguir as principais orientações que levaram ao desenvolvimento do TKIP: nunca re-utilizar o mesmo vector de inicialização (IV) com a mesma chave de sessão – dois pacotes diferentes nunca deverão ser cifrados com a mesma chave, de modo a garantir protecção contra ataques de colisão (re-utilização de chaves); utilizar um número de sequência e rejeitar pacotes que são recebidos fora de ordem, garantindo-se assim protecção contra ataques de repetição; geração automática de chaves aleatórias – a chave de sessão deve ser gerada antes de iniciado o contador de IV; geração de chaves por pacote – de modo a evitar geração de chaves fracas, as chaves de sessão são processadas por um esquema de mistura de chaves, de forma a se gerarem chaves por pacote, que depois são utilizadas pelo gerador RC4; nova função de

integridade da mensagem – é utilizada uma função de *hash* criptograficamente segura em substituição do valor linear CRC utilizado no WEP.

Os mecanismos que implementam o TKIP podem ser divididos em três componentes principais: MIC, sequenciação IV/extensão do IV e geração de chaves por pacote/distribuição de chaves. Descrevem-se a seguir esses componentes.

5.4.1.1. *Message Integrity Code* (MIC)

O ICV no WEP era obtido por um algoritmo linear e era apenas função dos dados a enviar. Para resolver esse problema, o TKIP utiliza uma função de *hash* segura, denominado de MIC. O MIC utiliza uma chave denominada chave MIC. Esta chave é diferente nos dois sentidos da comunicação, e é computacionalmente independente das chaves criptográficas utilizadas.

A Figura 5.20 representa o processo de geração da MIC. A geração do MIC, tal como indica a figura, depende de um algoritmo que é função do endereço MAC de destino, do endereço MAC de origem, da informação a enviar e da chave MIC. O algoritmo gera um código de *hash* com 64 *bits* (8 *bytes*); este código é depois anexado à trama antes de se iniciar a cifra dos dados. Como o MIC é uma função dos endereços MAC de origem e destino, a trama fica associada ao emissor e ao receptor, eliminando-se assim a possibilidade de ataques por falsificação.

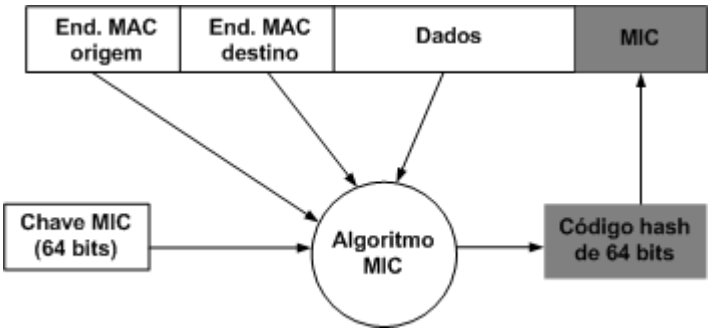


Figura 5.20 - Determinação do MIC.

O receptor de uma trama cifrada determina o MIC. Se o resultado for igual ao MIC enviado, então garante-se que a mensagem é autêntica; caso contrário a mensagem foi falsificada. Caso ocorram duas falsificações de mensagem no mesmo segundo, as chaves devem ser apagadas, e o cliente deve desassociar-se do AP e voltar a realizar o processo de autenticação.

5.4.1.2. IV estendido/ Novo contador de sequencia de IV

O TKIP utiliza um IV de 48 *bits*, enquanto que o WEP utiliza um IV de apenas 24 *bits*: este aumento do tamanho do IV permite aumentar a probabilidade de obter pares chave-

IV únicos. O novo IV do TKIP é obtido com o primeiro e o último *byte* do IV WEP e um novo IV de quatro *bytes* (Figura 5.21). A trama que se obtém utilizando o TKIP é doze *bytes* maior que a trama WEP (4 *bytes* para o IV estendido e 8 *bytes* para o MIC), o que representa um aumento no *overhead* introduzido.

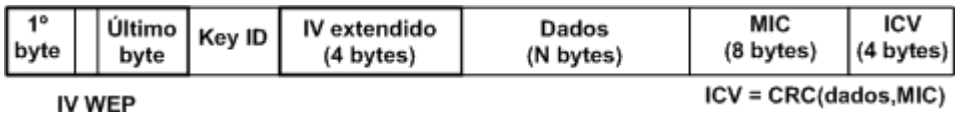


Figura 5.21 - Trama TKIP.

Os dois *bytes* obtidos do IV do WEP são combinados de modo a serem utilizados como contador de sequência. Este contador é denominado TKIP *Sequence Counter* (TSC) e é ele que permite prevenir contra ataques de repetição. A utilização do TSC é definida por um conjunto de regras: inicia-se a sequência a zero sempre que se estabelecerem novas chaves; o TSC é aumentado de um em cada pacote; a transferência de informação termina se o TSC atingir o seu valor máximo; e o receptor rejeita qualquer pacote recebido fora da sequência.

5.4.1.3. Geração de chaves por pacote/Gestão de chaves

O objectivo da geração de chaves por pacote é eliminar a utilização de chaves fracas. Esta chave é obtida pela combinação do endereço do emissor, do IV de 48 *bits* e de uma chave de 128 *bits* denominada de chave temporal. O nome chave temporal advém do facto de que essa chave é alterada assim que é re-iniciada a contagem de sequência. A chave por pacote é então utilizada para cifrar os dados, o MIC e o ICV.

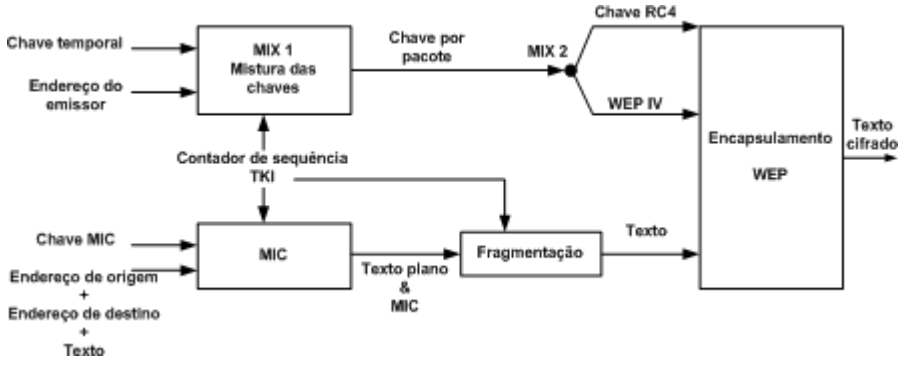


Figura 5.22 - TKIP: computação do MIC e geração de chaves por pacote.

A obtenção da chave temporal é o resultado de um processo hierárquico de gestão de chaves. O protocolo de autenticação da camada de rede permite obter uma chave de sessão denominada *Master Key* (MK); essa chave é depois utilizada para obter a *Pairwaise Master Key* (PMK), que por sua vez é utilizada para se obter a chave temporal, designada por *Pairwaise Transient Key* (PTK), que será utilizada para cifrar os dados. O processo de geração das chaves é apresentado na secção 5.4.3.

O processo de geração de chaves por pacote encontra-se ilustrado na Figura 5.22. A chave temporal não é directamente utilizada para cifrar os dados: esta passa por um processo que elimina os padrões que podem ser usados para se realizarem ataques às chaves fracas. A fase 1 da função de mistura (MIX 1) é utilizada para eliminar a possibilidade de utilização da mesma chave em todas as ligações, a fase 2 da função de mistura (MIX 2) é utilizada para eliminar a relação entre a chave temporal e o IV [31].

5.4.1.4. Protocolo de autenticação 802.1X

Um dos maiores defeitos do WEP é o facto da identidade de um cliente nunca ser na realidade validada através de qualquer tipo de integridade. O WPA elimina este defeito utilizando autenticação na camada superior, ou seja na camada de rede. O protocolo utilizado para implementar essa autenticação é o 802.1X. A *Wi-Fi Alliance* recomenda para autenticação na camada de rede a implementação 802.1X baseada no protocolo de autenticação EAP-TLS.

Uma autenticação EAP-TLS bem sucedida tem como resultado a determinação de uma chave de sessão; que será depois utilizada, tal como referido na secção anterior, para dar origem à chave temporal que permitirá cifrar os dados.

5.4.2. Arquitectura de autenticação e gestão de chaves

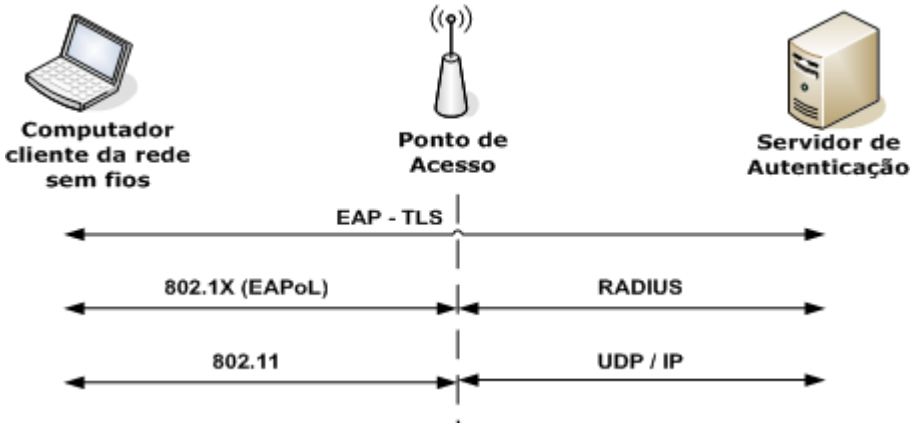


Figura 5.23 - Arquitectura de autenticação e gestão de chaves do WPA.

Nesta arquitectura, ilustrada na Figura 5.23, o protocolo utilizado para efectuar a autenticação entre o cliente e o AP é o EAP (EAP-TLS). O protocolo RADIUS é o protocolo de autenticação utilizado entre o AP e o servidor. Esta arquitectura é semelhante à das secções anteriores (802.1X).

Na sua arquitectura de gestão de chaves, o WPA define uma hierarquia de chaves. A hierarquia de chaves é constituída por dois tipos de chaves, a PTK com 512 *bits* e a *Group Transient Key* (GTK) com 256 *bits*. Assim que seja efectuada com sucesso a autenticação

entre o servidor de autenticação e o cliente, é derivada uma chave de sessão (MK). Como o cliente pretende efectuar comunicações seguras com o autenticador, o servidor de autenticação e o cliente obtêm, aplicando uma função geradora de pseudoaleatórios à MK, uma nova chave denominada PMK. O envio desta chave por parte do servidor de autenticação ao autenticador indica a este último que o cliente tem autorização para aceder ao meio 802.11. É agora possível ao autenticador e ao cliente derivarem as chaves de protecção das comunicações, denominadas de PTK. A PTK é um conjunto de quatro chaves: a *Key Integrity Key* (KIK), a *Key Encryption Key* (KEK), a *Data Integrity Key* (DIK) e a *Data Encryption Key* (DEK). Estas chaves são denominadas de chaves temporárias, pois sempre que um cliente se associa com um AP são determinadas novas chaves; todas estas chaves têm um tamanho de 128 *bits*. A KIK é utilizada durante a sessão estabelecida entre o autenticador e o cliente como prova de que o autenticador e o cliente possuem a PMK, ou seja associa a PMK a cada um dos sistemas intervenientes na comunicação. A DEK e a DIK são utilizadas, respectivamente, para cifrar todo o tráfego entre o autenticador e o cliente e para proteger os dados contra modificações. A Figura 5.24 apresenta a hierarquia de chaves do WPA.

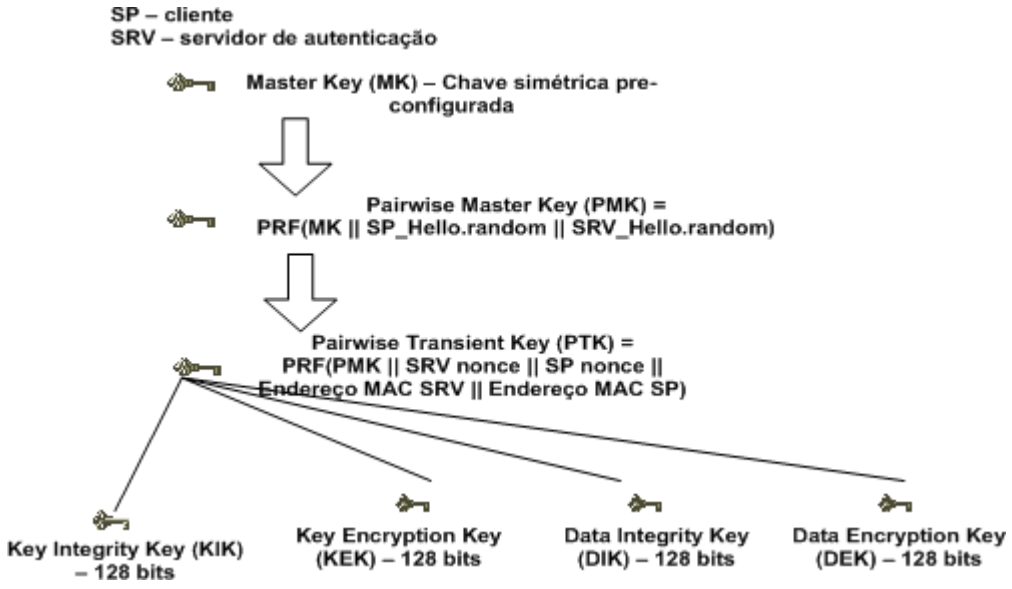


Figura 5.24 - Hierarquia de chaves WPA.

Como as redes sem fios permitem, dada a sua natureza, o envio de mensagens de *broadcast* e de *multicast*, é também definida uma chave, de 256 *bits*, utilizada para proteger este tipo de tráfego, denominada de GTK; a KEK é utilizada para distribuir de forma segura a GTK do autenticador ao cliente.

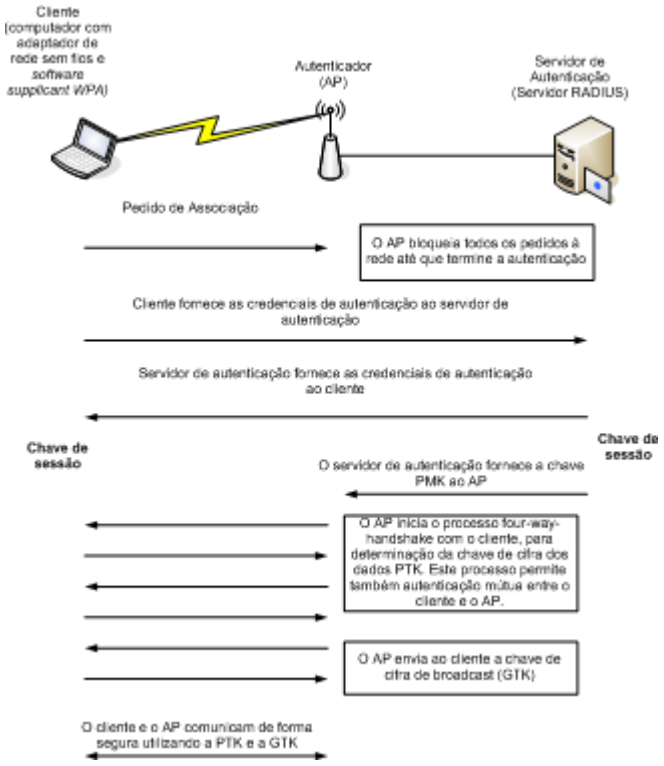


Figura 5.25 - Funcionamento do WPA.

5.4.3. Fases Operacionais do WPA

O WPA é caracterizado por definir para o seu funcionamento quatro fases: a fase da descoberta das capacidades de segurança entre o AP e o cliente, a fase de autenticação 802.1X entre o cliente e o servidor de autenticação, a fase da distribuição de chaves RADIUS do servidor de autenticação para o autenticador, e a fase de gestão das chaves 802.1X entre o autenticador e o cliente. Estas fases estão apresentadas na Figura 5.25.

A fase da descoberta das capacidades de segurança tem como objectivo determinar possíveis pares com quem estabelecer comunicações e anunciar as capacidades de segurança da rede aos clientes. Nesta fase, o cliente envia um pedido de associação ao AP; o AP bloqueia todos os pedidos de acesso à rede, e apenas permite o envio de mensagens 802.1X. Nesta fase, o AP indica ao cliente que protocolo de cifra é utilizado para proteger as comunicações. A fase de autenticação 802.1X tem como objectivo tornar o servidor num centro de decisão das políticas de acesso à rede e efectuar autenticação mútua entre o cliente e o servidor. Para ser possível a autenticação mútua, o cliente e o servidor trocam mensagens 802.1X. Nestas mensagens é enviada toda a informação relativa às credenciais do servidor de autenticação e do cliente. Nesta fase, são também geradas a MK e a PMK. A fase da distribuição de chaves tem como objectivo o envio da chave PMK do servidor de autenticação ao AP. Finalmente, a fase da gestão de chaves tem como objectivos: associar a PMK ao AP e ao cliente, permitir que tanto o AP como o cliente confirmem que conhecem a PMK

(autenticação mútua entre o AP e o cliente), gerar e sincronizar a utilização da chave PTK (processo conhecido por *four-way handshake*), e efectuar a distribuição da GTK.

5.4.3.1. Fase da descoberta

A Figura 5.26 ilustra a fase de descoberta entre o AP e o cliente. Nesta fase, o cliente efectua um pedido de associação aos APs disponíveis na rede; este pedido apenas indica o nome da rede (*ssid*) à qual o cliente pretende aceder. O AP, ao receber o pedido de associação, responde ao cliente e anuncia quais as suas capacidades de segurança, indicando qual o protocolo suportado para cifra dos dados (protocolo TKIP) e qual o tipo de autenticação pretendida (autenticação 802.1X). Depois é efectuada, e em sistema aberto, uma autenticação 802.11. Esta autenticação, consiste apenas no envio por parte do cliente do *ssid* da rede, e pela verificação do *ssid* por parte do AP. Se o *ssid* enviado pelo cliente, coincidir com o do AP, este responde ao cliente com uma mensagem de autenticação 802.11 bem sucedida. De seguida, o cliente envia um novo pedido de associação, mas neste pedido inclui também as suas capacidades de segurança, respondendo da mesma forma que o AP. Ao receber este novo pedido do cliente, com as capacidades de segurança inicialmente indicadas, o AP responde com uma mensagem de associação bem sucedida, concluindo assim a fase da descoberta.

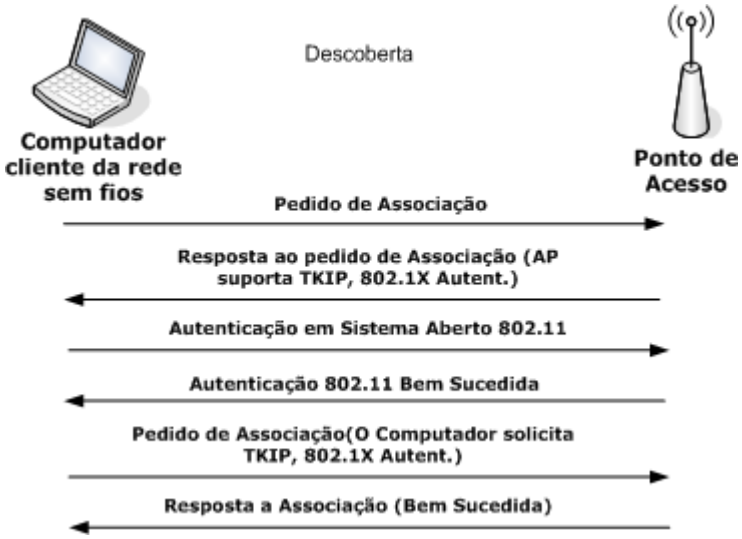


Figura 5.26 - Fase da Descoberta WPA.

5.4.3.2. Fase de autenticação 802.1X

Após a associação bem sucedida entre o AP e o cliente, tem início a fase de autenticação 802.1X (Figura 5.27). Uma característica importante desta fase é o facto de tanto o cliente como o AP bloquearem todo o tipo de tráfego à excepção do tráfego EAP. A autenticação 802.1X é realizada entre o cliente e o servidor de autenticação: todas as

mensagens EAP entre o cliente e o servidor de autenticação, que circulam entre o autenticador e o servidor, são encapsuladas pelo autenticador no protocolo RADIUS. As mensagens EAP trocadas nesta fase dependem do método de autenticação EAP utilizado. Esta fase termina após autenticação bem sucedida entre o cliente e o servidor, com a geração da chave de sessão, denominada PMK. Como esta chave é apenas conhecida entre o servidor e o cliente, é gerada através da aplicação de uma função geradora de pseudoaletaórios à MK. A PMK é utilizada pelo AP e pelo cliente para protegerem as suas comunicações.

A Figura 5.28 apresenta o processo de autenticação através de EAP-TLS. Inicialmente o servidor solicita ao cliente a sua identidade, o cliente responde com uma mensagem de identidade. Depois de verificar a identidade do cliente o servidor informa o cliente que terá início o processo de autenticação TLS. O cliente envia ao servidor uma mensagem com um número aleatório. Esta mensagem inclui também um conjunto de primitivas criptográficas e métodos de compressão suportados pelo cliente.

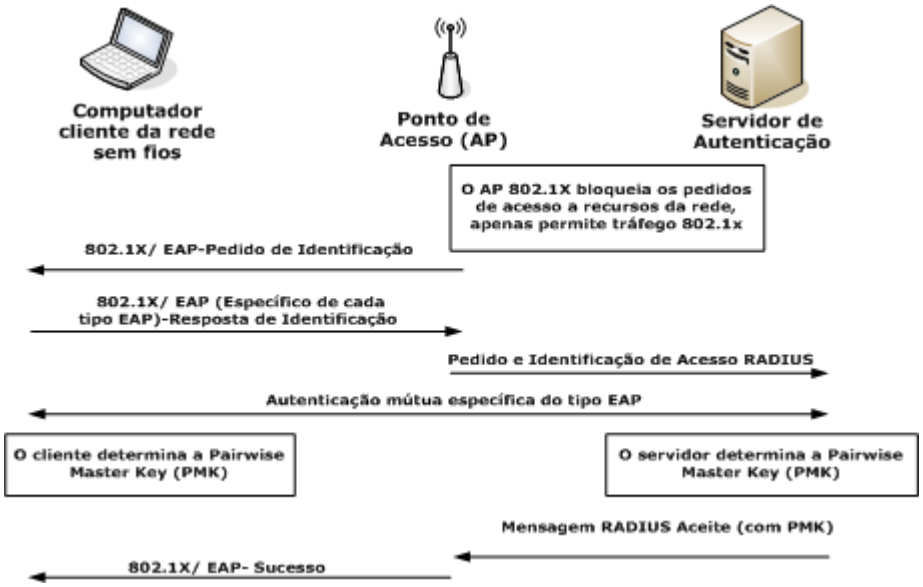


Figura 5.27 - Autenticação, descrição geral.

Inicia-se agora a fase de troca de certificados entre o servidor e o cliente. O servidor envia o seu certificado e um número aleatório ao cliente, e indica a primitiva criptográfica escolhida para cifrar os dados. Os números aleatórios gerados pelo cliente e pelo servidor serão mais tarde utilizados para gerar a chave de sessão. O certificado do servidor inclui a sua identidade e a sua chave pública. A chave pública serve para cifrar as mensagens enviadas para o servidor e para validar mensagens recebidas do servidor com a sua assinatura digital. Nesta mensagem é também efectuado o pedido do certificado do cliente, permitindo assim a sua autenticação.

Nesta fase, o cliente gera um outro número aleatório. Este número, designado por *pre-master secret*, será utilizado conjuntamente com os outros números aleatórios para gerar a chave de sessão ou MK. Como o cliente, através da chave pública da AC, verifica a validade do certificado do servidor, pode utilizar o número aleatório enviado pelo servidor para gerar a MK. O cliente envia então como resposta ao servidor, o seu certificado, o *pre-master secret* cifrado com a chave pública do servidor (*client_key_exchange*) e a indicação de que está preparado para utilizar a chave de sessão (*change_cipher_spec*).

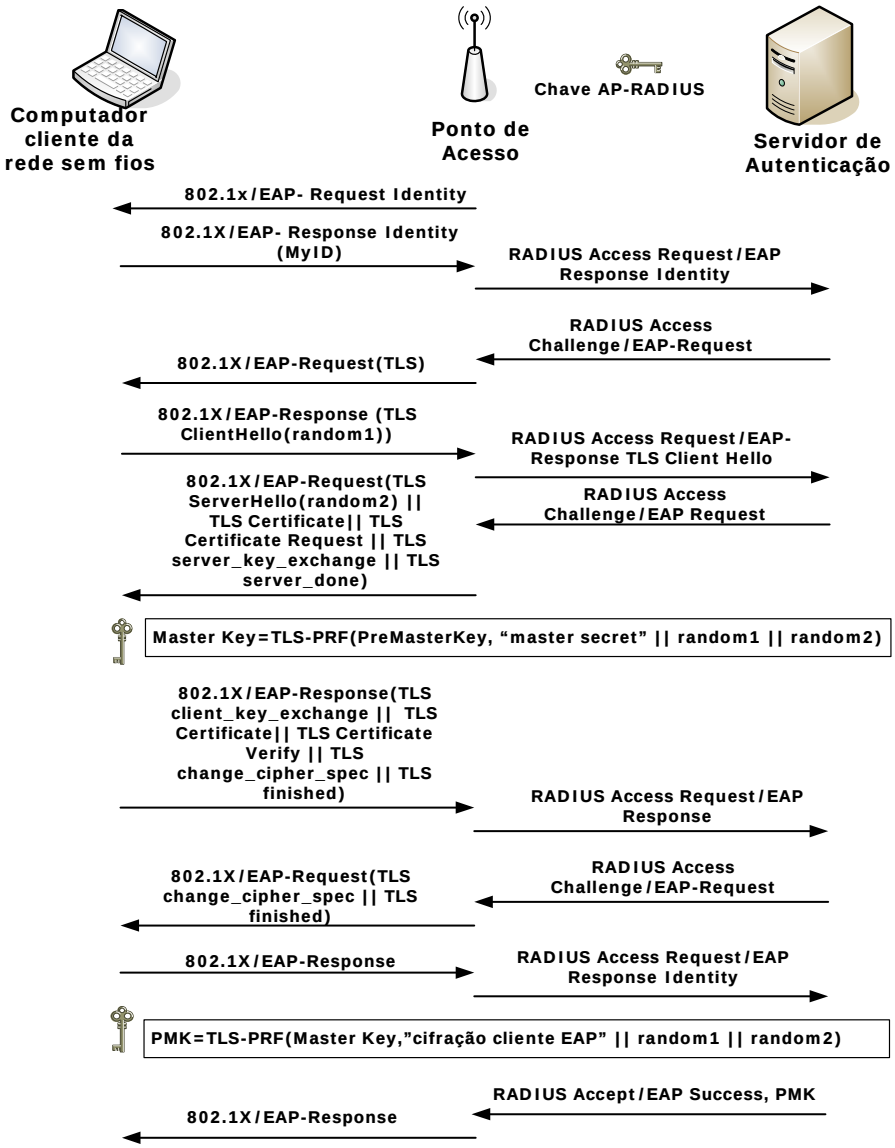


Figura 5.28 - Exemplo EAP-TLS.

O servidor utiliza a chave pública da AC para verificar a validade do certificado do cliente. Se o certificado for válido, verifica as assinaturas digitais nas mensagens anteriormente recebidas do cliente, utilizando a chave pública deste. Se as assinaturas digitais forem verificadas com sucesso, o servidor utiliza a sua chave privada para obter o *pre-master secret* e assim gerar da mesma forma a MK. A partir deste momento tanto o cliente como o

servidor conhecem a MK. O servidor indica então ao cliente, que também ele conhece a MK e que podem começar a utilizar a primitiva criptográfica para cifrar os dados (TLS *change_cipher_suite*). O servidor indica também o fim da autenticação TLS (*finished*).

Como o cliente não pretende obter mais informações do servidor, envia-lhe uma mensagem com a sua identidade. O servidor, ao receber a mensagem final do cliente, gera a chave PMK que é função da MK e envia uma mensagem de autenticação EAP bem sucedida ao cliente. O cliente gera a PMK da mesma forma que o servidor.

5.4.3.3. Fase da distribuição de chaves

Assim que o servidor obtém a PMK, deve enviá-la ao AP. A PMK será depois utilizada na fase seguinte para gerar as chaves de cifra dos dados. O envio da PMK para o AP é efectuado na mensagem RADIUS de autenticação TLS bem sucedida (última mensagem da Figura 5.28). É importante referir que o servidor envia e não copia a chave para o AP; isto evita, caso o servidor seja alvo de um ataque, a possibilidade de um atacante obter a PMK.

5.4.3.4. Fase da gestão de chaves

Concluída a fase de distribuição de chaves, inicia-se a fase de gestão das chaves. A gestão de chaves utiliza o processo conhecido como *four-way handshake* entre o cliente e o AP (Figura 5.29), para determinar a chave PTK utilizada para proteger o tráfego entre o cliente e o AP. De seguida, é efectuado o processo *group key handshake* (Figura 5.30) para determinar a chave GTK para proteger tráfego *broadcast* e *multicast* entre o cliente e o AP.

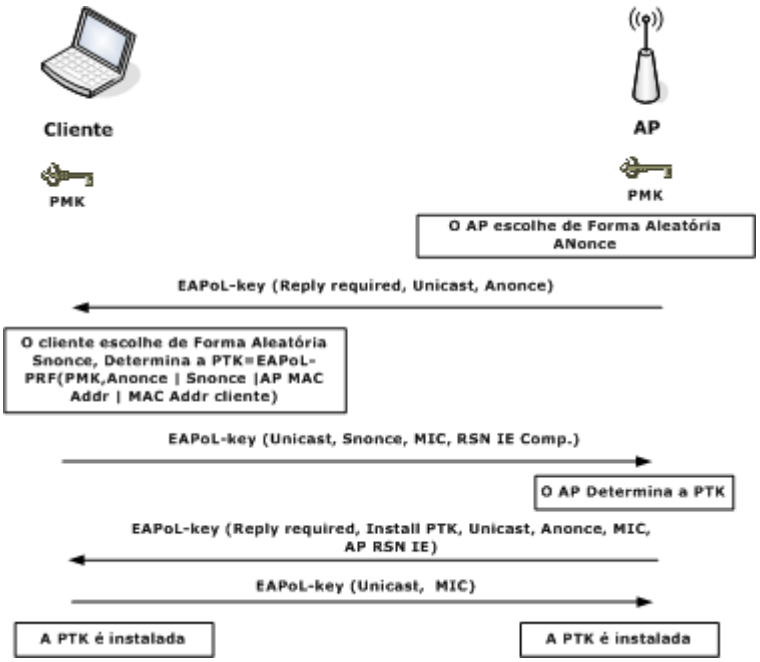


Figura 5.29 - *Four-way handshake.*

A análise da Figura 5.29 permite identificar os passos necessários à definição da PTK. Este processo utiliza apenas, para as comunicações entre o AP e o cliente, mensagens *EAPoL-key*. De uma forma resumida pode-se referir que, na fase inicial do processso *four-way handshake*, o autenticador e o cliente geram um par de *nonces*. O *nonce* do autenticador é designado por *Anonce* e o do cliente por *Snonce*. A partir dos *nonces,* da PMK obtida na fase anterior e dos endereços MAC do cliente e do AP, são geradas, aplicando uma função geradora de pseudoaleatórios, as chaves de cifra temporárias (PTK). Para cada uma das partes verificar que a outra possui a PTK, é incluído um MIC nas mensagens. O MIC permite também informar os intervenientes se a comunicação está a ser alvo de um ataque MITM.

O processo de criação da GTK, descrito na Figura 5.30, é muito semelhante ao processo utilizado para se determinar a PTK. Neste caso, apenas o autenticador precisa de determinar a GTK. Para obter a GTK o AP escolhe um número aleatório (*Gnonce*) de 256 *bits* que se tornará a GMK. A GMK é combinada com o endereço MAC do AP através de uma função geradora de pseudoaleatórios, obtendo-se a GTK. A GTK constitui duas novas chaves de 128 *bits* cada, a GEK e a GIK. Todo o tráfego de *broadcast/multicast* será cifrado utilizando a GEK. Como a GEK é utilizada para proteger tráfego *broadcast*, o AP deve enviar esta chave a todos os clientes que serão destinatários de uma mensagem de *broadcast/multicast*. Para ser possível o envio de forma segura da GEK do AP para os clientes, o AP cifra a GTK com a chave KEK. Como anteriormente referido, a KEK é uma das chaves que constituem a PTK. Os clientes, ao receberem a mensagem com a GTK, descodificam-na e obtêm a GEK. Neste ponto é possível enviar tráfego *broadcast* de forma segura. Para concluir o processo, o cliente confirma a recepção da GEK, enviando uma mensagem ao AP com uma MIC obtida através da GIK.

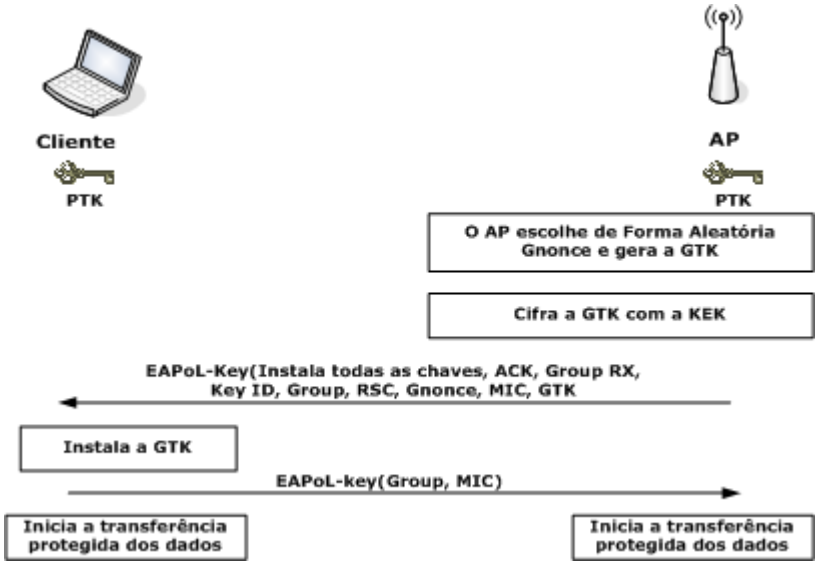


Figura 5.30 - *Group-key handshake.*

5.4.3.5. Transferência dos dados

O WPA define como protocolo para protecção da transferência de dados o TKIP descrito na secção 5.4.1.

5.4.4. Formato das mensagens

A mensagem protegida pelo WPA (Figura 5.31) é uma trama TKIP ao qual se acrescenta o cabeçalho 802.11 e o cabeçalho IP. O cabeçalho 802.11 é constituído pelo endereço MAC de origem e de destino, e o cabeçalho IP é formado pelos endereços IP de origem e destino. Os únicos campos a serem cifrados são os dados, o MIC e o ICV.

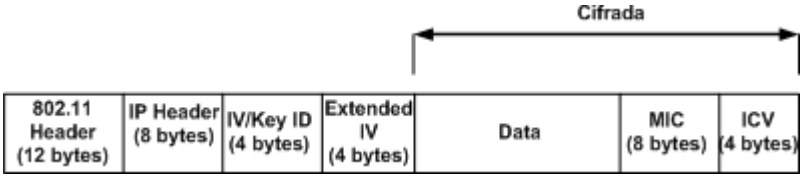


Figura 5.31 - Formato de uma mensagem protegida pelo WPA.

5.4.5. Implementação do WPA.

Nesta secção apresenta-se a implementação de um sistema WPA para protecção de uma rede sem fios. Apresentam-se também as vantagens e desvantagens ao nível de eficiência e complexidade. A Figura 5.32 ilustra os componentes constituintes de um sistema WPA. A implementação WPA passa por desenvolver uma infra-estrutura 802.1X. A concepção desta infra-estrutura requer que os seus elementos sejam actualizados para suportar o WPA.

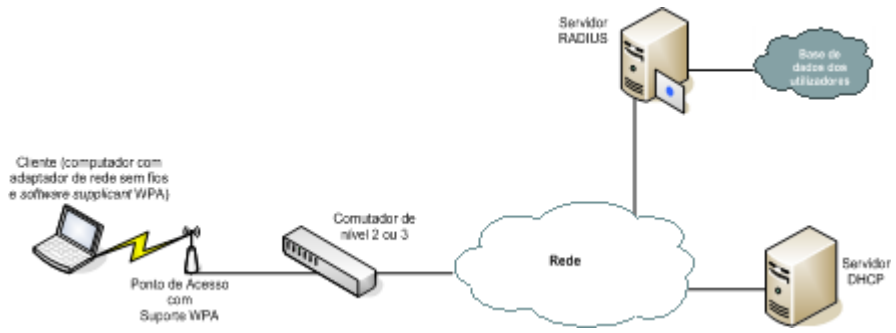


Figura 5.32 - WPA.

Os clientes necessitam de permitir a autenticação 802.1X e serem compatíveis com o WPA. Estes elementos devem também suportar o TKIP e, tal como no 802.1X, têm de ser actualizados com o *supplicant* EAP.

Os APs necessitam do suporte de WPA ou suporte de activação por actualização de *firmware*; o 802.1X e o TKIP são também funcionalidades necessárias nos APs. Os APs ligam-se à rede com fios através de comutadores de nível 2 ou 3.

Para garantir toda a autenticação de utilizadores, o servidor de autenticação tem de incluir um servidor RADIUS. Finalmente, é também implementado um servidor de DHCP para atribuição dos endereços IP.

Todo o processo de funcionamento do WPA é muito semelhante ao da autenticação 802.1X, sendo a arquitectura exactamente igual à arquitectura 802.1X/EAP. A única diferença será a necessidade de utilizar APs e clientes da rede sem fios compatíveis com o WPA. O funcionamento do sistema implementado é semelhante ao apresentado na secção anterior.

5.4.5.1. Vantagens

Este tipo de implementação elimina o perigo dos *Sniffers* de pacotes devido à codificação da informação, e elimina o perigo de acesso não autorizado à rede pois apenas os utilizadores autenticados é que lhe têm acesso. Além disso, a natureza da autenticação mútua da autenticação EAP-TLS e da autenticação mútua entre o AP e o cliente, através da verificação do conhecimento da PMK, combinadas com o MIC, permitem prevenir que intrusos se coloquem no caminho das comunicações, eliminando o perigo de ataques MITM.

Uma outra vantagem consiste no facto de, se um intruso não se conseguir autenticar na rede sem fios, não poderá realizar ARP *Spoofing*. Ataques de repetição e de modificação da informação também não são possíveis, pois o ICV protege contra este tipo de ataques.

A utilização de chaves dinâmicas permite que a confidencialidade das comunicações seja efectivamente conseguida. Finalmente, se não for possível a um intruso realizar autenticação, não pode realizar descoberta da rede, ficando limitado aos protocolos que não são filtrados.

Relativamente à implementação 802.1X, O WPA apresenta como vantagens a eliminação dos ataques MITM, ataques de repetição e de modificação da informação.

5.4.5.2. Desvantagens

Como desvantagem desta solução encontra-se a possibilidade de realização de ataques de DoS, bastando para tal efectuar dois pedidos de autenticação não autorizados ao AP no mesmo segundo. Depois de receber o segundo pedido de autenticação não-autorizado, o AP rejeita todos os pedidos de acesso à rede, incluindo pedidos de acesso devidamente autorizados.

Outra das desvantagens do WPA é a utilização, tal como no WEP, da cifra de fluxos RC4 para cifra dos dados. Tal como anteriormente referido, esta cifra apresenta falhas que poderão ser utilizadas por um intruso para atacar uma implementação WPA.

Comparativamente com o 802.1X, esta solução apresenta como desvantagem uma maior carga de processamento dos sistemas, já que utiliza chaves com tamanho superior. Outra da desvantagem é a necessidade, em certas situações, de se ter que adquirir novos adaptadores de rede sem fios e APs compatíveis com o WPA.

5.4.6. WPA – *Pre Shared Key* (PSK)

A *Wi-Fi Alliance* definiu, para situações em que não é possível a utilização de um servidor de autenticação, um modo de funcionamento do WPA denominado de WPA-PSK. Neste modo de funcionamento, a chave PMK deve ser previamente configurada no AP e no cliente. A Figura 5.33 apresenta a implementação de um sistema WPA-PSK. Como ilustrado na figura, para a implementação WPA-PSK é apenas necessário um AP com suporte WPA e clientes da rede sem fios com o *software* cliente WPA (*supplicant* WPA). Como neste modelo de operação não é efectuada a autenticação dos utilizadores, é apenas necessário efectuar o processo *four-way-handsha*ke para obtenção da chave PTK. Depois de obtida a PTK é calculada a chave GTK da mesma forma que para o sistema WPA.

Figura 5.33 - Sistema WPA-PSK.

Este modo de funcionamento, para além das desvantagens associadas a um sistema WPA, introduz a possibilidade de um intruso efectuar ataques de dicionário à PMK configurada no cliente, permitindo-lhe aceder à rede de forma não autorizada.

As principais vantagens deste sistema são a simplicidade de implementação e de configuração, não sendo necessário implementar um servidor de autenticação.

5.5. RSN/ IEEE802.11i

O grupo IEEE802.11i desenvolveu uma nova norma denominada de *Robust Security Network* (RSN)[5] [117]. A norma de segurança 802.11i inclui as seguintes funcionalidades: cifra de dados na rede sem fios através do AES, autenticação dos utilizadores através do 802.1X, e gestão de chaves de cifra através do TKIP (o WPA não define como algoritmo de cifra o AES, sendo essa a principal diferença entre as normas). O grupo recomenda também uma série de correcções ao WEP em adaptadores 802.11b que, devido a limitações na sua

[5] O RSN também é denominado WPA2

concepção, não suportem ou não possam ser actualizados para suportar AES; estas correcções concentram-se na actualização dos dispositivos para suporte de TKIP.

De forma a garantir um conjunto de modos de cifra dos dados na mesma rede sem fios, a especificação 802.11i requer que os dispositivos anunciem as suas capacidades de cifra aos APs nos pedidos de associação dos clientes. O AP e o cliente estabelecem de seguida a chave apropriada para a cifra dos dados com base nas suas capacidades mútuas e em qualquer das políticas configuradas para a rede (por exemplo:"permitir apenas a associação de clientes AES").

A autenticação 802.1X garante a regeneração de chaves durante uma sessão (dado o facto de não existirem técnicas conhecidas para descodificar a cifra AES, não é evidente ainda a necessidade de regeneração de chaves numa sessão de forma regular). A regeneração automática das chaves de uma sessão elimina a necessidade de configurar manualmente cada cliente com uma chave, e a necessidade de que cada AP tenha que ter de forma actualizada a chave única de cada utilizador.

A utilização do AES para cifra dos dados exige a utilização de equipamentos especialmente concebidos para esse fim. Para ser possível a utilização do 802.11i em equipamentos existentes (esta regra não se aplica a equipamentos WEP), a norma define como protocolo de gestão e cifra dos dados o TKIP.

Nas secções seguintes apresenta-se o princípio de funcionamento do IEEE802.11i e dos seus módulos associados, a descrição da implementação de uma rede com IEEE802.11i e suas vantagens e desvantagens.

5.5.1. Modelo Genérico de Operação do RSN

O IEEE802.11i funciona de acordo com um modelo de políticas [126]. O IEEE802.11i define no seu modelo de operação os seguintes elementos: ponto de decisão da política (*Policy Decision Point* - PDP) que identifica o dispositivo lógico que realiza as decisões ao nível da política de segurança do sistema; ponto de reforço da política (*Policy Enforcement Point* - PEP) que identifica o dispositivo lógico de reforço das decisões ao nível da política de segurança. Existe ainda um *token* de decisão da sessão (*Session Decision Token* - SDT) que identifica a estrutura de dados que representa a decisão ao nível da política de segurança, e um *token* de reforço da sessão (*Session Enforcement Token* - SET) que identifica a estrutura de dados utilizada para reforçar a política de decisão. O modelo de operação do IEEE802.11i é ilustrado na Figura 5.34.

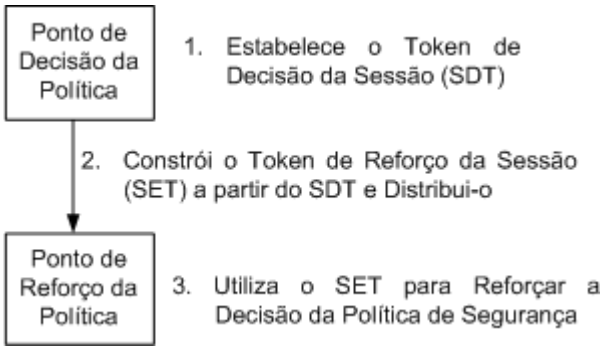


Figura 5.34 - Modelo de Operação do IEEE802.11i/ RSN.

O modelo anteriormente descrito aplica-se à norma 802.11i da seguinte forma: o cliente e o servidor de autenticação representam os dois pontos de decisão de políticas. A decisão de política permite ou não ao cliente ter acesso à rede 802.11, caso a autenticação seja bem ou mal sucedida, respectivamente.

Na implementação 802.11i o *token* de decisão de políticas é denominado *Master Key* (MK); apenas o servidor de autenticação e o cliente podem possuir a MK, e efectuar decisões apenas quando se obtém a MK. Os dois pontos de reforço da política são também o cliente e o AP. O *token* de reforço da política é a PMK; esta controla o acesso do cliente e do AP ao canal 802.11 durante a sessão. Como foi referido anteriormente, a posse de uma PMK determina a autorização de aceder ao canal 802.11 durante a sessão.

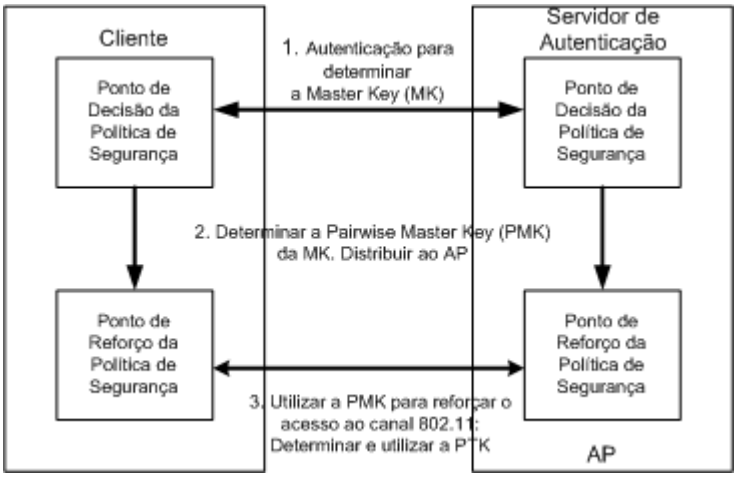


Figura 5.35 - 802.11i.

A Figura 5.35 apresenta a forma como a implementação 802.11i define os diversos pontos de decisão das políticas.

5.5.2. Arquitectura de autenticação e gestão de chaves

A arquitectura de autenticação do IEEE802.11i é em tudo semelhante à da WPA (secção 5.4.2). Também é recomendado como protocolo de autenticação entre o cliente e o AP o EAP-TLS, sendo o protocolo de autenticação entre o AP e o servidor, o RADIUS.

Na sua arquitectura de gestão de chaves, o IEEE802.11i define, tal como o WPA, uma hierarquia de chaves. Para as situações em que é utilizado o TKIP para proteger os dados, a hierarquia de chaves é em tudo semelhante ao da WPA (5.4.2).

Para os equipamentos que possibilitam a utilização do AES, a hierarquia de chaves apresenta pequenas diferenças relativamente à da TKIP. Nesta situação, a PTK é composta apenas por três chaves (384 *bits*) e a GTK é composta apenas por uma chave (128 *bits*). No modo AES, a chave PTK é composta pela chave KEK (128 *bits*), KIK (128 *bits*) e por uma chave (128 *bits*) que permite a cifra e a integridade dos dados. A GTK é apenas uma chave que permite a cifra e a integridade das comunicações de *broadcast* e de *multicast*. Com o AES são apenas necessários 512 *bits* para armazenar todas as chaves; o TKIP necessita de 768 *bits*. Estas diferenças têm implicações ao nível do *overhead* das mensagens.

5.5.3. Fases Operacionais do 802.11i

O 802.11i define as mesmas fases que o WPA, com os mesmos objectivos

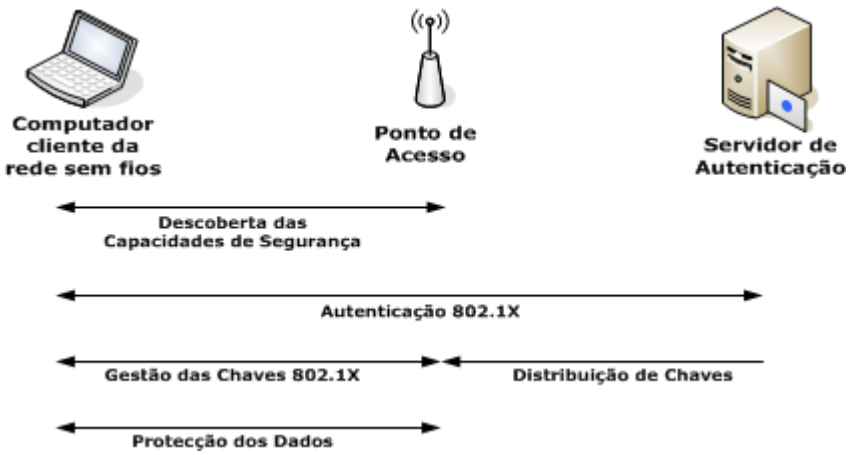


Figura 5.36 - Fases Operacionais do 802.11i.

A principal diferença reside na fase da descoberta das capacidades de segurança, que tem como objectivo determinar possíveis pares com quem estabelecer comunicações e anunciar as capacidades de segurança da rede aos clientes.

A Figura 5.37 ilustra a fase de descoberta entre o AP e o cliente. Nesta fase, o cliente efectua um pedido de associação aos APs disponíveis na rede, da mesma forma que no WPA. As capacidades de segurança, não negociáveis, são enviadas numa mensagem de *broadcast* denominada de RSN *Information Element* (IE). A mensagem RSN IE informa o cliente do protocolo de autenticação (802.1X), protocolo de cifra dos dados *unicast* (CCMP) e protocolo de cifra dos dados *broadcast* (CCMP) suportados pelo AP. O cliente verifica as suas capacidades de segurança e, caso suporte as capacidades anunciadas pelo AP, responde da mesma forma.

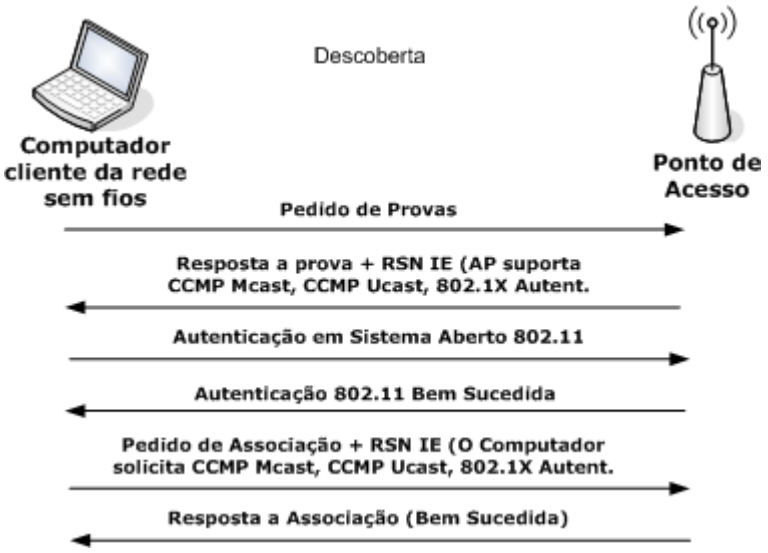


Figura 5.37 - Fase da Descoberta.

As fases de autenticação 802.1X, distribuição de chaves e de gestão de chaves são semelhantes às do WPA (ver secção 5.4.3).

O 802.11i define três protocolos para protecção da transferência de dados: o *Counter Mode with Cipher Block Chaining Message Authentication Code* (CCMP), o *Wireless Robust Authenticated Protocol* (WRAP) [130] e o TKIP.

	WEP	TKIP	CCMP
Cifra	RC4	RC4	AES
Tamanho Chave	40 ou 104 *bits*	128 *bits* de cifra, 64 bits de autenticação	128 *bits*
Vida da Chave	IV 24 *bits*, WRAP	IV 48 *bits*,	IV 48 *bits*
Chave do Pacote	Concatenação	Função de Mistura	Não é Necessário
Integridade dos Dados	CRC-32	MIC	CCM
Integridade do Cabeçalho	Não	MIC	CCM
Repetições	Não	Utilização do IV	Utilização do IV
Gestão das Chaves	Não	EAP	EAP

Tabela 5-1 - Comparação WEP, TKIP e CCMP.

O CCMP será o protocolo *de facto* do IEEE802.11i/RSN; é considerado, no entanto, uma solução a longo prazo, sendo baseado no protocolo AES no modo CCM. Este protocolo encontra-se descrito na secção 3.1.1.3.2. O CCMP é um protocolo inteiramente novo com poucas concessões ao WEP e que permite proteger fragmentos de tramas 802.2 [127]. Este protocolo deriva das aprendizagens realizadas com os protocolos 802.10 [128] e IPsec, estando alicerçado na correcta utilização de primitivas criptográficas fortes, o que garante protecção a todos os ataques conhecidos. A principal desvantagem do CCMP é a necessidade de se adquirir novo *hardware* para os dispositivos. O CCMP utiliza uma cifra AES de 128 *bits* para protecção dos dados; um IV de 48 *bits* garante a não reutilização das chaves. Este protocolo não exige a utilização de chaves por pacote. Outra característica importante deste protocolo é a utilização do EAP para a geração dinâmica de chaves por sessão.

O WRAP, baseado no AES no modo *Offset CodeBook* (OCB) [129], é a proposta original para o protocolo de cifra do 802.11i. Este foi, no entanto, substituído pelo CCMP por questões de direitos de propriedade intelectual.

O TKIP foi desenvolvido como camada de protecção suplementar ao WEP, podendo ser implementado em *software* e reutilizando o *hardware* WEP já existente. A Tabela 5-1 apresenta de forma resumida as principais diferenças entre o WEP, TKIP e o CCMP.

5.5.4. Formato das mensagens 802.11i

O formato de uma mensagem 802.11i (Figura 5.38) é muito semelhante ao formato de uma mensagem WPA. Existem apenas duas diferenças: a integridade dos dados é apenas função do MIC, não sendo por isso utilizada a função linear CRC para obtenção do ICV; o 802.11i pode utilizar, caso seja suportado pelos equipamentos, o processo de cifra baseado no protocolo criptográfico AES.

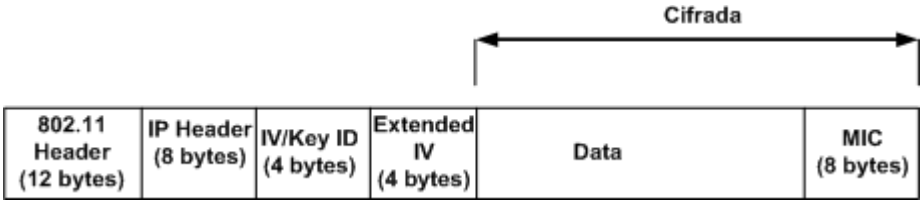


Figura 5.38 - Formato de uma mensagem IEEE802.11i.

5.5.5. Implementação 802.11i/RSN

Nesta secção apresenta-se a implementação de um sistema 802.11i para protecção de uma rede sem fios. Apresenta-se também uma descrição do funcionamento da solução, das suas vantagens e desvantagens ao nível de eficiência e complexidade. A Figura 5.39 ilustra os componentes constituintes de um sistema 802.11i. A implementação 802.11i contém, tal

como o WPA, uma infra-estrutura 802.1X. A concepção desta infra-estrutura requer que os seus elementos sejam actualizados para suportar o IEEE802.11i.

Os clientes necessitam de permitir a autenticação 802.1X e de ser compatíveis com o 802.11i, incluindo já esta opção ou podendo o seu *firmware/software* ser actualizável. Estes elementos devem também suportar o TKIP e o AES e, tal como no 802.1X, têm de ser actualizados com o *supplicant* EAP.

Os APs necessitam do suporte de 802.11i; apenas os APs mais recentes o permitem, não sendo possível a activação do 802.11i por actualização de *firmware* em APs mais antigos. O 802.1X, TKIP e o AES são também funcionalidades necessárias nos APs. Os APs ligam-se à rede corporativa através de comutadores de nível 2 ou 3.

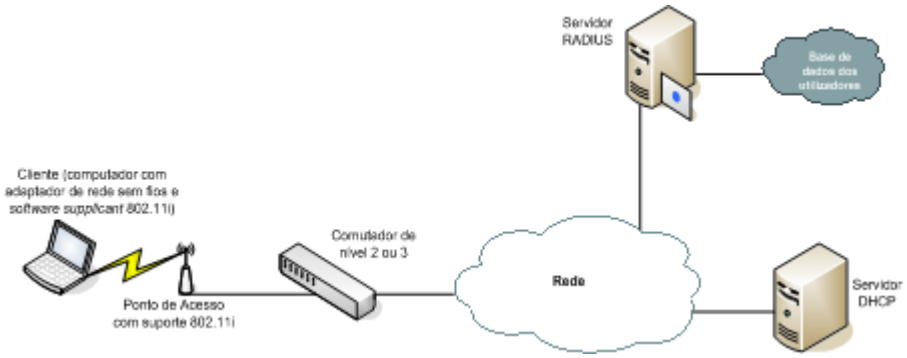


Figura 5.39 - RSN/IEEE802.11i.

Para garantir toda a autenticação de utilizadores para os clientes, o servidor de autenticação tem de incluir um servidor RADIUS. Finalmente, é também implementado um servidor de DHCP para atribuição dos endereços IP.

O processo de funcionamento do 802.11i é semelhante ao do WPA (ver secção 5.4.5). A única diferença centra-se na necessidade de utilizar APs e clientes compatíveis com o 802.11i. Como já anteriormente referido, o 802.11i aumenta as capacidades de segurança utilizando como protocolo de cifra dos dados, o AES no modo CCM. Tal como o WPA, introduz o conceito de chaves de sessão dinâmicas, com a introdução de chaves por utilizador, por sessão e por pacote.

5.5.5.1. Vantagens

Este tipo de implementação elimina o perigo dos *Sniffers* de pacotes devido à codificação da informação, e elimina o perigo de acesso não autorizado à rede.

O 802.11i permite definir políticas de associação entre os clientes e os APs. Pode-se configurar nos APs uma política de segurança que apenas permita a associação de clientes AES.

A utilização do algoritmo AES para cifrar os dados garante a confidencialidade das comunicações, pois ainda não se conhecem técnicas para decifrar de forma não autorizada este algoritmo. Este algoritmo, dado ser um protocolo muito seguro, permite a utilização de chaves com tamanho menor.

Relativamente à implementação WPA, o 802.11i, permite uma maior flexibilidade na implementação de uma rede sem fios protegida, permitindo definir políticas de associação entre os clientes e os APs. O 802.11i está também mais centrado no algoritmo de cifra AES, ao contrário do WPA que está mais centrado no RC4. O AES permite a utilização de chaves de cifra mais pequenas, conseguindo consideráveis ganhos ao nível do desempenho da rede.

5.5.5.2. Desvantagens

Tal como no WPA, é possível realizar ataques do tipo DoS. O envio de duas mensagens não autorizadas de associação ao AP, no mesmo segundo, activa o sistema de protecção do AP, fazendo com que este não permita que mais nenhum cliente se associe.

No entanto, a principal desvantagem do 802.11i é a necessidade de se adquirir equipamentos novos (nomeadamente APs e adaptadores de redes sem fios) que permitam a utilização do AES. O AES exige maior carga de processamento dos equipamentos, não sendo compatível com equipamentos menos actuais (inclusivé muitos dos equipamentos actuais não o suportam).

5.6. Conclusão

A segurança constitui um dos aspectos mais importantes em qualquer tipo de transmissão de dados. No entanto, nas redes sem fios, esse é um aspecto essencial; o tipo de meio de transmissão que utilizam é um meio partilhado, em que a transmissão dos dados é feita por difusão. Os mecanismos de segurança das redes sem fios devem ser robustos de forma a se conseguirem níveis de segurança semelhantes aos das redes com fios. A segurança deve ser obtida através de dois conceitos: autenticação (controlo dos acessos) e criptografia (protecção dos dados). Este capítulo apresentou uma descrição dos sistemas de segurança existentes para redes sem fios. De seguida, é efectuada uma reflexão crítica de cada um deles.

Nos últimos anos, a norma IEEE802.11 tem emergido como a norma dominante das redes sem fios. A primeira tentativa do IEEE802.11 de introduzir mecanismos de segurança nas redes sem fios foi a definição do protocolo WEP. Este protocolo cedo se mostrou um método ineficaz. O WEP utiliza como algoritmo de cifra o RC4 e um vector de inicialização de 24 *bits*. O RC4 é um algoritmo de chave simétrica; no WEP essa chave é uma chave estática, devendo ser configurada manualmente no AP e no cliente. Várias ferramentas de

software permitem recuperar a chave, permitindo assim a um intruso atacar uma rede protegida com o WEP. Outro dos pontos críticos do WEP é o tamanho do seu IV: numa mesma sessão é possível que o mesmo IV seja utilizado mais do que uma vez, possibilitando a realização de ataques de repetição por parte de um intruso. Outro dos problemas do WEP é utilizar mecanismos de autenticação muito fracos.

Para reduzir ou até mesmo eliminar alguns dos defeitos do WEP, utilizou-se como mecanismo de autenticação o 802.1X. O 802.1X é um mecanismo de autenticação baseado em portas, que utiliza métodos baseados em sistemas de chave pública e certificados, nomeadamente o EAP-TLS. Esses mecanismos permitem a autenticação mútua entre o cliente e o servidor de autenticação. No entanto, o 802.1X define apenas um método robusto para controlo de acessos, não definindo métodos robustos para a protecção dos dados nem para a geração de chaves por pacotes.

De modo a ser possível o desenvolvimento de um mecanismo robusto de segurança para as redes sem fios, o IEEE80.11 criou o grupo de trabalho "i". O IEEE802.11i apresenta um conjunto de aperfeiçoamentos relativamente ao WEP, nomeadamente a utilização de chaves dinâmicas para cifrar os dados, a utilização de forma nativa de autenticação 802.1X e a utilização do algoritmo AES para protecção dos dados. A utilização do AES exige que seja utilizado *hardware* totalmente novo, devido à maior exigência ao nível de processamento.

Como a maioria das redes sem fios instaladas utiliza *hardware* incompatível com o AES, houve a necessidade de introduzir um mecanismo de segurança mais robusto que o WEP, mas compatível com esse *hardware*. A *Wi-Fi Alliance* definiu o protocolo WPA. Este protocolo é um sub-sistema do IEEE802.11i. O WPA define também como método de autenticação o 802.1X, chaves geradas de forma dinâmica por sessão, sendo a principal diferença para com o IEEE802.11i utilizar para algoritmo de cifra o RC4. Estas características possibilitam que apenas seja necessária uma pequena actualização do *firmware* em sistemas compatíveis com o WEP para ser possível implementar sistemas protegidos com o WPA.

O WPA definiu dois modos de operação: o WPA-PSK (também conhecido por WPA Pessoal) e o WPA-EAP (também conhecido por WPA Empresarial). No primeiro modo não é necessário a utilização do 802.1X para a autenticação mútua e para a geração das chaves. Neste modo, as chaves de sessão são geradas utilizando uma palavra-chave configurada no cliente e no AP. O WPA-PSK utiliza também geração dinâmica de chaves por cada sessão que o cliente estabelece com o AP. Este é um modo de operação WPA muito utilizado em situações de utilização doméstica ou em pequenas empresas. O WPA-EAP é utilizado em ambientes empresariais onde a segurança é um factor essencial para a implementação de

redes. Este modo utiliza o mecanismo 802.1X para controlo de acesso à rede, e o TKIP para protecção dos dados.

A aplicação de mecanismos de segurança em redes com fios pode ser também uma das soluções na introdução de mecanismos de segurança nas redes sem fios. Um desses exemplos é o IPsec. O IPsec é um mecanismo extremamente robusto e versátil para ser implementado em qualquer sistema. O IPsec é na realidade um conjunto de protocolos que permitem criar mecanismos poderosos de autenticação mútua e de protecção dos dados entre um cliente e um AP. Outra das vantagens do IPsec é que é independente do *hardware*, não sendo necessário efectuar actualizações de *firmware* ou substituição de *hardware*. O IPsec é um mecanismo com muita aceitação, tendo-se mostrado robusto e eficaz na protecção das comunicações nas mais variadas situações. No entanto, o IPsec, devido ao maior *overhead* introduzido nas mensagens, exige maior capacidade de processamento nos equipamentos. Esta característica é de extrema importância, pois pode diminuir de forma significativa o desempenho da rede. Outro aspecto a salientar no IPsec, é que apenas efectua protecção na camada de rede, não autenticando os dispositivos que intervêm na comunicação. Isto é um problema, pois é possível mediante a utilização de APs não autorizados realizar ataques MITM. Um sistema IPsec exige também maiores necessidades ao nível da administração da rede, o que pode induzir em falhas na gestão da mesma.

Capítulo 6

Implementação das soluções de segurança

Neste capítulo descreve-se a implementação de algumas das soluções de segurança em redes sem fios referidas no capítulo 5. As implementações apresentadas são baseadas no protocolo *Internet Protocol Security* (IPsec) e no protocolo *Wi-fi Protected Access* (WPA). As implementações baseadas na norma IEEE802.11i não são apresentadas, devido a não estarem ainda disponíveis equipamentos e sistemas baseados nessa norma (a norma IEEE802.11i só foi ratificada em Junho de 2004).

Todas as implementações aqui descritas são baseadas, sempre que possível, em ferramentas de *software* e em sistemas operativos em código aberto. Utilizam-se ferramentas e *software* comerciais sempre que não exista nenhuma versão em código aberto. Na descrição das diversas implementações indica-se o processo de instalação, configuração e utilização de todas as ferramentas utilizadas.

O principal objectivo deste capítulo é o de fornecer as directrizes necessárias à configuração e utilização dos principais componentes dos mecanismos de segurança utilizados. Outro dos objectivos é mostrar que é possível utilizar ferramentas gratuitas para implementar esses mecanismos, reduzindo-se assim de forma significativa o custo final das soluções a implementar.

A implementação IPsec é utilizada para proteger o acesso dos clientes da rede sem fios a uma rede com fios – vulgo rede corporativa. Para tal, é necessário configurar uma *gateway Virtual Private Network* (VPN) de forma a permitir o acesso dos clientes VPN à rede protegida. Esta implementação é baseada na implementação descrita na secção 5.2.7, em que a autenticação utiliza certificados digitais (o servidor RADIUS não é utilizado).

Uma das implementações baseadas no protocolo WPA utiliza o processo WPA-*Extensible Authentication Protocol* (EAP) descrito na secção 5.4, em que o mecanismo EAP utilizado é o *Transport Layer Security* (TLS). O servidor de autenticação é baseado no RADIUS. Outra característica desta implementação é o facto de ser possível o seu desenvolvimento sem ser necessário substituir o *hardware* disponível. Para ser possível realizar esta implementação utiliza-se um AP (*Access Point*) comercial devido a não existir, quando se iniciou o processo de implementação da solução, um AP em código aberto que permitisse a utilização do protocolo WPA.

A outra implementação baseada no protocolo WPA utiliza o mecanismo WPA-*Pre Shared Key* (PSK), descrito na secção 5.4.6. Esta implementação apresenta um menor nível de segurança. Utiliza-se nesta implementação, o AP utilizado na implementação WPA-EAP. Nesta implementação não é necessário um servidor de autenticação, facto que diminui em muito a complexidade da mesma. As funções de servidor de autenticação são da responsabilidade do AP, o que torna o processo 802.1X muito mais simples.

A solução baseada no protocolo 802.1X-EAP (secção 5.3) não foi implementada pelo facto de esse processo apenas ser considerado um mecanismo de autenticação (apenas acrescenta o serviço de autenticação), e não um conjunto de mecanismos que realmente introduzem segurança numa rede sem fios.

Este capítulo começa por apresentar na secção 6.1 os procedimentos necessários à implementação de um sistema IPsec, nomeadamente a implementação de uma VPN IPsec. São referidas nessa secção as ferramentas necessárias e os procedimentos de configuração necessários para se implementar a solução referida. Na secção 6.2 discutem-se todas as ferramentas, sistemas e procedimentos de configuração necessários à implementação de um sistema WPA-EAP; na secção 6.3 são descritos os procedimentos necessários à implementação de um sistema WPA-PSK. Este capítulo termina na secção 6.4 com uma breve conclusão.

6.1. Cenário 1 – VPN IPsec

O cenário de suporte de segurança em redes sem fios através da criação de VPNs IPsec encontra-se apresentado na Figura 6.1. Como se referiu na secção 5.2.7, cada terminal ligado à rede sem fios tem de conter *software* de cliente VPN para poder criar os túneis cifrados desde o terminal à *gateway*.

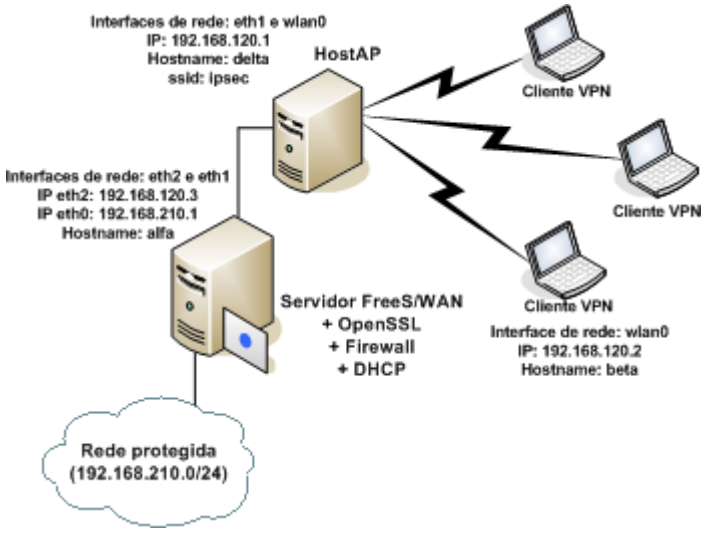


Figura 6.1 - Cenário experimental VPN.

O AP é um PC com placa de rede sem fios com o software *Hostap* [36]. A *gateway* contém suporte IPsec com *Internet Key Exchange* (IKE), suporte de autenticação através de um sistema de criptografia de chave pública e certificados, uma *firewall* para efectuar a filtragem dos protocolos autorizados e um servidor DHCP.

Os elementos constituintes deste cenário e respectiva configuração ao nível da rede e dos elementos de *software* que a constituem, encontram-se apresentados na Figura 6.1. Note-se que o AP apenas efectua *bridging* entre a rede sem fios e a rede com fios, não tendo activado nenhum mecanismo de protecção.

Os elementos constituintes da rede, suas características e *software* suportado encontram-se apresentados na Tabela 6-1, Tabela 6-2 e Tabela 6-3.

AP

Designação	Características	Código Aberto
Computador	*Celeron* 433 MHz 256 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Placa de rede	Placa PCI 10/100, RJ45	
Placa de rede sem fios	U.S.*Robotics* 202410, 802.11b	
Software	*HostAP* a funcionar em modo *master*	Sim
Software de bridging	*Bridge utils*	Sim

Tabela 6-1 - AP num cenário VPN IPsec.

***Gateway* VPN**

Designação	Características	Código Aberto
Computador	*Pentium* II 400 MHz 256 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Placa de rede	Placa PCI 10/100, RJ45	
Software VPN	*FreeS/Wan*	Sim
Software para criação de certificados e de um sistema de chave pública	*OpenSSL*	Sim
Software Firewall	*Iptables*	Sim
Software DHCP	ISC DHCP	Sim

Tabela 6-2 - *Gateway* VPN num cenário VPN IPsec.

Cliente VPN

Designação	Características	Código Aberto
Computador	*Celeron* 433 MHz 128 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Placa de rede sem fios	*Linksys* WMP54g-EU, 802.11b e g, WPA, 802.1X	
Software cliente VPN	*FreeS/Wan*	Sim
Software cliente DHCP	ISC DHCP	Sim
Software driver da placa de rede sem fios	*Ndiswrapper*	Sim

Tabela 6-3 - Cliente VPN num cenário VPN IPsec.

As secções seguintes descrevem a implementação dos diferentes componentes do cenário a ser implementado.

6.1.1. *FreeS/WAN*

A ferramenta *FreeS/Wan* [23] é uma ferramenta gratuita que implementa a família de protocolos IPsec, com a finalidade de proteger a comunicação dos dados contra ataques de personificação, integridade, autenticação e não-repúdio. Esta ferramenta implementa, de forma nativa, os algoritmos de chaves assimétrica *Rivest Shamir Adleman* (RSA) e simétrica *Triple Data Encryption Standard* (3DES), além das funções de *hash Message Digest* 5 (MD5) e *Secure Hash* (SHA-1). Para permitir a utilização do *FreeS/WAN* em ambientes diferentes foram desenvolvidos diversos *patches* que incorporam funcionalidades extra ao programa original: suporte a certificados digitais X.509, adição de algoritmos de criptografia de chave simétrica como o *Advanced Encryption Standard* (AES), *Blowfish* [114] e *Serpent* [115], entre outras funcionalidades. O *Blowfish* é uma cifra de blocos de 64 *bits* com uma chave de tamanho variável. Este algoritmo foi desenvolvido por *Bruce Schneier* e consiste em duas partes: expansão de chaves e cifra dos dados. A expansão das chaves converte uma chave de 448 *bits* em vários vectores de sub-chaves, no total 4168 *bits*. A cifra dos dados consiste numa função iterada 16 vezes. Este algoritmo foi concebido para ser implementado em super-processadores.

O *Serpent* é também uma cifra de blocos, mas com tamanhos de 128 ou 256 *bits.* Este algoritmo pretende ser concorrente do AES. Embora de implementação muito eficiente, este

algoritmo apresenta uma estrutura muito conservadora. Na sua estrutura, e tal como o 3DES, utiliza *S-boxes*. No entanto, apresenta uma deslocação de *bits* mais eficiente. Dadas as potencialidades do AES, este algoritmo nunca teve grande aceitação.

O *FreeS/WAN* contém dois componentes fundamentais: o *Kernel IPsec Support* (KLIPS) e o *Pluto*. O *KLIPS* é a componente que permite a interacção necessária do IPsec com o *Kernel* do *Linux*; permite também o tratamento de pacotes IPsec de autenticação, cálculos relacionados com a autenticação de pacotes, criação de cabeçalhos *Authentication Header* (AH) e *Encapsulation Security Payload* (ESP) (actualmente e por questões de segurança o *FreeS/Wan* suporta apenas o protocolo ESP); é ainda responsável por verificar todos os pacotes não IPsec, de forma a se assegurar que não estão a ultrapassar as políticas de segurança desse protocolo. O *Pluto* é o "*deamon*" responsável por implementar o protocolo IKE e pela fase das associações seguras do *Internet Security Association Key Management Protocol* (ISAKMP), de execução das autenticações dos clientes e das negociações com as demais *gateways* VPN. Este componente é responsável pela criação das *Security Associations* (SAs) do IPsec e revisão dos dados para envio ao *KLIPS*. Além disso, realiza também o ajuste das rotas e configurações da *firewall* necessárias para adequação aos requisitos do IPsec.

6.1.1.1. Instalação do *FreeS/WAN*

Na implementação utiliza-se a distribuição *Mandrake* 9.2 do *Linux* com a versão do *kernel* 2.4.22-28mdk. Devido a incompatibilidades entre o KLIPS do *FreeS/Wan* e o *kernel* do *Mandrake*, utiliza-se o *SuperFreeS/Wan*, versão 1.99.8-7. Este pacote já inclui os *patches* que permitem a utilização de certificados X.509. O pacote encontra-se disponível em [111] (*super-freeswan-1.99.8-7mdk.i586.rpm*).

Note-se que o *FreeS/Wan* tem de ser instalado nos clientes e nas *gateways* VPN. Após a instalação do pacote *super-freeswan-1.99.8-7mdk*, são criadas as directorias */etc/freeswan* e */etc/freeswan/ipsec.d*. São também instalados todos os ficheiros binários relativos ao IPsec (*ipsec setup*, *ipsec verify*, *ipsec look*, etc.). Na directoria */etc/freeswan* encontram-se os ficheiros de configuração do *SuperFreeS/Wan* – o *ipsec.conf* e o *ipsec.secrets*. Na directoria */etc/freeswan/ipsec.d* disponibilizam-se todos os ficheiros relativos aos certificados X.509.

Para a VPN funcionar correctamente é ainda necessário desactivar o filtro do caminho inverso – *Reverse Path Filter (rp_filter)* – nas *gateways* VPN. Esse filtro é uma das características de segurança do *Linux* que previne ataques do tipo *Denial of Service* (DoS). No entanto, o filtro *rp_fliter*, quando activo, permite que o *kernel* rejeite pacotes que entram na rede e cujas respostas não correspondem à mesma *interface* daqueles que chegaram. Deste modo, os pacotes da VPN são rejeitados pelo *kernel*. Para prevenir que este problema

aconteça, edita-se o ficheiro */etc/rc.d/rc.local*, e adiciona-se a seguinte linha, o que permite desactivar o filtro de caminho inverso:

```
echo 0 > /proc/sys/net/ipv4/conf/eth0/rp_filter
```

O encaminhamento de pacotes IP (*ip forward*), tem de ser activado para que a VPN funcione. O parâmetro de activação encontra-se no ficheiro de configuração do *FreeS/Wan* (*ipsec.conf*).

Uma boa política de segurança introduz a necessidade de se activar a *firewall* na *gateway* VPN (caso esta exista), sendo necessário permitir o tráfego que passa pela porta UDP 500 (negociações IKE) e pelo protocolo 50 (protocolo ESP de cifra e/ou autenticação) entre a *gateway* VPN e os clientes da VPN. Este processo encontra-se descrito na secção 6.1.4.

6.1.1.2. Configuração do *FreeS/WAN*

Após a ferramenta *FreeS/Wan* estar devidamente instalada, é necessário configurá-la para que esta funcione de acordo com as necessidades pretendidas. No caso concreto vai ser utilizado como base da configuração o sistema denominado "*Road Warrior*" (rw) que permite a atribuição dinâmica de endereços IP aos clientes (não esquecendo que são utilizados certificados X.509 para a autenticação dos utilizadores). Os ficheiros que necessitam de configuração são o */etc/ipsec.secrets* e o */etc/ipsec.conf*. Estes ficheiros são a base do funcionamento da VPN pois possuem, respectivamente, as configurações das chaves utilizadas e dos túneis criados.

O ficheiro *ipsec.secrets* contém informação relativa ao RSA, chave pública do cliente, que posteriormente será utilizada para autenticação (mais informações na secção 6.1.2).

Para que o *FreeS/WAN* reconheça ou não a validade dos certificados dos clientes é necessário indicar no ficheiro *ipsec.secrets* a localização das chaves dos clientes (ficheiro *betaKey.pem*) e da *gateway* (ficheiro *gwKey.pem*). Deve também ser indicada a palavra-chave utilizada para proteger esses ficheiros; isso é conseguido alterando-se as seguintes linhas de configuração RSA do ficheiro *ipsec.secrets*:

Para a *gateway* VPN:

```
:RSA        /etc/ipsec.d/private/gwKey.pem  "palavra-chave"
```

Para os clientes VPN, sendo o exemplo a seguir relativo ao cliente "*beta*":

```
:RSA        /etc/ipsec.d/private/betaKey.pem  "palavra-chave"
```

Indica-se a seguir um exemplo de um ficheiro *ipsec.conf* que permite as seguintes opções: *gateway* VPN e cliente VPN (ou *roadwarrior*), autenticação por certificados X.509 e suporte de IP virtual com DHCP sobre IPsec. O ficheiro *ipsec.conf* do cliente VPN está

dividido em três secções: a secção que define os parâmetros das configurações globais (*config setup*), a secção que define os parâmetros das configurações por defeito para todas as ligações (*conn %default*), e a secção que define os parâmetros de configuração do túnel para a ligação da rede sem fios (*conn wlan*). Na secção das configurações globais indica-se que a *interface* de rede a utilizar pelo tráfego IPsec é a que está configurada como cliente DHCP (*interfaces=%defaultroute*). Nesta secção não é necessário alterar outros parâmetros. Na secção da configuração por defeito para todas as ligações, indica-se que a autenticação é baseada em chaves RSA (*authby=rsasig*) e que as chaves públicas são obtidas por certificados X.509 (*rightrsasigkey=%cert* e *leftrsasigkey=%cert*). Na secção relativa à configuração dos parâmetros do túnel da ligação da rede indica-se o endereço da *gateway* VPN (*right=192.168.120.3*) e o endereço da sub-rede à qual o cliente pretende ter acesso (*rightsubnet=192.168.210.0/24*). Nessa secção indica-se também o certificado X.509 da *gateway* VPN (*rightcert=gwCert.pem*) e do cliente (*leftcert=betaCert.pem*). Verifica-se que a *interface* de ligação será atribuída por DHCP (*left=%defaultroute*). A secção termina com a indicação de que o túnel da rede sem fios deverá ser, sempre que o computador seja iniciado, activado (*auto=start*).

```
config setup
      interfaces=%defaultroute
      ……..
conn %default
      ……..
      authby=rsasig
      …….
      rightrsasigkey=%cert
      leftrsasigkey=%cert
conn wlan
      right=192.168.120.3
      rightsubnet=192.168.210.0/24
      rightcert=gwCert.pem
      left=%defaultroute
      leftcert=betaCert.pem
      auto=start
```

Apresenta-se, a seguir, o ficheiro *ipsec.conf* para a *gateway* VPN. Este ficheiro está, tal como o ficheiro do cliente, dividido nas três secções. Na secção *config setup* indica-se a *interface* que permite o tráfego IPsec (*interfaces=”ipsec0=eth2”*), e que deverá ser verificado o valor do *ip_forward* em */proc/sys/net/ipv4/ip_forward*. Se o valor verificado for zero, coloca o valor *ip_forward* a um; se não, mantém o valor um (este parâmetro controla o

encaminhamento dos pacotes). A secção *conn %default* é em tudo idêntica à secção equivalente do ficheiro *ipsec.conf* do cliente. Na secção *conn wlan* indica-se o endereço da *gateway* (*right=192.168.120.3*), o endereço da sub-rede que se pretende aceder (*rightsubnet=192.168.210.0/24*), os certificados da *gateway* (*rightcert=gwCert.pem*) e do cliente (*leftcert=betaCert.pem*), e que o túnel deve ser sempre activado quando o computador arranca (*auto=start*). Nesta secção é também indicado que a *gateway* deverá aceitar pedidos, desde que devidamente autenticados, de qualquer cliente VPN (*left=%any*).

```
config setup
      interfaces=″ipsec0=eth2″
      forwardcontrol=yes
      ……..
conn %default
      …….
      authby=rsasig
      rightrsasigkey=%cert
      leftrsasigkey=%cert
conn wlan
      right=192.168.120.3
      rightsubnet=192.168.210.0/24
      rightcert=gwCert.pem
      left=%any
      leftcert=betaCert.pem
      auto=start
```

6.1.1.3. Configuração da ferramenta IPsec para *Windows* 2000/XP

Para implementar uma VPN IPsec entre clientes Windows (Windows 2000/Windows XP) e o *FreeS/Wan* utiliza-se a ferramenta IPsec para Windows. Caso a implementação utilize o Windows 2000 (é necessário instalar um *service pack igual* ou superior ao 2), a ferramenta IPsec a utilizar deverá ser a *ipsecpol* [40]. No caso da implementação utilizar o Windows XP a ferramenta IPsec será a *ipseccmd* [112].

Após se ter instalado uma destas ferramentas (dependendo da versão do S.O. utilizado) apenas é necessário configurar o ficheiro *ipsec.conf*. Este ficheiro é, ao contrário dos ficheiros *ipsec.conf* da *gateway* e do cliente *Linux*, constituído apenas pela secção de configuração dos parâmetros da ligação de redes (*conn wlan*). Nessa secção indica-se o endereço da *gateway* VPN (*right=192.168.120.3*), o endereço da sub-rede à qual se pretende aceder (*rightsubnet=192.168.210.0/24*) e que a configuração IP do cliente é atribuída por DHCP (*left=%any*). Indica-se também, que o tipo de rede será escolhido automaticamente

(*network=auto*), que o processo de negociação de chaves é protegido (*pfs=yes*), e que o túnel deve ser activado sempre que se reiniciar o computador (*auto=start*). Nesta secção indica-se também o sujeito do certificado da *gateway* (*rightca=”C=PT, S=Porto, L=Porto, O=FCUP, OU=DCC, CN=alfa”*). O valor do sujeito do certificado permite verificar a identidade da Autoridade Certificadora (AC), e é obtido tal como se indica na secção 6.1.2.2.

```
conn wlan
      left=%any
      right=192.168.120.3
      rightsubnet=192.168.210.0/24
      rightca=”C=PT, S=Porto, L=Porto, O=FCUP, OU=DCC, CN=alfa”
      network=auto
      pfs=yes
      auto=start
```

6.1.2. *OpenSSL* – Implementação da Autoridade de Certificação (AC)

Para ser possível a ligação à VPN de clientes sem endereço IP fixo, é necessário implementar uma AC. Para o efeito utiliza-se a ferramenta *OpenSSL* [11] disponível em [111].

Antes de se proceder à criação da nova AC e dos novos certificados devem-se alterar, de acordo com os requisitos do sistema, alguns dos parâmetros do ficheiro *openssl.cnf (/usr/lib/ssl/openssl.cnf*). O valor do parâmetro *´default_days*’ deve ser elevado para permitir uma validade elevada dos certificados, e o valor do parâmetro ‘*default_bits*’ deve ser 2048, o que permite criar chaves privadas mais seguras. Nesse mesmo ficheiro, pode-se também definir novos valores por defeito na secção “*[req_distinguished_name]*” (por exemplo *countryName_default* no caso específico é PT).

Após se terem concluído as alterações ao ficheiro *openssl.cnf*, edita-se o ficheiro */usr/lib/ssl/misc/CA.sh*, alterando-lhe a linha de configuração que indica ‘DAYS=”days 365”’ para um número elevado, superior ao indicado no parâmetro ‘*default_days’* do ficheiro *openssl.cnf*, garantindo assim a validade da AC (caso contrário a AC emitiria certificados que já não seriam válidos).

De seguida, procede-se à criação da nova autoridade de certificação através dos seguintes comandos:

```
# cd /usr/local/ssl/misc
# rm -rf demoCA   (este comando garante-nos que a AC criada é totalmente nova)
# ./CA.pl –newca
```

De seguida, copia-se o novo certificado para a sua directoria. Como o *FreeS/Wan* está configurado para utilizar listas de revogação de certificados (*Certificate Revogation List* - CRL), é necessário criar estas listas:

```
cd /usr/local/ssl/misc
/usr/local/ssl/bin/openssl ca –gencrl –crldays 3650 –out crl.pem
```

O comando anterior cria uma CRL (*-gencrl*) com o nome *crl.pem*, válida por 10 anos (*-crldays 3650*). De seguida, cria-se o certificado de pedidos para a *gateway FreeS/WAN* e assina-se o mesmo com o certificado raiz AC.

```
cd /usr/lib/ssl/misc
./CA.sh –newreq
```

Para verificar se esse certificado se encontra ou não criado, efectua-se o comando seguinte:

```
/usr/local/ssl/misc > ls -lrt n*
-rw-r--r--  1 root  root  2187 May 15 09:44 newreq.pem
```

Para verificar se a chave é ou não encontrada, pode-se utilizar o comando seguinte:

```
#ipsec showhostkey
```

Se como resultado do comando surgir a chave pública, significa que o processo foi realizado com sucesso. Apresenta-se a seguir um exemplo de uma chave pública obtida pelo comando anterior:

```
; RSA 2192 bits   localhost   Mon Nov 15 12:40:31 2004
localhost.  IN    KEY   0x4200 4 1
AQObXSbif/J3c/B7thwIpj07GdPMUR94hlXKpyrJ00qpT93zK1A0QLb6kq4luTlODPpu3
SwG+9qaClPwo6bZfcxhaVNYwz5/Ww5ZUxPRw+eEd2bgagjfmN6Cq25qn8SoV23gV4L7vg
oDqFFz+d8RjtEpwQ8mDjQm0hqLhGPlGmaHzUfgSwNIERpjw40AbiizYkLmkegcjDIklb/
9YuSVZXipfQgHLL7i31x3KQOqEB3PRDywxCY2liGpFViB+CCNLK1c01ssS/GsN1oXNbiB
gL2I/LCRjys73LSFP04uCJoli2tcyQBoLqE+z2OuDw7gXjfmV40dgdo6SkO+GyKtIlA5k
vt5tGcqxeGyXnxV1TqTq6fV
```

6.1.2.1. Criar os certificados de autenticação dos clientes *Linux*

Para o funcionamento correcto da AC é necessário criar um certificado para cada cliente IPsec, utilizando para isso o nome do utilizador, o endereço de *e-mail* e a palavra-chave atribuída. Os comandos utilizados para criar estes certificados são os mesmos que os utilizados para obter o certificado da *gateway*:

```
cd /usr/lib/ssl/misc
./CA.sh newreq
```

De seguida, deve-se introduzir a palavra-chave utilizada no certificado de pedidos do *FreeS/WAN*; toda a restante configuração será personalizada em função do utilizador. O "*Common Name*" de cada máquina deverá ser por questões de segurança o *hostname* da mesma.

Após se terem efectuado as operações de criação das chaves e certificados para todos os intervenientes na VPN, editam-se os ficheiros que fornecem as chaves dos certificados (*gwKey.pem*, <*nomecliente*>*Key.pem*), de forma a obter um ficheiro em que seja simples a identificação da palavra-chave.

Para ser possível ao *FreeS/Wan* encontrar as chaves e os certificados, estes devem ser colocados nas directorias apropriadas (*etc/freeswan/ipsec.d*). Nos clientes, para além da chave e do certificado de cada cliente, deve-se ter acesso ao certificado da *gateway*, ao certificado raiz AC e à CRL (obtidos na secção 6.1.2).

Como todos os ficheiros anteriormente mencionados são criados na *gateway*/AC, estes devem ser copiados de forma segura para os clientes remotos.

6.1.2.2. Criar os certificados de autenticação dos clientes *Windows*

Para se obter um certificado capaz de ser utilizado em máquinas *Windows* têm de ser efectuadas mais algumas operações. As máquinas *Windows* não reconhecem ficheiros com extensão *.pem* ou *.der*, sendo necessário a partir do ficheiro *.pem* obtido na secção anterior, criar um ficheiro com extensão *.p12*.

Efectua-se também o seguinte comando, que nos permite obter o nome do sujeito do certificado, ou seja a identificação do sujeito ao qual o certificado diz respeito, que depois será utilizado no ficheiro de configuração da ferramenta IPsec para *Windows* (*rightca* ou *leftca* do ficheiro *ipsec.conf* na secção 6.1.1.3):

```
# openssl x509 –in demoCA/cacert.pem –noout –subject
```

Depois de criado o ficheiro <*wincliente*>*.p12*, copia-se o mesmo para a máquina cliente *Windows*, de forma segura.

Para iniciar o cliente numa máquina *Windows* inicia-se a consola de gestão do *Windows* – mmc (*Microsoft Management Console*).

De seguida adiciona-se uma nova zona de configuração (*snap-in*) à consola (Figura 6.2). Na janela de adição de um *snap-in* indica-se que se pretende adicionar um certificado (Figura 6.3). Escolhe-se depois a opção “conta de computador” e selecciona-se “computador local”.

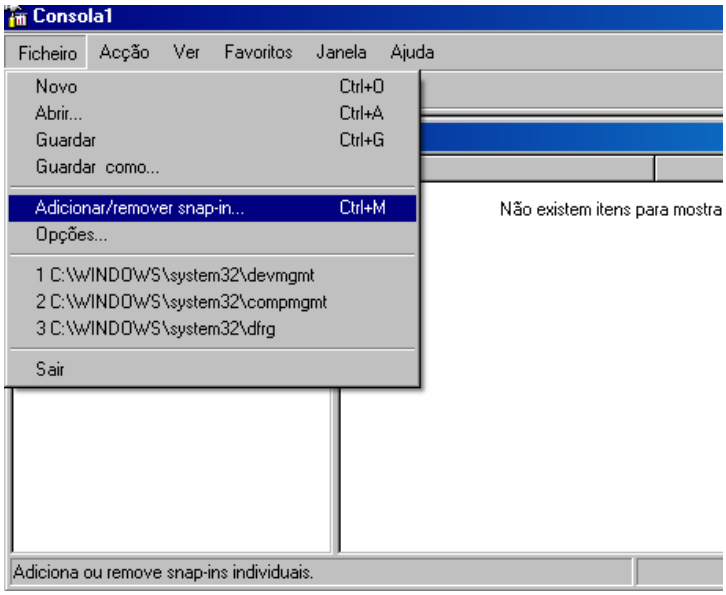


Figura 6.2 - Adicionar um *snap-in*.

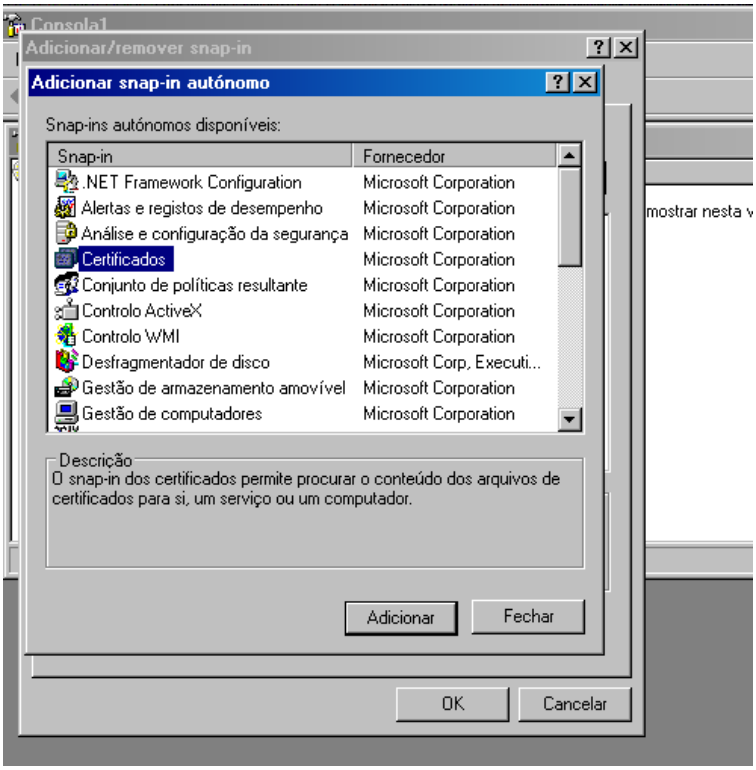


Figura 6.3 - Adicionar certificados na mmc.

Depois de adicionado o *snap-in* de certificados à consola, adiciona-se o certificado. Para esse efeito, importa-se o certificado (Figura 6.4), indica-se o caminho para o certificado, e coloca-se a palavra-chave utilizada para criar a chave privada.

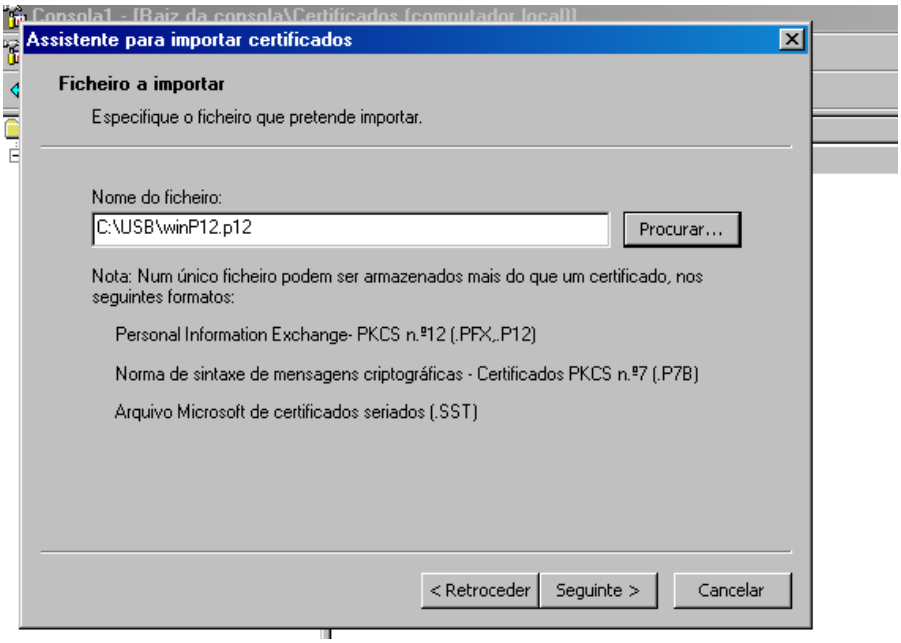


Figura 6.4 - Localização do certificado em formato *.p12*.

6.1.3. Implementação de um Ponto de Acesso (AP) baseado no *Hostap*

Ao nível das necessidades de *hardware*, a implementação de um AP baseado no sistema operativo *Linux* não é muito exigente. Uma alternativa muito económica é utilizar uma *system board* baseada num processador compatível com *Intel*. Ao nível do *software*, para além do sistema operativo, o *Linux*, é necessário um *software* específico que permita realizar as funções de AP. Esse software pode ser o *HostAP* disponível em [36], ou o AVS AP disponível em [37]. Note-se que o *HostAP* e o AVS AP apenas funcionam com placas de rede sem fios baseadas no *chipset* PRISM 2, 2.5 ou 3.

O *HostAP* possibilita o funcionamento com a norma 802.11b. Para implementar o AP é necessário utilizar o *Linux Kernel* 2.4.x ou 2.6.x, o *Hostap driver*, as *Hostap utils* e as *Linux wireless extensions* v15 ou superiores disponíveis em [111].

O *kernel* do *Mandrake* utilizado, o 2.4.22-28mdk, já vem compilado com a condição "*Network device support-> wireless LAN (non-hamradio)*" activa, o que permite de forma imediata a utilização de placas de rede sem fios compatíveis com a norma IEEE802.11 e a utilização das *wireless extensions*.

6.1.3.1. Instalação do *HostAP driver*

A distribuição do *Linux Mandrake* 9.2 com o *kernel* 2.4.28mdk já se encontra com o *HostAP* pré-compilado no *kernel*, residindo os módulos *HostAP* em */lib/modules/kernel-2.4.28mdk/kernel/hostap*.

Após a instalação do *HostAP* acrescenta-se no ficheiro */etc/modules.conf*, o *driver HostAP* necessário:

```
alias wlan0 hostap-pci (para placas pci)
```

ou

```
alias wlan0 hostap-cs  (para placas pcmcia, neste caso deve-se também activar o pcmcia)
```

ou

```
alias wlan0 hostap-plx (para placas híbridas, i.e. com adaptador pcmcia/pci)
```

Depois de realizado este passo efectua-se o comando *dmesg* de forma a carregá-lo para o *syslog*.

6.1.3.2. Utilizar o *HostAP* com as "*Linux wireless extensions*"

A ferramenta *iwconfig* permite configurar o modo, o canal e o *ssid* da placa. A forma de iniciar o sistema em modo AP (*master*) é a seguinte:

```
# ifconfig wlan0 192.168.120.1
# iwconfig wlan0 essid ipsec
# iwconfig wlan0 channel 6 (a escolha deste canal é meramente exemplificativa)
# iwconfig wlan0 mode master
```

6.1.3.3. *HostAP* a efectuar *bridging*

Para se conseguir um AP completo, este deve ser capaz de efectuar *bridging* entre a rede sem fios e a rede com fios. Existe um conjunto de ferramentas de *bridging* [113], que em conjunto com o *HostAP*, permitem criar um AP com todas as funcionalidades.

São então instaladas estas ferramentas que permitem realizar *bridging* entre duas redes. Para configurar o *HostAP* como *bridge* entre a rede sem fios (*wlan0*) e a rede com fios (*eth1*), cria-se inicialmente uma *interface* de *bridging br0* (*brctl addbr br0*). De seguida, adiciona-se a *interface eth1* (*brctl addif br0 eth1*) e a *interface wlan0* (*brctl addif br0 wlan0*) à *interface* de *bridging*. Para ser possível configurar o endereço IP da *interface* de *bridging*, deve-se colocar o endereço IP das *interfaces eth1* e *wlan0* a zero (*ifconfig eth1 0.0.0.0* e *ifconfig wlan0 0.0.0.0*). Finalmente, activa-se a *interface* de *bridging* com o endereço IP pretendido (*ifconfig br0 192.168.120.1 up*). Transcrevem-se a seguir todos os comandos necessários à configuração de uma *interface* de *bridging*:

```
# brctl addbr br0
# brctl addif br0 eth1
# brctl addif br0 wlan0
```

```
# ifconfig eth1 0.0.0.0
# ifconfig wlan0 0.0.0.0
# ifconfig br0 192.168.120.1 up
```

Para verificar se a *bridge* está a funcionar correctamente, faz-se um *display* das características da *interface* de *bridging*:

```
# brctl show
bridge name        bridge id               STP enabled     interfaces
br0              00.00004c9f0bd2             no              wlan0

                                                             eth1
```

Verifica-se que esta *interface* está associada às *interfaces* das redes com e sem fios.

6.1.3.4. *HostAP* nos clientes

Para permitir a utilização do *HostAP* nos clientes, este deve ser instalado no modo *managed*. O exemplo a seguir é baseado na máquina cliente *beta*:

```
# iwconfig wlan0 essid ipsec
# iwconfig wlan0 channel 6
# iwconfig wlan0 mode managed
```

Para verificar se o *HostAP driver* no modo *managed* se associou com o *HostAP driver* no modo *master*, insere-se o comando *iwconfig*, que indica o endereço *MAC* do AP ao qual se associou; neste caso responde com o endereço *MAC* da placa de rede que utiliza o *HostAP driver* em modo *master*, tal como indica a transcrição seguinte*:*

```
eth1      IEEE 802.11-DS  ESSID:"ipsec"  Nickname:"beta"
          Mode:Managed  Frequency:2.412GHz  Access Point: 00:02:2D:02:8F:EF
          Bit Rate:11Mb/s   Tx-Power=20 dBm   Sensitivity=1/3
          RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality:176  Signal level:0  Noise level:0
          Rx invalid nwid:4304  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:46862   Missed beacon:0
```

6.1.4. *Firewall*

A maioria das *gateways* VPN são instaladas ou em *firewalls* ou em redes protegidas por *firewalls*. Sendo assim, é necessário configurar a *firewall* para permitir que o tráfego originado pela porta UDP 500 (negociações IKE), e pelo protocolo 50 (protocolo de cifra e autenticação ESP) seja permitido. Para garantir uma maior segurança no sistema, apenas deverá ser permitido o tráfego de e para a interface IPsec. Uma forma muito simples de permitir esse tráfego será criar um ficheiro, utilizando a ferramenta *Linux iptables*, com um conjunto de regras que satisfaçam esses requisitos. O ficheiro está dividido em duas secções, numa primeira secção configuram-se as regras que permitem negociações IKE e ESP. Nesta

secção criam-se duas regras que permitem (*ACCEPT*) os pedidos de negociação do protocolo ESP: uma regra refere o ESP utilizando o número da porta por ele utilizada (*iptables –A INPUT –p 50 –j ACCEPT*), e a outra utiliza o nome (*iptables –A INPUT –p esp –j ACCEPT*). Esta secção compreende também uma regra que permite o envio (*OUTPUT*) de respostas de negociação IKE (*iptables –A OUTPUT –p udp--sport 500 –dport 500 –j ACCEPT*). Para concluir esta secção criam-se duas regras que permitem o envio de respostas de negociação do protocolo ESP. Uma regra refere-se ao ESP pelo nome (*iptables –A OUTPUT –p esp –j ACCEPT*) e a outra pelo número da porta utilizada (*iptables –A OUTPUT –p 50 –j ACCEPT*).

A segunda secção permite apenas tráfego de e para a interface IPsec. As primeiras regras definem que não é permitido (*DROP*) o encaminhamento de tráfego (*FORWARD*) não protegido entre a *interface eth1* e *eth2* (*iptables –A FORWARD –i eth1 –o eth2 –j DROP*), e entre *eth2* e *eth1* (*iptables –A FORWARD –i eth2 –o eth1 –j DROP*). Criam-se também duas novas regras que permitem o encaminhamento de tráfego IPsec entre a *interface ipsec0* e a *eth1* (*iptables –A FORWARD –i ipsec0 –o eth1 –j ACCEPT*), e entre a *interface eth1* e a *ipsec0* (*iptables –A FORWARD –i eth1 –o ipsec0 –j ACCEPT*). Esta secção termina com a definição de duas regras, que não permitem tráfego não protegido, originado (*iptables –A INPUT –i eth2 –j DROP*) e com destino (*iptables –A OUTPUT –o eth2 –j DROP*), à interface *eth2*. A seguir, apresenta-se o ficheiro resultante:

```
# Seccção que permite negociações IKE, autenticação e cifra ESP
iptables –A INPUT –p udp - -sport 500 - -dport 500 –j ACCEPT
iptables –A INPUT –p 50 –j ACCEPT
iptables –A INPUT –p esp –j ACCEPT
iptables –A OUTPUT –p udp --sport 500 –dport 500 –j ACCEPT
iptables –A OUTPUT –p 50 –j ACCEPT
iptables –A OUTPUT –p esp –j ACCEPT
# Secção que permite apenas tráfego de e para a interface IPsec
iptables –A FORWARD –i eth1 –o eth2 –j DROP
iptables –A FORWARD –i eth2 –o eth1 –j DROP
iptables –A FORWARD –i ipsec0 –o eth1 –j ACCEPT
iptables –A FORWARD –i eth1 –o ipsec0 –j ACCEPT
iptables –A INPUT –i eth2 –j DROP
iptables –A OUTPUT –o eth2 –j DROP
```

6.1.5. Descrição da colocação em funcionamento

A colocação em funcionamento da implementação antes descrita é muito simples. Primeiro é necessário criar um *script* na *gateway* VPN com as regras de *firewall* descritas na secção 6.1.4, executando-se de seguida o *script*. De seguida, activa-se o AP tal como descrito

na secção 6.1.3. Como os equipamentos da implementação estão configurados para estabelecerem de forma automática o túnel, é apenas necessário ligar um dos clientes VPN para que o túnel IPsec se active.

6.2. Cenário 2 – WPA-EAP

O cenário de suporte de segurança através de WPA-EAP encontra-se apresentado na Figura 6.5. Como se referiu na secção 5.4.5, cada terminal ligado à rede sem fios deve conter um *software* cliente WPA (*supplicant*); o AP deve suportar o protocolo WPA, e é utilizado um servidor de autenticação que implementa o RADIUS. Os elementos constituintes deste cenário e respectiva configuração encontram-se apresentados na Figura 6.5.

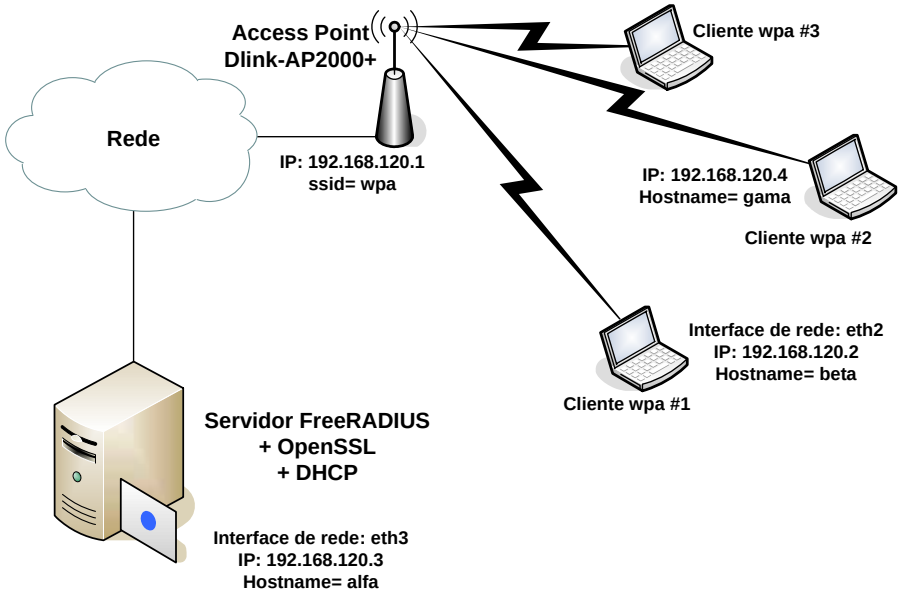


Figura 6.5 - Cenário experimental WPA -EAP.

Para implementar a estrutura relativa ao servidor de autenticação, utilizam-se duas implementações disponíveis: o *OpenSSL* [11] e o *FreeRADIUS* [12]. O *OpenSSL* fornece as primitivas criptográficas e algumas ferramentas que permitem manipular e criar certificados de chave pública. O *OpenSSL* permite criar certificados digitais auto-assinados, podendo-se implementar um sistema limitado de chaves públicas e certificados, mas suficiente para condições de teste. O *FreeRADIUS* fornece as capacidades de servidor de autenticação, utilizando o protocolo *RADIUS* como ponto central da infra-estrutura de autenticação.

Caso se pretenda utilizar o WPA numa rede de média a grande dimensão será necessário implementar uma infra-estrutura de chave pública. A implementação em código aberto que satisfaz esses requisitos é o *OpenCA* [38]; no entanto, é ainda pouco flexível e difícil de implementar. Assim, será utilizado o *OpenSSL* para criar os certificados digitais.

Os elementos constituintes da rede, suas características e *software* encontram-se apresentados na Tabela 6-4, Tabela 6-5 e Tabela 6-6.

AP

Designação	Características	Código Aberto
D-Link DWL-2000AP+	802.11b, 802.11g, WPA e 802.1X	

Tabela 6-4 - AP num cenário WPA-EAP.

Servidor de Autenticação

Designação	Características	Código Aberto
Computador	*Pentium* II 400 MHz 256 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Placas de rede	Placa PCI 10/100, RJ45	
Software Autenticação	*FreeRADIUS + OpenSSL*	Sim

Tabela 6-5 - Servidor de autenticação num cenário WPA-EAP.

Cliente

Designação	Características	Código Aberto
Computador	*Celeron* 433 128 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Placas de redes sem fios	*Linksys* WMP54g-EU, 802.11b e g, WPA e 802.1X	
Software cliente	*wpa_supplicant*	Sim

Tabela 6-6 - Cliente WPA num cenário WPA-EAP.

As secções seguintes descrevem a implementação dos diferentes componentes do cenário a ser implementado.

6.2.1. *OpenSSL*

Devido a problemas de interacção entre o *OpenSSL* e o *FreeRADIUS*, convém utilizar as versões mais recentes para ambos. A última versão do *OpenSSL* encontra-se disponível em [11].

Após a instalação do *OpenSSL* é necessário efectuar pequenas alterações no ficheiro de configuração, de forma a ser mais fácil a emissão dos certificados. O ficheiro no qual se efectuam essas alterações é o *openssl.cnf*, localizado em */usr/local/openssl/ssl/openssl.cnf*. As alterações apenas dizem respeito aos valores por defeito relativos a algumas das opções afectas à criação dos certificados (*countryName_default* - país, *challengePassword_default* – palavra-chave, *default_days* – validade do certificado, *default_bits* – número de *bits* utilizado para criação de chaves).

Criação dos certificados de chave pública.

A implementação de certificados capazes de serem utilizados com o 802.1X requer vários passos. Primeiro, é necessário criar o certificado da autoridade de certificação. Este certificado é auto-assinado, ou seja, a autoridade de certificação atesta a validade de si própria (embora possa parecer estranho é este o método mais aceite para se criar o certificado raiz - *root*). Assim que a autoridade de certificação, ou por outras palavras, o certificado raiz é criado, é necessário criar o certificado do servidor – para o servidor RADIUS – e, por último, os certificados para todos os clientes.

Existem vários exemplos de *scripts*, que nos permitem, de forma automática criar a autoridade de certificação [41]. Para ser possível criar uma AC totalmente nova deve ser eliminada a directoria onde residem os ficheiros de outras AC (*rm –rf demoCA*). Depois inicia-se o processo de geração de um certificado auto-assinado (*openssl req -new -x509 -keyout newreq.pem -out newreq.pem -passin pass:whatever -passout pass:whatever*), deve-se indicar qual a palavra-chave que será utilizada para assinar o certificado (*pass:whatever*). O ficheiro *newreq.pem* contém a chave privada e o certificado da AC. Para permitir a utilização do certificado por máquinas *Windows* é necessário gerar um ficheiro PKCS#12; o ficheiro criado (*root.p12*) utiliza o certificado e a chave anteriormente criados (*openssl pkcs12 -export -in demoCA/cacert.pem -inkey newreq.pem -out root.p12 -cacerts -passin pass:whatever -passout pass:whatever*). Neste comando é indiferente utilizar *demoCA/cacert.pem* ou *newreq.pem.* De seguida, converte-se o ficheiro *root.p12*, que representa a chave privada e o certificado da AC, num ficheiro com formato PEM, o *root.pem* (*openssl pkcs12 -in root.p12 -out root.pem -passin pass:whatever -passout pass:whatever*). Para apenas se ter um ficheiro com o certificado auto-assinado da AC, converte-se o ficheiro *root.pem* para o formato *.der* (*openssl x509 -inform PEM -outform DER -in root.pem -out root.der*). Finalmente, e por questões de segurança remove-se o ficheiro *newreq.pem* (*rm –rf newreq.pem*). Apresenta-se a seguir um exemplo de um *script* que permite criar a AC.

```
#!/bin/sh
export PATH=/usr/lib/misc:${PATH}
```

```
export LD_LIBRARY_PATH=/usr/lib
rm -rf demoCA
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -passin
pass:whatever -passout pass:whatever
openssl pkcs12 -export -in demoCA/cacert.pem -inkey newreq.pem -out
root.p12 -cacerts -passin pass:whatever  -passout pass:whatever
openssl pkcs12 -in root.p12 -out root.pem -passin pass:whatever -
passout pass:whatever
openssl x509 -inform PEM -outform DER -in root.pem -out root.der
rm -rf newreq.pem
```

O ficheiro que implementa o *script* é denominado de *CA.root*. Após criado o *script,* este é executado através do comando seguinte:

```
#/bin/bash CA.root
```

Para ser possível a autenticação mútua entre o servidor RADIUS e o *supplicant*, é necessário um certificado de chave pública no servidor e o certificado do cliente. Exemplos de *scripts* que criam esses certificados podem ser encontrados em [70]. Depois de criados os certificados dos clientes, é necessário copiá-los para uma directoria acessível ao *software* cliente WPA (*wpa_suppplicant* - por exemplo */etc/1x*). Os certificados devem ser criados da seguinte forma: inicialmente efectua-se um novo pedido de criação de um certificado, que será guardado no ficheiro *newreq.pem* (*openssl req -new -keyout newreq.pem -out newreq.pem -passin pass:whatever -passout pass:whatever*). Assina-se, de seguida, o certificado utilizando a chave da AC (*openssl ca -policy policy_anything -out newcert.pem -passin pass:whatever -key whatever –infiles newreq.pem*). Para assinar o certificado é necessário indicar a sua localização (*--infiles newreq.pem*) e a palavra-chave da AC (*-key whatever*); o resultado será guardado no ficheiro *newcert.pem*. Para possibilitar a utilização do certificado por máquinas *Windows*, cria-se um ficheiro PKCS#12 (*openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12 -clcerts -passin pass:whatever -passout pass:whatever*). O ficheiro é criado a partir do novo certificado (*newcert.pem*) e da sua chave privada (*-inkey newreq.pem*); para ser possível criá-lo, indica-se a palavra-chave utilizada para criar a chave (*-passin pass:whatever*). Converte-se, de seguida, este ficheiro para o formato *.pem* (*openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:whatever -passout pass:whatever*), o que possibilita obter um certificado num ficheiro com o formato DER (*openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der*). Depois de obtido o certificado assinado pela AC, eliminam-se os ficheiros *newcert.pem* e *newreq.pem* (*rm –rf newcert.pem newreq.pem*). Apresenta-se a seguir o *script* que cria os certificados do cliente ou do servidor:

```
#!/bin/sh
export PATH=/usr/lib/misc:${PATH}
export LD_LIBRARY_PATH=/usr/lib
openssl req -new -keyout newreq.pem -out newreq.pem -passin
pass:whatever -passout pass:whatever
```

```
openssl ca -policy policy_anything -out newcert.pem -passin
pass:whatever -key whatever -infiles
newreq.pem
openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12
-clcerts -passin pass:whatever -passout pass:whatever
openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:whatever -passout
pass:whatever
openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der
rm -rf newcert.pem newreq.pem
```

Após criado o *script*, este é executado da forma seguinte para o cliente:

```
#/bin/bash CA <nome do cliente>
```

Como resultado deste comando são criados os ficheiros <*nome do cliente*>*.p12*, <*nome do cliente*>*.pem* e <*nome do cliente*>*.der*. Da mesma forma executa-se o *script* para o servidor.

6.2.2. Servidor RADIUS

O servidor RADIUS utilizado neste cenário é o *FreeRADIUS* [12]. O RADIUS é um protocolo complexo e que pode ser configurado de várias formas diferentes. Após a sua instalação são efectuadas as alterações necessárias aos ficheiros de configuração do *FreeRADIUS*. Existem quatro ficheiros que devem ser alterados e que estão localizados na directoria */usr/local/radius/etc/raddb*. Esses ficheiros são o *clients.conf*, *radiusd.conf, eap.conf* e *users*.

O ficheiro *clients.conf* é responsável por determinar "quem/o quê" se pode ligar ao servidor RADIUS para autenticar os utilizadores. No caso da implementação WPA pretende-se que os APs tenham autorização para se ligarem ao servidor RADIUS. Existem duas formas de adicionar a informação relativa aos APs: uma consiste em listar cada AP de forma individual; a outra consiste em listar a sub-rede à qual pertencem os APs. O exemplo a seguir ilustra como é adicionada a informação de cada AP individualmente. Pode-se verificar que a primeira linha (*client*) indica ao servidor que o AP com o endereço IP 192.168.120.1 está autorizado a efectuar uma ligação. O campo *shortname* serve apenas para atribuir uma identificação ao AP. O campo *secret* é utilizado para autenticação entre o AP e o servidor, representando a palavra-chave comum ao AP e ao servidor.

```
client 192.168.120.1{
        secret      = <palavra-chave comum ao AP e ao servidor RADIUS>
        shortname   = AP-DLink
        }
```

No ficheiro *eap.conf,* que é responsável pela configuração dos parâmetros relativos à autenticação EAP, é necessário alterar as secções relativas ao tipo de EAP utilizado para autenticação, ao nome do ficheiro da chave privada, à localização do certificado e da AC. É

também necessário alterar a *private_key_password*, que representa a palavra-chave utilizada para assinar os certificados.

A criação da chave de cifra utilizada no processo de *hashing*[6] e da chave de sessão utilizada para a cifra dos dados, entre o cliente e o AP, utiliza caracteres aleatórios contidos em dois ficheiros, *random* e *dh*. A chave de sessão é então enviada pelo servidor de autenticação (servidor RADIUS) para o AP, possibilitando assim a criação de um túnel seguro entre o cliente e o AP (o cliente já conhece a chave de sessão). Apresenta-se de seguida um exemplo de um ficheiro *eap.conf*. Neste ficheiro podem ser alterados os parâmetros das secções *eap* e *tls*. O valor do parâmetro *default_eap_type* é alterado para *tls*: este parâmetro indica ao servidor que deverá ser utilizado por defeito o protocolo de autenticação TLS com o método EAP. A sub-secção *tls* serve para indicar ao servidor onde está localizado o seu certificado (*certificate_file*), a sua chave privada (*private_key_file*), o certificado da AC (*CA_file*), os ficheiros *random* e *dh*, e a palavra-chave (*private_key_password*) utilizada quando se criam as chaves privadas do servidor.

```
eap {
      default_eap_type = tls
      timer_expire = 60
      ignore_unknown_eap_types = no
      cisco_accounting_username_bug = no
      md5 {
      }
      leap {
      }
      gtc {
      auth_type = PAP
      }
tls {
      private_key_password = whatever
      private_key_file = /etc/1x/cert-srv.pem
      certificate_file = /etc/1x/cert-srv.pem
      CA_file = /etc/1x/demoCA/cacert.pem
      dh_file = /etc/1x/dh
      random_file = /etc/1x/random
      fragment_size = 1024
      include_length = yes
}
```

[6] Processo de utilização de uma função de *hash*.

No ficheiro *radiusd.conf*, que contém os parâmetros de configuração da autenticação RADIUS, é apenas necessário configurar as secções de autenticação e autorização de forma a suportarem o EAP. Esta configuração é apresentada na transcrição seguinte:

```
authorize {
            eap
}
authenticate {
            eap
}
```

No ficheiro *users* com informação dos clientes, apenas tem que se indicar o nome utilizado no parâmetro "*Common Name*" (quando se criam os certificados dos clientes) e o tipo de autenticação. Como se pretende uma autenticação baseada no mecanismo EAP, a forma de configurar tal situação no ficheiro users será *Auth-Type: = EAP*. O exemplo a seguir indica ao servidor que o "*Common Name*" utilizado na criação do certificado do cliente *beta* é *beta*, e que a autenticação será baseada no mecanismo EAP:

```
beta  Auth-Type:=EAP
```

Após todas estas alterações, o servidor RADIUS está pronto a ser utilizado.

```
# radiusd –X –A
```

6.2.3. Configuração do Ponto de Acesso (AP)

Figura 6.6 - Pedido de autenticação do AP.

O AP escolhido para esta implementação é o DWL-2000AP+ da *DLink* [45]. A principal razão para se utilizar este AP centra-se na indisponibilidade de uma implementação de *software* em código aberto com a implementação de AP com suporte WPA[7]. A configuração do AP pode ser realizada através de uma *interface web*; note-se que o objectivo é configurar o AP para funcionar em modo WPA-EAP.

A *interface web* do AP é acessível do endereço http://192.168.0.50 (endereço configurado por defeito pelo fabricante do AP). A janela ilustrada na Figura 6.6 permite

[7] Muito recentemente, nomedamente no final de 2004, foi disponibilizada uma versão do *HostAP* com suporte WPA

efectuar a autenticação do AP. A configuração é efectuada da seguinte forma. No menu "W*ireless"* (Figura 6.7) configura-se o "*ssid*" para "*wpa*" e o tipo de WPA para "*WPA*".

Figura 6.7 - Configuração dos parâmetros WPA do AP.

No menu "*LAN"* (Figura 6.8) atribui-se o endereço IP 192.168.120.1 ao AP.

Figura 6.8 - Configuração TCP/IP do AP.

No menu "*Advanced"* (Figura 6.9) selecciona-se o sub-menu "*802.1X*", activa-se o 802.1x e indica-se o endereço IP do servidor *FreeRADIUS*, 192.168.120.3, bem como a palavra-chave definida no ficheiro *clients.conf* do servidor *FreeRADIUS*.

Após estas configurações o AP está pronto a ser utilizado.

Figura 6.9 - Configuração dos parâmetros 802.1X do AP.

6.2.4. *Software supplicant – wpa_supplicant*

Para permitir a activação de todas as funcionalidades WPA é necessário utilizar nos clientes o *software* cliente WPA (o *wpa_supplicant* [39]). As versões mais recentes deste *software* já integram o cliente WPA (*wpa_supplicant*) e o cliente 802.1X (*xsupplicant*). A integração destes dois *supplicant*s num único torna a implementação mais simples e homogénea. Depois de obtido o *wpa_supplicant* é necessário alterar o ficheiro *.config* de modo a permitir a utilização de autenticação EAP-TLS (e outras opções) na maioria dos *drivers* para placas de rede sem fios. Estas alterações estão apresentadas de seguida. Para não haver problemas de suporte de algum *driver* ou protocolo de autenticação, todas as opções são colocadas em modo activo (*y*).

```
CONFIG_WIRELESS_EXTENSION=y
CONFIG_DRIVER_HOSTAP=y
CONFIG_DRIVER_PRISM54=y
CONFIG_DRIVER_WEXT=y
CONFIG_DRIVER_NDISWRAPPER=y
CONFIG_IEEE8021X_EAPOL=y
CONFIG_EAP_MD5=y
CONFIG_MSCHAPV2=y
CONFIG_EAP_TLS=y
CONFIG_EAP_PEAP=y
CONFIG_EAP_TTLS=y
CONFIG_EAP_GTC=y
CONFIG_EAP_OTP=y
CONFIG_EAP_SIM=y
CONFIG_EAP_LEAP=y
```

Após a instalação do *software*, verifica-se se o cliente e o AP se conseguem associar. Para esse efeito, antes de executar o *wpa_supplicant*, executa-se o comando seguinte:

```
/sbin/iwconfig eth2 essid wpa
/sbin/ifconfig eth2 up
```

Ao activar a placa de rede, configura-se o ficheiro de configuração do *wpa_supplicant*, o *wpa_supplicant.conf*, de forma a satisfazer os requisitos da solução. Nesse ficheiro deve ser criada uma secção de configuração de rede (parâmetro *network*). Nessa secção indica-se a identificação da rede (*ssid="wpa"*), indica-se a forma como é efectuada a gestão das chaves de cada sessão (*key_mgmt=WPA-EAP*) e o protocolo utilizado para a gestão das chaves (*pairwise=TKIP*). Na mesma secção é indicado o tipo de autenticação EAP a utilizar (*eap=TLS*). Como a autenticação EAP-TLS é baseada em certificados, é necessário indicar o local onde se encontra o certificado da AC (*ca_cert*), o certificado do cliente (*client_cert*), a

chave privada do cliente (*private_key*) e a palavra-chave utilizada na criação da chave privada do cliente (*private_key_passwd*). Apresenta-se a seguir um ficheiro *wpa_supplicant.conf* com as estas opções:

```
# Para a configuração TLS
network={
      ssid="wpa"
      key_mgmt=WPA-EAP
      pairwise=TKIP
      group=TKIP
      eap=TLS
      identity="beta"
      ca_cert="/etc/1x/root.pem"
      client_cert="/etc/1x/cert-clt.der"
      private_key="/etc/1x/cert-clt.pem"
      private_key_passwd="whatever"
      priority=1
}
```

De seguida, executa-se o *wpa_supplicant* com as opções de indicação da localização do ficheiro de configuração *wpa_supplicant.conf* (opção –c */etc/wpa_supplicant.conf*), com a indicação da *interface* de rede a utilizar (opção –i *eth2*), e com a opção do nível de *debug* a utilizar (opção –dd). É também possível indicar qual o *driver* a utilizar pelo *wpa_supplicant* (opção -D).

```
# wpa-supplicant –c /etc/wpa_supplicant.conf  –ieth2 –D hostap -dd
```

Para se verificar se a autenticação e o protocolo WPA são bem sucedidos, utiliza-se o seguinte comando:

```
# wpa_cli status
```

Este comando apresenta no ecrã um conjunto de informações relativas ao processo WPA-EAP. As informações mais importantes são o estado de sucesso da autenticação do cliente (*supplicant PAE state=AUTHENTICATED*) e o estado da autenticação EAP (*EAP state=SUCCESS*). Apresenta-se a seguir o resultado obtido pelo comando *wpa_cli status* após uma autenticação WPA-EAP bem sucedida.

```
Selected interface 'eth3'
bssid=00:0f:3d:09:d3:ea
ssid=wpa
pairwise_cipher=TKIP
group_cipher=TKIP
key_mgmt=WPA/IEEE 802.1X/EAP
wpa_state=COMPLETED
Supplicant PAE state=AUTHENTICATED
```

```
heldPeriod=60
authPeriod=30
startPeriod=30
maxStart=3
suppPortStatus=Authorized
portControl=Auto
Supplicant Backend state=IDLE
EAP state=SUCCESS
reqMethod=0
selectedMethod=13
methodState=DONE
decision=UNCOND_SUCC
ClientTimeout=60
```

6.2.5. Descrição da colocação em funcionamento

Para ser possível utilizar o mecanismo WPA-EAP é necessário, em primeiro lugar, criar os certificados digitais do servidor de autenticação e dos clientes (secção 6.2.1). De seguida, é necessário copiar os certificados de forma segura para os clientes. A seguir deve-se configurar e iniciar o serviço de autenticação RADIUS (secção 6.2.2). Depois de o serviço RADIUS estar em funcionamento e o AP estar activo, activa-se o *software supplicant* nos clientes.

6.3. Cenário 3 – WPA-PSK.

O cenário de suporte de segurança em redes sem fios através do protocolo WPA no modo PSK encontra-se apresentado na Figura 6.10 Este cenário requer que cada terminal contenha um *software* cliente WPA, e um AP com suporte WPA. Os elementos constituintes deste cenário encontram-se apresentados na Figura 6.10.

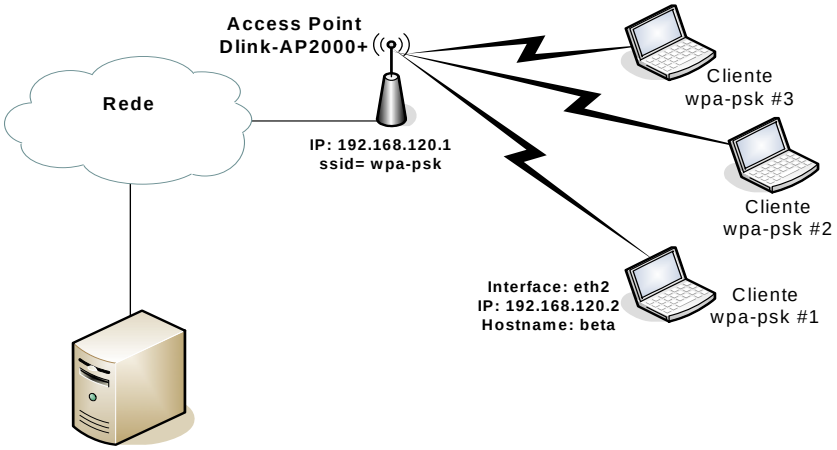


Figura 6.10 - Cenário experimental WPA-PSK.

Os elementos constituintes da rede, suas características e *software* suportado encontram-se apresentados na Tabela 6-7 e na Tabela 6-8.

AP

Designação	Características	Código Aberto
D-Link DWL-2000AP+	802.11b, 802.11g, WPA e 802.1X	

Tabela 6-7 - AP num cenário WPA-PSK.

Cliente WPA-PSK

Designação	Características	Código Aberto
Computador	*Celeron* 433 128 MB de RAM Disco rígido por NFS	
Sistema Operativo	*Linux Mandrake 9.2*	Sim
Software Cliente	*wpa_supplicant*	Sim
Placas de rede sem fios	*Linksys* WMP54g-EU, 802.11b e g, WPA, 802.1X	

Tabela 6-8 - Cliente WPA-PSK num cenário WPA-PSK.

Para se implementar um cenário WPA-PSK em código aberto, utiliza-se o *software wpa_supplicant* desenvolvido pelo grupo de trabalho do *HostAP* [36]. A configuração do *AP* é também muito semelhante à do cenário WPA-EAP, sendo apenas necessário que no menu "*wireless*" (Figura 6.11) seja activada a opção *wpa-psk*, seja escrita a palavra-chave partilhada entre o AP e os clientes da rede sem fios, e seja definido o *ssid* da rede. A palavra-chave representa a *Pairwaise Master Key* (PMK) responsável por determinar a chave de cifra dos dados da rede (*Pairwaise Transient Key* - PTK). Para se garantir um nível de segurança significativo a palavra-chave deve ser aleatória e conter mais de vinte caracteres.

Figura 6.11 - Configuração WPA-PSK do AP.

O *wpa_supplicant* é o mesmo que o utilizado no cenário anterior e a sua instalação é efectuada da mesma forma. Após a instalação do *software,* e tal como no cenário WPA-EAP, ficam disponíveis duas aplicações: a *wpa_passphrase* (apenas útil para este cenário) e o *wpa_suppliant*.

O *wpa_passphrase* permite, através da combinação entre o *ssid* e a palavra-chave partilhada, escrever a palavra-chave de uma forma pouco legível à mente humana:

```
# wpa_passphrase wpa-psk <palavra-chave partilhada com o AP>
```

O resultado é utilizado no ficheiro de configuração do *wpa_supplicant* (*wpa_supplicant.conf*). A transcrição seguinte representa um exemplo do ficheiro *wpa_supplicant.conf* que possibilita o WPA-PSK. Este ficheiro é muito semelhante ao do cenário WPA-EAP. Neste ficheiro deve-se indicar a palavra-chave obtida pela execução da aplicação *wpa_passphrase* (parâmetro *psk*) e indicar o método de gestão das chaves (*key_mgmt=WPA-PSK*). Toda a restante configuração é muito semelhante à do cenário WPA-EAP.

```
network={
      ssid="wpa-psk"
      proto=WPA
      key_mgmt=WPA-PSK
      pairwise=CCMP TKIP
      group=CCMP TKIP WEP104 WEP40
      psk=<resultado obtido pelo wpa_passphrase>
}
```

Por último, activa-se o *wpa_supplicant*:

```
# wpa_suppplicant –c /etc/wpa_supplicant.conf –i eth2
```

6.3.1. Descrição da colocação em funcionamento

O processo de colocação em funcionamento desta implementação é muito simples. Após se ter activado o AP, é apenas necessário activar no cliente o *wpa_supplicant*.

6.4. Conclusão

Ao longo deste capítulo foram apresentados um conjunto de cenários que possibilitam a implementação de vários mecanismos de segurança nas redes sem fios. As implementações aqui descritas baseiam-se no protocolo IPsec e no protocolo WPA. Com base no protocolo IPsec, foi descrita a implementação de uma VPN IPsec em redes sem fios. Com base no protocolo WPA, foi descrita a implementação de um sistema WPA-EAP e de um sistema WPA-PSK. Em cada uma das implementações foram descritas ferramentas que permitem a

execução desses cenários, utilizando-se sempre que possível ferramentas em código aberto. Foram também descritos os passos necessários à configuração e utilização das ferramentas necessárias.

As implementações WPA-EAP e IPsec utilizam, para autenticação mútua entre o cliente e o servidor de autenticação, certificados digitais. A implementação WPA-PSK utiliza uma palavra-chave partilhada, entre o cliente e o AP, para permitir a cifra dos dados entre ambos. A implementação IPsec garante apenas a autenticação entre o cliente e o servidor, não garantindo a autenticação entre o AP e o cliente. Na realidade o AP é apenas utilizado como *bridge* entre a rede sem fios e a rede com fios, não sendo configurado com qualquer mecanismo de segurança. Esta situação pode ser um problema, já que permite ataques de DoS ao AP, bem como a possibilidade de se utilizarem APs não autorizados para atacarem a implementação. Outra particularidade desta implementação é o facto de todo o processamento relativo aos mecanismos de segurança ser realizado no cliente e no servidor, possibilitando assim a utilização de um AP sem especificações especiais.

A implementação WPA-EAP exige a utilização de um AP que implemente o protocolo WPA. Esta implementação garante, para além da autenticação mútua entre o cliente e o servidor, a autenticação mútua entre o cliente e o AP. Todos os mecanismos de segurança dos dados são realizados entre o AP e o cliente. Em situações de muito congestionamento da rede, ou seja numa rede com muitos clientes, esta implementação pode apresentar uma degradação do desempenho da rede. Esta implementação é bastante robusta em termos de segurança, sendo no entanto, muito fácil a realização de ataques de DoS.

A implementação WPA-PSK é a menos exigente em termos de configuração, mas é também aquela que menos garantia de segurança disponibiliza. Não utiliza nenhum mecanismo de autenticação robusto, baseando-se o seu funcionamento no conhecimento prévio de uma palavra-chave comum ao AP e ao cliente. Neste tipo de implementação é possível efectuar ataques ao AP e ao cliente de forma a se obter a palavra-chave.

A utilização de ferramentas em código aberto apresenta como principal desvantagem, relativamente à utilização de ferramentas comerciais, a maior exigência ao nível da configuração. Esta característica exige um maior conhecimento técnico para possibilitar a configuração e utilização de forma correcta destas ferramentas. Esta desvantagem é, em contrapartida, minimizada pela sua disponibilidade gratuita e a possibilidade de adaptar estas ferramentas às necessidades de cada situação.

Capítulo 7

Resultados de desempenho e protecção de segurança

O processo de suporte de segurança numa rede sem fios envolve a interoperação de um conjunto de protocolos que, dependendo da solução utilizada, apresentam maior ou menor complexidade, com maior ou menor resistência aos ataques de segurança. Antes de utilizar um protocolo de segurança é importante conhecer o seu funcionamento. Perceber o funcionamento das principais fases de negociação do protocolo pode indicar onde esses protocolos apresentam as suas falhas, e assim tomar as medidas necessárias para eliminar ou reduzir os perigos dessas falhas.

É também importante conhecer contra que tipos de ataques os protocolos de segurança são mais eficazes, para definir em que situações um determinado protocolo de segurança é mais eficaz. As políticas de segurança são também baseadas no conhecimento da eficácia e nível de segurança que cada protocolo acrescenta a uma rede. A definição de uma boa política de segurança garante a protecção e a utilização segura de uma rede.

Embora necessários, os protocolos de segurança diminuem de forma significativa o desempenho de uma rede. A utilização de um determinado protocolo de segurança deve ser bem ponderada. Por vezes a activação de um desses protocolos torna a rede um ponto de estrangulamento das comunicações. Nas situações em que a informação que circula na rede é sensível, deve ser escolhido um mecanismo de segurança que permita uma boa relação protecção/desempenho da rede.

Este capítulo tem três objectivos distintos. O primeiro consiste em verificar experimentalmente o funcionamento das soluções implementadas, tendo em conta as mensagens que são trocadas entre os diversos elementos nos processos de negociação de chaves, autenticação e envio de informação protegida. O segundo objectivo consiste em averiguar a eficiência das soluções implementadas ao nível da sua resistência a ataques de segurança. Finalmente, pretende-se averiguar o desempenho e complexidade das soluções, ao nível do *throughput*, ocupação da largura de banda, atrasos na rede, variação do atraso (*jitter*) e a carga de processamento nos pontos críticos (por exemplo na *gateway Virtual Private Network* -VPN). Os resultados das experiências de desempenho da rede são comparados entre si, e também com os resultados de uma rede a funcionar sem o recurso a nenhuma das implementações descritas (rede plana), para ser possível determinar o verdadeiro custo que cada implementação introduz na rede.

Este capítulo é organizado da seguinte forma: na sua secção 7.1 apresenta-se uma análise ao funcionamento dos protocolos utilizados nas diversas implementações; a secção 7.2 descreve os ataques à segurança dos protocolos implementados e os resultados por eles obtidos; e a secção 7.3 apresenta os testes e resultados de desempenho das soluções de segurança implementadas. Este capítulo conclui na secção 7.4 com uma pequena conclusão e discussão dos resultados obtidos.

7.1. Análise ao funcionamento dos protocolos de segurança implementados

Esta secção apresenta uma análise das mensagens trocadas nas várias fases de funcionamento dos protocolos de segurança (autenticação, negociação de chaves e envio de informação protegida), nas várias implementações efectuadas. A ferramenta utilizada para verificar o funcionamento dos protocolos é o *ethereal* [17]. Este analisador de protocolos fornece uma análise bastante organizada e detalhada dos pacotes capturados, suas mensagens e conteúdos, ao nível de todas as camadas do modelo *Internet*.

7.1.1. Análise ao funcionamento do protocolo IPsec

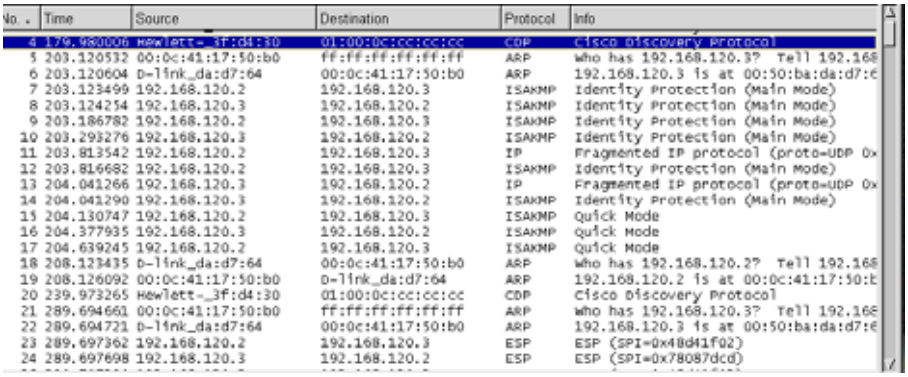
O cenário experimental necessário à análise do funcionamento do protocolo *Internet Protocol Security* (IPsec) é idêntico ao ilustrado na Figura 6.1. A única diferença é a instalação e a configuração da ferramenta *ethereal* num dos clientes e na *gateway* VPN.

Para ser possível analisar a sequência de negociação de mensagens do protocolo IPsec, activa-se uma comunicação entre um dos clientes e a *gateway*. O resultado esperado é a negociação do protocolo IPsec (negociação dos IPsec e *Internet Key Exchange* (IKE) *Security Associations* (SAs) e a utilização do *Encapsulation Security Payload* (ESP) para protecção da comunicação (devido à utilização do *FreeS/Wan*).

A Figura 7.1 apresenta uma captura de negociação IKE IPsec: as tramas 7 a 17 correspondem à troca de informação relativa à negociação do túnel IPSec. As tramas 7 a 14 correspondem à negociação do IKE SA no modo *main mode*; a negociação do IPSec SA no modo *quick mode* corresponde às tramas 15 a 17. Os intervenientes no processo de criação do túnel são o cliente (iniciador) e a *gateway* (receptor). A negociação do túnel IPsec é efectuada em aproximadamente 1.5 segundos. As tramas 23 e seguintes representam a troca de informação protegida pelo túnel IPSec.

Efectua-se agora uma análise mais detalhada a cada uma das tramas de negociação e estabelecimento do túnel. O conteúdo do primeiro pacote *Internet Security Association Key*

Management Protocol (ISAKMP) (trama número 7) está apresentado na Figura 7.2. Este pacote inicia a primeira fase da negociação ISAKMP do tipo *main mode*. Pela análise da figura verifica-se que o protocolo ISAKMP utiliza como protocolo de transporte, o protocolo UDP através da porta 500. O pacote ISAKMP é constituído por um cabeçalho e um SA *payload*. Este transporta uma *Proposal payload*, que por sua vez transporta, no caso específico do *FreeS/Wan*, seis *Transform Payloads* que contêm propostas dos métodos criptográficos a serem utilizados na troca de informação. Verifica-se que, sendo este o primeiro pacote da negociação, o campo *Responder cookie* tem o valor zero, indicando que o receptor ainda não respondeu ao pedido do iniciador. O campo *Exchange type* indica que o iniciador pretende dar início a uma negociação do tipo *Main Mode*; este tipo de negociação tem como finalidade proteger a identidade dos intervenientes.



No.	Time	Source	Destination	Protocol	Info
4	179.980006	Hewlett-_3f:d4:30	01:00:0c:cc:cc:cc	CDP	Cisco Discovery Protocol
5	203.120532	00:0c:41:17:50:b0	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.120.3? Tell 192.168
6	203.120604	D-Link_da:d7:64	00:0c:41:17:50:b0	ARP	192.168.120.3 is at 00:50:ba:da:d7:6
7	203.123499	192.168.120.2	192.168.120.3	ISAKMP	Identity Protection (Main Mode)
8	203.124214	192.168.120.3	192.168.120.2	ISAKMP	Identity Protection (Main Mode)
9	203.186782	192.168.120.2	192.168.120.3	ISAKMP	Identity Protection (Main Mode)
10	203.293276	192.168.120.3	192.168.120.2	ISAKMP	Identity Protection (Main Mode)
11	203.813542	192.168.120.2	192.168.120.3	IP	Fragmented IP protocol (proto=UDP 0x
12	203.816682	192.168.120.2	192.168.120.3	ISAKMP	Identity Protection (Main Mode)
13	204.041266	192.168.120.3	192.168.120.2	IP	Fragmented IP protocol (proto=UDP 0x
14	204.041290	192.168.120.3	192.168.120.2	ISAKMP	Identity Protection (Main Mode)
15	204.130747	192.168.120.2	192.168.120.3	ISAKMP	Quick Mode
16	204.377935	192.168.120.3	192.168.120.2	ISAKMP	Quick Mode
17	204.639245	192.168.120.2	192.168.120.3	ISAKMP	Quick Mode
18	208.123435	D-Link_da:d7:64	00:0c:41:17:50:b0	ARP	Who has 192.168.120.2? Tell 192.168
19	208.126092	00:0c:41:17:50:b0	D-Link_da:d7:64	ARP	192.168.120.2 is at 00:0c:41:17:50:b
20	239.973265	Hewlett-_3f:d4:30	01:00:0c:cc:cc:cc	CDP	Cisco Discovery Protocol
21	289.694661	00:0c:41:17:50:b0	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.120.3? Tell 192.168
22	289.694721	D-Link_da:d7:64	00:0c:41:17:50:b0	ARP	192.168.120.3 is at 00:50:ba:da:d7:6
23	289.697362	192.168.120.2	192.168.120.3	ESP	ESP (SPI=0x48d41f02)
24	289.697698	192.168.120.3	192.168.120.2	ESP	ESP (SPI=0x78087dcd)

Figura 7.1 - Captura da negociação IKE IPSec.

Como já foi referido, o sistema iniciador deve fazer uma proposta ao sistema receptor dos métodos criptográficos a serem utilizados na troca de informação. As seis propostas que constituem a *Proposal payload transforms* encontram-se de forma mais detalhada na Figura 7.3.

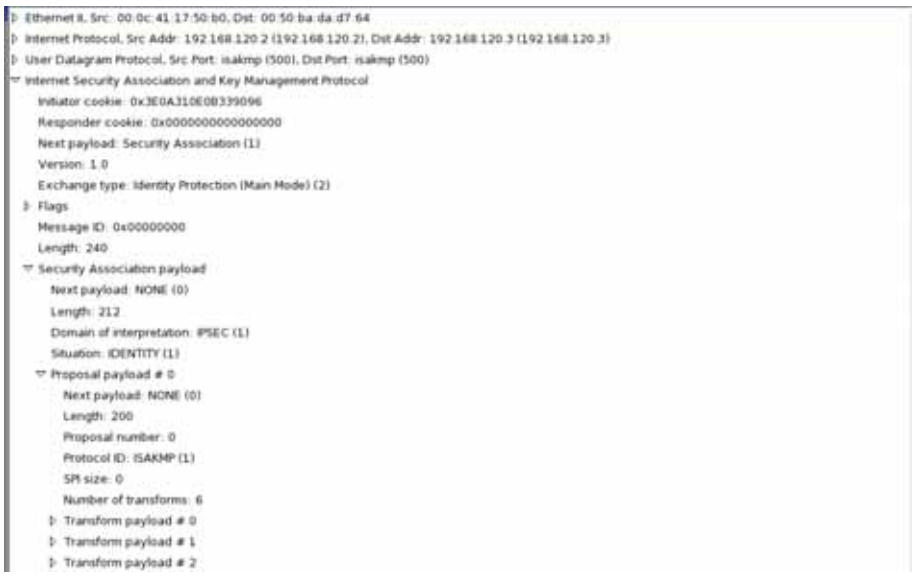


Figura 7.2 - Pacote ISAKMP de início de negociação IKE SA.

Pela análise da Figura 7.3 verifica-se que o algoritmo criptográfico indicado para proteger o IKE SA é o *Triple Data Encryption Security* (3DES), e a função de *hash* a utilizar poderá ser o *Message Digest 5* (MD5) ou o *Security Hash Algorithm* (SHA-1). Como método de autenticação (*Authentication-Method*) é indicado a utilização de certificados digitais (*RSA-SIG*). Outra informação relevante apresentada na Figura 7.3 é o campo *Group-Description*. Este campo indica o grupo pelo qual é realizada a negociação *Diffie-Hellman*. Aos grupos estão atribuídos valores de 1 a 5 e de 14 a 18. Os grupos utilizados para a negociação *Diffie-Hellman* suportados pelo *FreeS/Wan* são o 2 e o 5. O grupo 2 indica que é proposta uma exponenciação modular de 1024 *bits* para a geração de chaves, e o grupo 5 indica a utilização de uma exponenciação modular de 1536 *bits* para a geração de chaves.

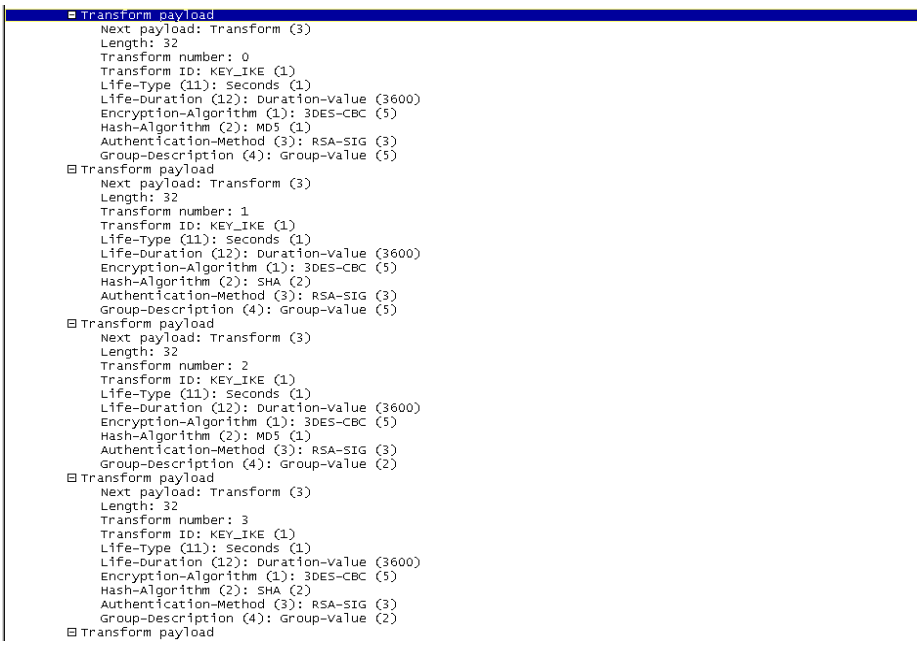


Figura 7.3 - *Transform Payload*.

O segundo pacote ISAKMP (trama 8), apresentado na Figura 7.4, representa a resposta da *gateway* ao pedido do iniciador da negociação IKE SA. Este pacote é semelhante ao primeiro, com diferenças no campo *Responder cookie* e no campo *Proposal payload*. O campo *Responder cookie* apresenta agora um valor atribuído, e o campo *Proposal payload* inclui apenas a *Transform* escolhida para protecção da negociação IKE SA entre o iniciador e o receptor, com algoritmo criptográfico 3DES, função de *hash* MD-5 e método de autenticação *Rivest Shamir Addleman* (RSA) com certificados digitais. A partir deste instante, os valores dos campos *Responder cookie* e *Initiator cookie* não são mais alterados.

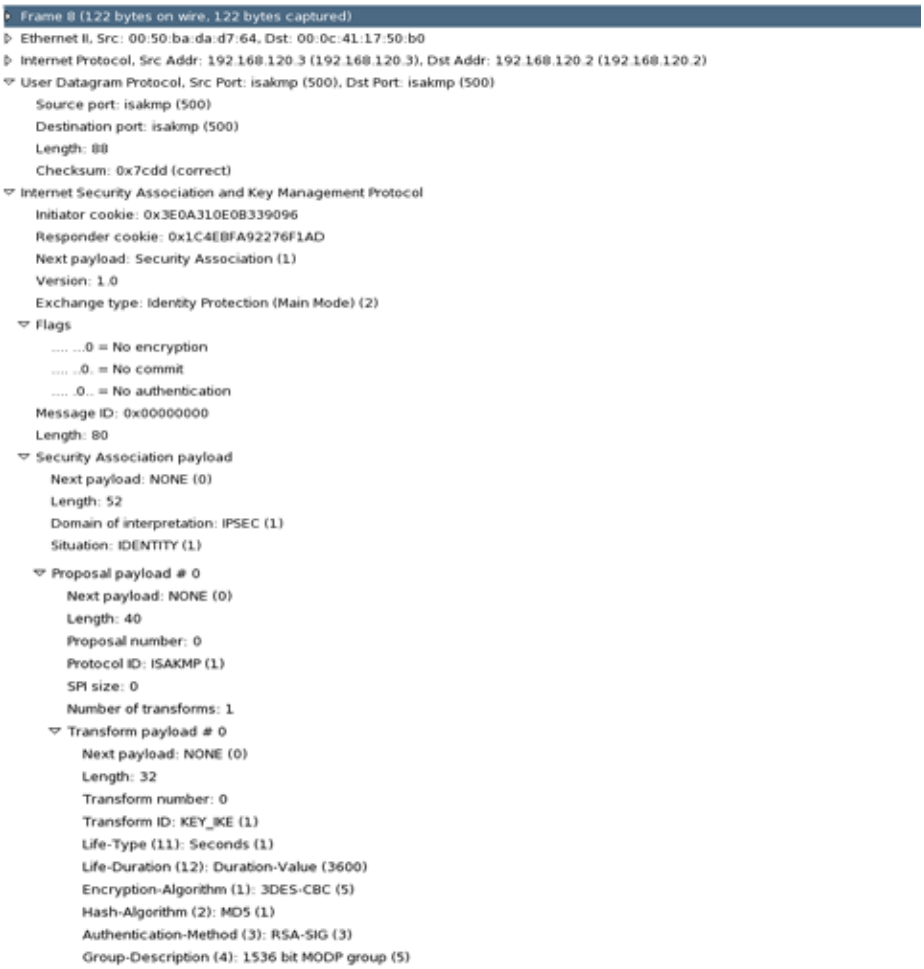


Figura 7.4 - Pacote ISAKMP de resposta, com a escolha da *transform* a utilizar.

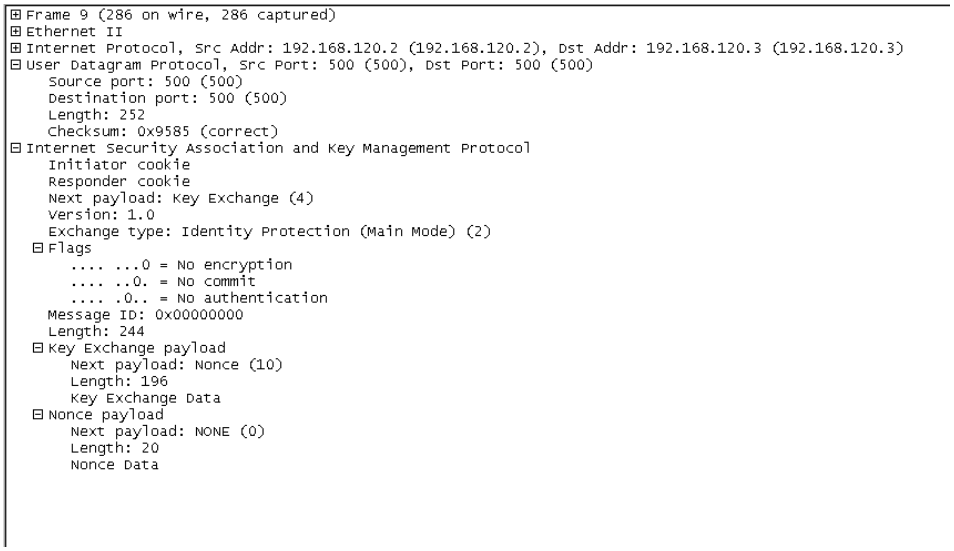


Figura 7.5 - Pacote de troca de informação de chaves públicas do iniciador.

Os dois pacotes seguintes (tramas 9 e 10) correspondem à troca de informação relativa às chaves públicas utilizadas para a cifra dos dados, de cada um dos intervenientes. A Figura 7.5 representa o pacote de troca de informação de chaves pública originado pelo iniciador. Neste pacote, o campo *Key Exchange payload* define a troca de chaves (*Key Exchange*). Este

pacote inclui *nonces* no campo *Next payload*, para evitar ataques de repetição às chaves públicas. Uma particularidade deste pacote é que a informação dos campos *Key Exchange Data* e *Nonce Data* é transmitida sem protecção. Isto pode ser um problema, já que a informação relativa à chave pública a utilizar para a cifra dos dados fica disponível para um atacante.

O pacote de troca de informação de chaves públicas enviado pelo receptor ao iniciador está apresentado na Figura 7.6. O receptor, além de enviar a sua chave e o *nonce*, solicita ao iniciador um certificado digital (*Certificate Request*) para assim o autenticar. O certificado solicitado é do tipo X.509 (*Certificate type*). Após a recepção deste pacote por parte do iniciador, tem início a troca de informação de identificação e a autenticação entre o iniciador e o receptor (mensagens 11 a 14).

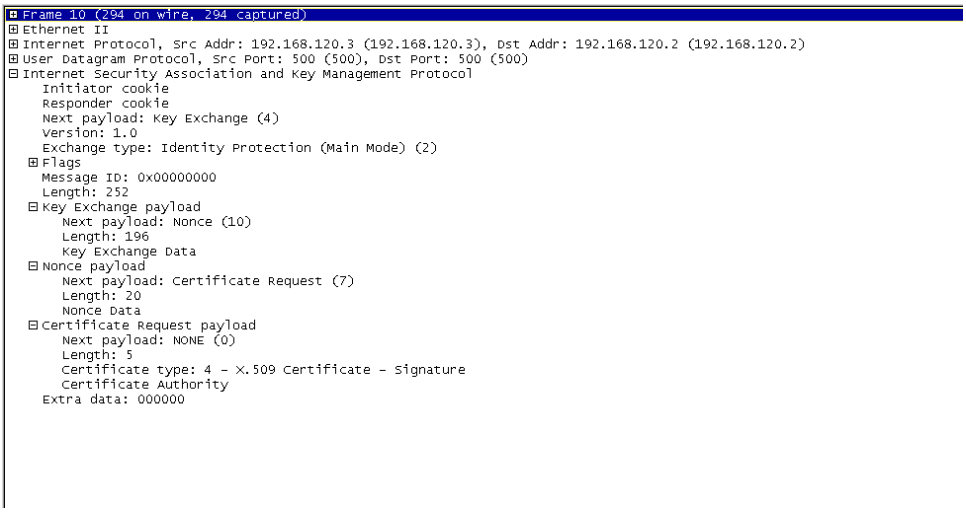


Figura 7.6 - Pacote de troca de informação de chaves públicas do respondente.

A Figura 7.7 mostra o conteúdo de um destes pacotes. O valor do campo *Next payload* indica a troca de informação de identificação. Uma análise ao campo *Flags* mostra que a informação de identificação é cifrada, protegendo-se assim a troca de informação relativa à identificação dos intervenientes. A troca de informação de identificação é toda ela protegida pelo método de cifra e pela chave pública antes negociados. Neste ponto está terminada a fase de *main mode* e está estabelecida a IKE SA.

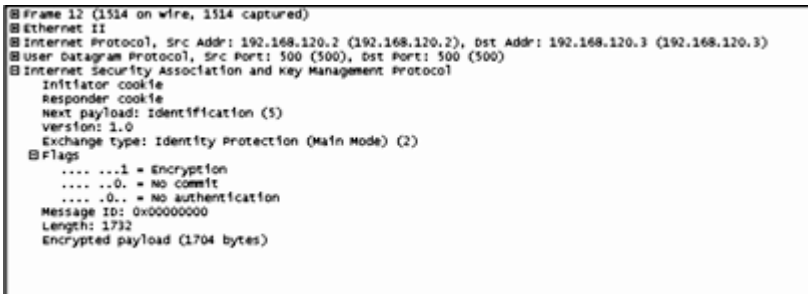


Figura 7.7 - Pacote de troca de informação de identificação.

Após o estabelecimento do IKE SA, inicia-se o processo de obter um IPSec SA. Todo o processo de negociação do IPSec SA recorre sob a protecção do IKE SA. A trama 15 corresponde ao primeiro pacote da segunda fase da negociação IPSec, *quick mode*. A Figura 7.8 apresenta o conteúdo desse pacote, que tem como objectivo iniciar a negociação necessária ao estabelecimento de um IPsec SA. O campo *Next payload* deste pacote indica que a informação que se segue ao cabeçalho ISAKMP, para garantir a sua integridade, contém um código de *hash*. Como a informação que se segue ao cabeçalho ISAKMP está cifrada (valor 1 do campo *Flags*), o analisador de protocolos não consegue analisar o seu conteúdo.

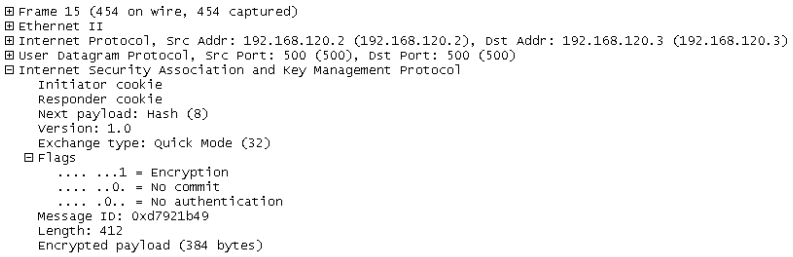


Figura 7.8 - Pacote ISAKMP relativo à fase *quick mode*.

A fase *quick mode* é composta por três pacotes, todos com cabeçalhos iguais entre si, à excepção do campo *Message ID*. Os seus conteúdos são diferentes, mas como são cifrados não é possível efectuar a sua análise. No final da fase *quick mode* o tráfego passa a ser protegido pelo IPsec SA negociado. Após a criação do IPSec SA, tem início a transferência de dados entre o iniciador e o receptor. Nos pacotes de transferência de dados apenas é visível o conteúdo do cabeçalho ESP (Figura 7.9), sendo toda a informação de transferência de dados protegida. A protecção da transferência dos dados é conseguida em modo túnel, devido à utilização da ferramenta *FreeS/Wan*.

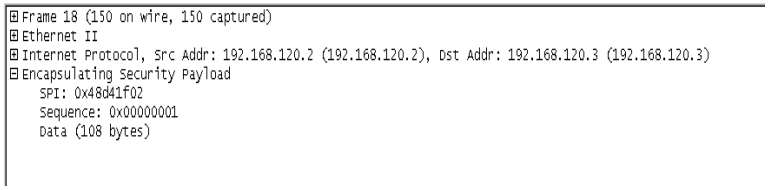


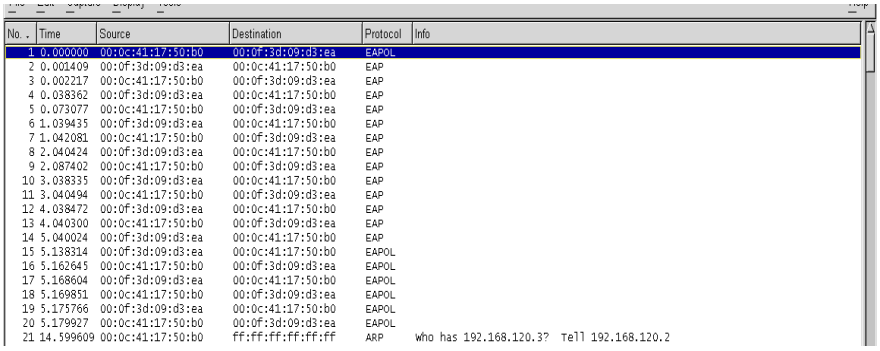
Figura 7.9 - Transferência de informação protegida pelo IPSec.

A negociação do protocolo IPsec aqui analisada corresponde à sua descrição apresentada na secção 5.2.6. Verifica-se que o protocolo IPsec apresenta duas fases de negociação distintas, *main mode e quick mode*. Verifica-se também que, após a fase *main mode*, a negociação efectuada na fase *quick mode* é protegida. Verifica-se ainda que todo o tráfego de dados transferido entre o cliente e a *gateway* é protegido pelo ESP.

7.1.2. Análise ao funcionamento do protocolo WPA-EAP

Na negociação *Wi-fi Protected Access* (WPA)-*Extensible Authentication Protocol* (EAP) são necessários três intervenientes: o cliente WPA (também conhecido por *supplicant*), o autenticador (função desempenhada pelo AP) e o servidor de autenticação (servidor RADIUS). O cenário experimental necessário à análise do funcionamento do protocolo WPA-EAP é idêntico ao ilustrado na Figura 6.5. A única diferença é a instalação e a configuração do analisador de protocolos (*ethereal*) num dos clientes e no servidor RADIUS, permitindo assim obter a sequência de mensagens entre um cliente e o servidor de autenticação. O protocolo de autenticação utilizado é o *Transport Layer Security* (TLS).

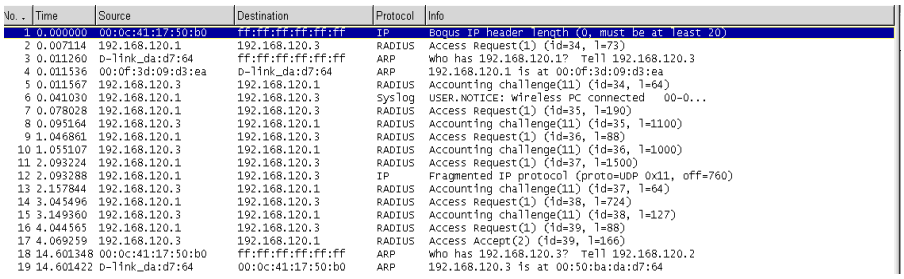
A análise do funcionamento da negociação WPA-EAP é dividida em duas partes, uma que corresponde à troca de mensagens entre o cliente WPA e o AP, onde as mensagens são enviadas utilizando o protocolo EAP, e outra correspondente à troca de mensagens entre o AP e o servidor RADIUS, sendo o protocolo RADIUS o responsável pelas trocas de mensagens. A Figura 7.10 representa toda a troca de mensagens entre o AP e o cliente (pacotes 1 a 20). A Figura 7.11 em contrapartida representa as mensagens necessárias à negociação entre o AP e o servidor (pacotes 2 a 17). Convém referir que apenas o cliente e o servidor suportam a autenticação EAP-TLS; o AP apenas suporta o processo de autenticação 802.1X/EAP.



No.	Time	Source	Destination	Protocol	Info
1	0.000000	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAPOL	
2	0.001409	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
3	0.002217	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
4	0.038362	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
5	0.073077	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
6	1.039435	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
7	1.042081	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
8	2.040424	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
9	2.087402	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
10	3.038335	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
11	3.040494	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
12	4.038472	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
13	4.040300	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAP	
14	5.040024	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAP	
15	5.138314	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAPOL	
16	5.162645	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAPOL	
17	5.168604	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAPOL	
18	5.169851	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAPOL	
19	5.175766	00:0f:3d:09:d3:ea	00:0c:41:17:50:b0	EAPOL	
20	5.179927	00:0c:41:17:50:b0	00:0f:3d:09:d3:ea	EAPOL	
21	14.599609	00:0c:41:17:50:b0	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.120.3? Tell 192.168.120.2

Figura 7.10 - Mensagens WPA-EAP do cliente.

O processo de negociação do protocolo WPA-EAP define duas fases. Numa primeira fase é efectuada a autenticação mútua entre o cliente e o servidor de autenticação (autenticação TLS). Esta fase está definida pelos pacotes 1 a 14 da Figura 7.10 e pelos pacotes 2 a 17 da Figura 7.11. Nesta fase o AP apenas encaminha as mensagens entre o cliente e o servidor, e efectua também o encapsulamento das mensagens EAP em mensagens RADIUS e vice-versa. Na segunda fase do processo são derivadas as chaves utilizadas para proteger a comunicação entre o cliente e o AP. Esta fase está definida pelos pacotes 15 a 20 da Figura 7.10. Esta fase utiliza apenas comunicação entre o cliente e o AP. A negociação das duas fases do protocolo WPA-EAP tem uma duração aproximada de 5 segundos.



No.	Time	Source	Destination	Protocol	Info
1	0.000000	00:0c:41:17:50:b0	ff:ff:ff:ff:ff:ff	IP	Bogus IP header length (0, must be at least 20)
2	0.007114	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=34, l=73)
3	0.011260	D-link_da:d7:64	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.120.1? Tell 192.168.120.3
4	0.011536	00:0f:3d:09:d3:ea	D-link_da:d7:64	ARP	192.168.120.1 is at 00:0f:3d:09:d3:ea
5	0.011567	192.168.120.3	192.168.120.1	RADIUS	Accounting challenge(11) (id=34, l=64)
6	0.041030	192.168.120.1	192.168.120.3	Syslog	USER.NOTICE: Wireless PC connected 00-0...
7	0.078028	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=35, l=190)
8	0.095164	192.168.120.3	192.168.120.1	RADIUS	Accounting challenge(11) (id=35, l=1100)
9	1.046861	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=36, l=88)
10	1.055107	192.168.120.3	192.168.120.1	RADIUS	Accounting challenge(11) (id=36, l=1000)
11	2.093224	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=37, l=1500)
12	2.093288	192.168.120.1	192.168.120.3	IP	Fragmented IP protocol (proto=UDP 0x11, off=760)
13	2.157844	192.168.120.3	192.168.120.1	RADIUS	Accounting challenge(11) (id=37, l=64)
14	3.045496	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=38, l=724)
15	3.149360	192.168.120.3	192.168.120.1	RADIUS	Accounting challenge(11) (id=38, l=127)
16	4.044565	192.168.120.1	192.168.120.3	RADIUS	Access Request(1) (id=39, l=88)
17	4.069259	192.168.120.3	192.168.120.1	RADIUS	Access Accept(2) (id=39, l=166)
18	14.601348	00:0c:41:17:50:b0	ff:ff:ff:ff:ff:ff	ARP	Who has 192.168.120.3? Tell 192.168.120.2
19	14.601422	D-link_da:d7:64	00:0c:41:17:50:b0	ARP	192.168.120.3 is at 00:50:ba:da:d7:64

Figura 7.11 - Mensagens WPA-EAP do servidor.

O cliente inicia o processo de troca de mensagens enviando ao AP um pacote do tipo *EAP over LAN* (EAPoL) (pacote 1 - cliente), pelo facto de ainda não possuir um endereço IP. Este pacote, apresentado na Figura 7.12, é utilizado para indicar ao AP o início de uma autenticação 802.1X, através do campo *Type* com o valor *Start*.

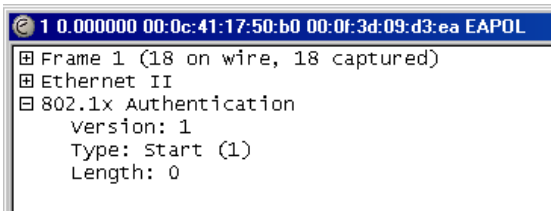


Figura 7.12 - Mensagem EAPoL.

Assim que o AP recebe o pacote anterior solicita ao cliente um pedido de identidade (pacote 2 – cliente). Este pedido é apresentado na Figura 7.13, onde se pode observar que o campo *Code* indica que este pacote é um pedido (*Request*), e que o campo *Type* indica que o tipo de pedido é de identificação (*Identity*).

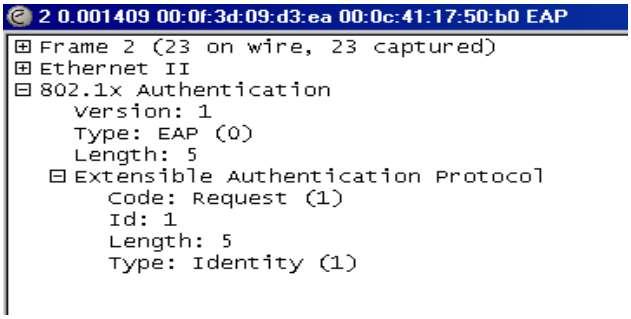


Figura 7.13 - Pedido de Identificação do AP ao cliente.

O cliente responde ao AP com um pacote de resposta (pacote 3 – cliente convertido em pacote 2 – servidor). Este pacote é semelhante ao pacote de *Request*, sendo apenas diferente o valor do campo *Code*, que agora é *Response* (resposta), e é também incluída a identificação do utilizador *beta* (Figura 7.14).

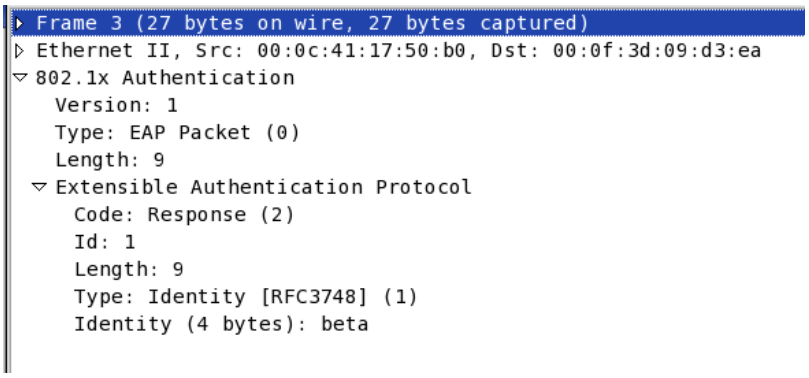


Figura 7.14 - Resposta do cliente ao pedido de identificação do AP.

O AP envia esta mensagem encapsulada numa mensagem RADIUS *Access-Request* (pacote 2 – servidor) ao servidor, pedindo o acesso do utilizador *beta* (Figura 7.15). Esta mensagem inclui um campo que fornece ao servidor o endereço IP que identifica o *Network Access Server* (NAS) que solicita o pedido de autenticação do utilizador (*NAS IP Address*); este campo é apenas utilizado em mensagens RADIUS *Access-Request*. Na mensagem é também incluído um código MD5 (campo *Message Authenticator*) de toda a mensagem. Este código é depois utilizado pelo servidor para verificar a integridade da mensagem recebida. Ao receber a mensagem, o servidor determina o código MD5 da mensagem e compara-o com o recebido; se forem iguais aceita a mensagem; caso contrário elimina a mensagem.

Figura 7.15 - Envio da resposta do cliente pelo AP ao servidor RADIUS.

A partir deste instante inicia-se o processo de autenticação entre o servidor RADIUS e o cliente. Como é utilizado o protocolo TLS, a autenticação é efectuada através de certificados digitais. Neste processo apenas se referem os pacotes originados pelo cliente e o servidor, não esquecendo que o AP efectua sempre a transformação de pacotes RADIUS/EAP em pacotes EAP/RADIUS. Para que seja efectuada a autenticação baseada em TLS, o servidor envia uma mensagem RADIUS de *Access-Challenge* (pacote 5 – servidor convertido em pacote 4 – cliente) ao cliente, indicando que o método a utilizar para a autenticação e protecção dos dados é o TLS, representado pelo campo *Type* da Figura 7.16. Tal como na

mensagem anterior é enviada uma *Message Authenticator* para permitir ao cliente verificar a integridade da mensagem recebida. Inicia-se assim a negociação EAP-TLS.

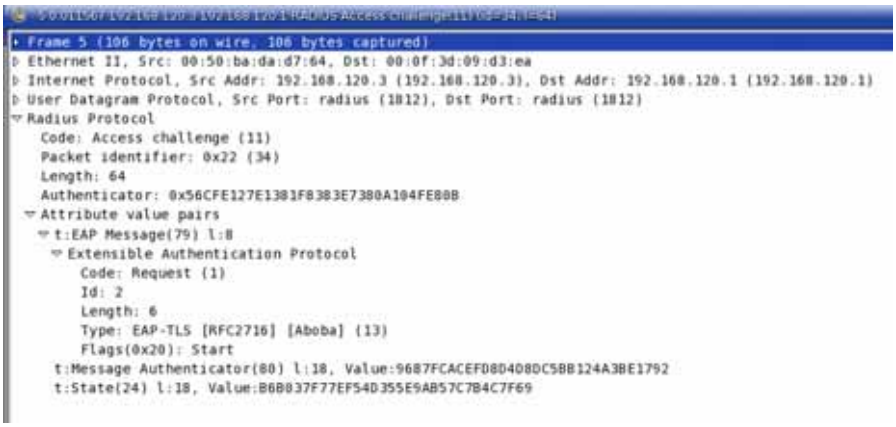


Figura 7.16 - Pedido de autenticação EAP-TLS do servidor ao cliente.

O cliente, depois de receber do AP o pacote de início do processo de autenticação EAP-TLS, responde ao desafio do servidor com um pacote de resposta EAP (pacote 5 – cliente convertido em pacote 7 – servidor), em que o valor do campo *Type* é EAP-TLS. Nesse pacote, representado na Figura 7.17, o campo dos dados encapsula vários registos TLS no formato de camada de registos TLS (TLS *Record Layer*). A camada de registos TLS inclui uma mensagem de *handshake Client Hello*. Esta mensagem informa o servidor que o cliente concorda com o início do processo de autenticação TLS. A mensagem *Client Hello* contém a versão TLS suportada pelo cliente, um número aleatório que mais tarde será utilizado para gerar a PMK, e um conjunto de primitivas criptográficas (*Cipher Suites*) suportadas pelo cliente (num total de 27 primitivas). Em cada grupo de primitivas criptográficas, o cliente indica quais os algoritmos suportados para a troca de chaves, para as assinaturas digitais (algoritmo criptográfico de chave pública), e para a protecção da camada de registos (algoritmo criptográfico de chave simétrica). Indica também o código de *hash* utilizado para verificar a integridade das mensagens. Para a troca de chaves verifica-se que podem ser utilizados os algoritmos RSA e *Diffie-Hellman* (DHE), e para as assinaturas digitais podem ser utilizados os algoritmos RSA, *Digital Signatures Standard* (DSS) e o *EXPORT*. Para proteger a camada de registos, ou seja para cifrar as mensagens trocadas entre o cliente e o servidor, é possível utilizar os algoritmos *Advanced Encryption Standard* (AES) no modo *Counter Block Code* (CBC), o 3DES no modo *Encrypt-Decrypt-Encrypt* (EDE) ou CBC. A integridade das mensagens pode ser verificada utilizando códigos MD5 ou SHA.

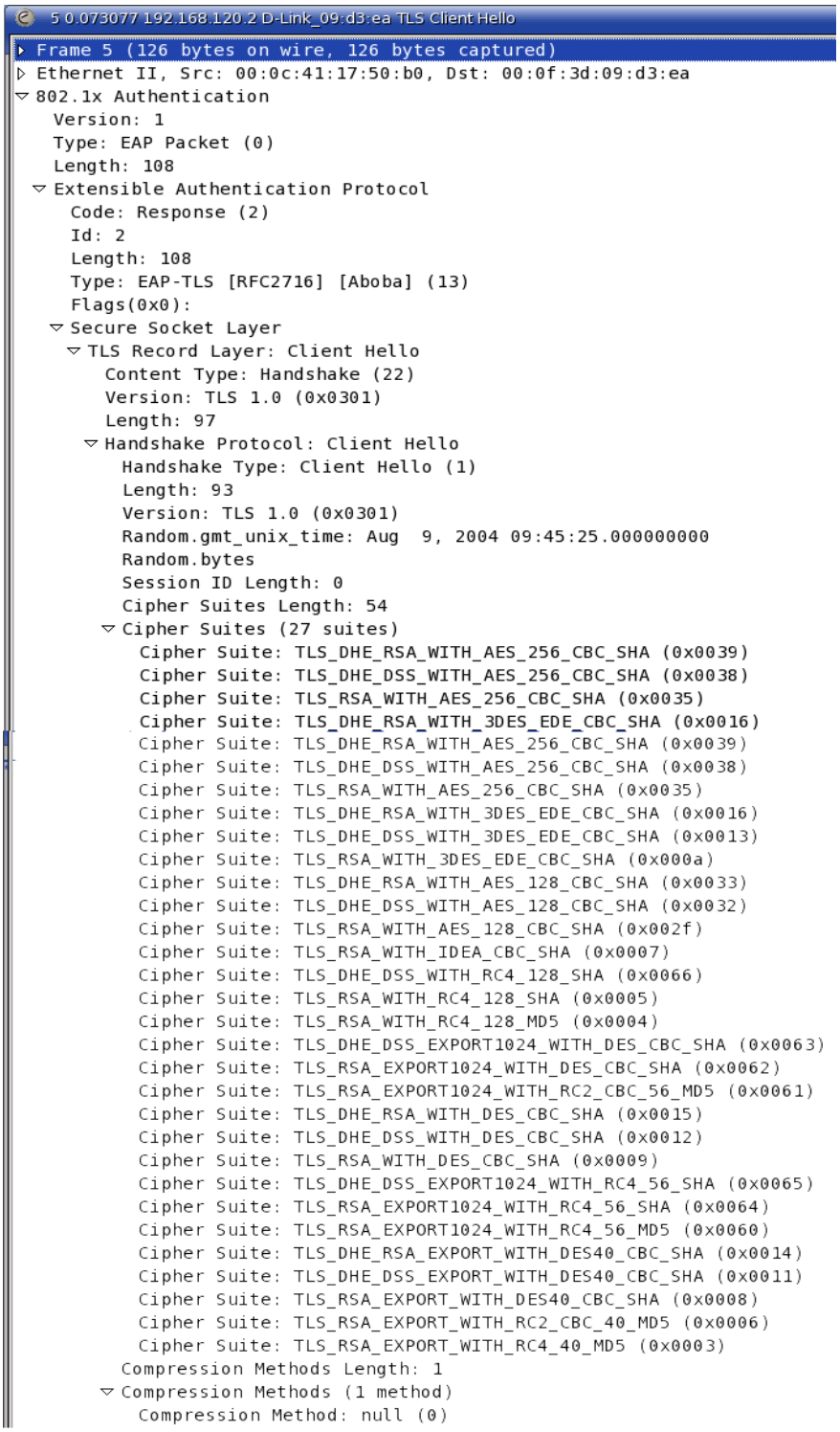


Figura 7.17 - Mensagem do cliente indicando a aceitação do processo de autenticação EAP-TLS.

Neste momento é iniciada a fase da troca de certificados digitais para autenticar a identidade dos intervenientes no processo. Assim que o servidor recebe a mensagem *Client Hello* do cliente, o servidor responde ao cliente com um pacote de pedido EAP (pacote 8 – servidor convertido em pacote 6 – cliente), em que o valor do campo *Type* é EAP-TLS. A Figura 7.18 ilustra o pacote enviado pelo servidor ao cliente. O campo de dados deste pacote encapsula vários registos TLS. Estes registos incluem uma mensagem de *Handshake Server*

Hello, uma mensagem de certificado TLS (campo *Certificate*) e uma mensagem *Server Hello Done.* A mensagem de *Handshake Server Hello* indica a versão do TLS, um número aleatório que será utilizado para gerar a *Pairwaise Master Key* (PMK), um identificador de sessão (*Session ID*), e as primitivas criptográficas suportadas pelo servidor. Como a identificação de sessão enviada pelo cliente era nula ou não reconhecida pelo servidor, o servidor escolhe uma nova identificação de sessão, para assim ser possível estabelecê-la. A primitiva criptográfica indicada pelo servidor é escolhida entre uma das anteriormente enviadas pelo cliente. Como o servidor também pretende autenticar o cliente, envia um pedido de certificado (*Certificate Request*), notificando o cliente que este deverá enviar o seu certificado e assinatura digital. Para o cliente autenticar o servidor é necessário que o cliente conheça a identidade do servidor; para tal, o servidor indica a sua identidade no seu certificado. Como nesta mensagem são enviados vários registos TLS, não sendo possível enviá-los num único pacote RADIUS, é utilizada fragmentação. A fragmentação tem também a vantagem de proteger contra ataques de *Denial of Service* (DoS) e de re-assemblagem. A indicação de fragmentação das mensagens é indicada através do campo *Flags* do EAP. A opção *Length* do campo *Flags* indica o tamanho da mensagem e a opção *More* indica a utilização de fragmentação. Para identificar os fragmentos é utilizado o campo *Id*. O campo *Id* é incrementado por cada fragmento enviado, sendo apenas possível ao servidor incrementar o seu valor.

Antes de enviar outro fragmento o servidor deve esperar pela confirmação de recepção do fragmento por parte do cliente (pacote 7 – cliente convertido em pacote 9 – servidor). A mensagem de resposta do cliente com a confirmação de recepção do fragmento (Figura 7.19), contém apenas o campo *Type* que deverá ser igual a EAP-TLS e o campo *Id* com o valor igual ao *Id* do fragmento recebido. De seguida, o servidor envia o segundo fragmento (pacote 10 – servidor convertido em pacote 8 – cliente).

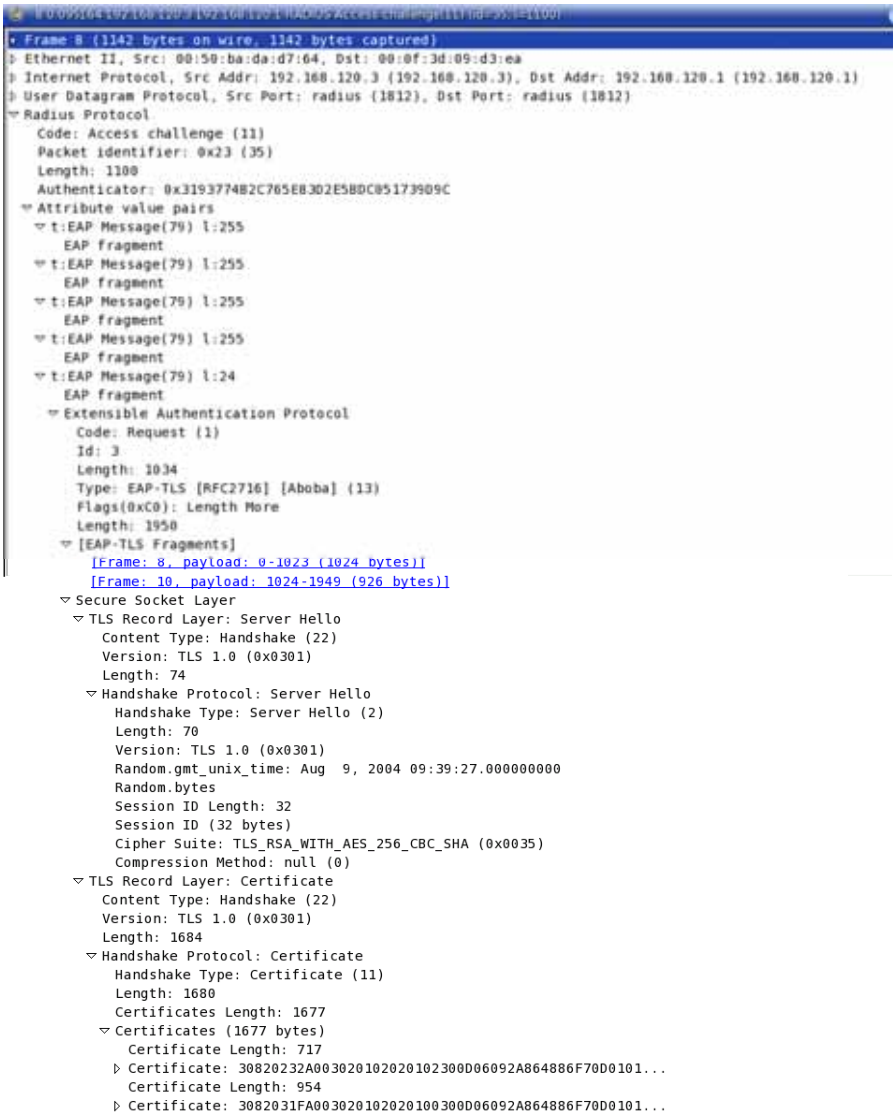


Figura 7.18 - Primeiro fragmento da mensagem de pedido de certificado do servidor ao cliente.

O cliente recebe o certificado do servidor e verifica a validade do mesmo com a chave pública da Autoridade Certificadora (AC). Como o certificado é válido, o cliente responde ao servidor com um pacote de resposta EAP (pacote 9 – cliente convertido em pacote 11 – servidor). O campo de dados deste pacote (Figura 7.20) encapsula vários registos TLS, que contêm uma mensagem TLS *Change Cipher Spec*, uma mensagem de certificado (*Certificate*), uma mensagem *Certificate Verify* e uma mensagem *Client Key Exchange*. A mensagem *Certificate* contém a chave pública do cliente (certificado do cliente). A mensagem *Change Cipher Spec* notifica o servidor que o cliente está pronto a utilizar a nova chave de sessão para cifrar as mensagens. A mensagem *Certificate Verify* é a assinatura digital com a chave privada do cliente das mensagens de *handshake*, o que permite ao servidor autenticar o cliente. O servidor verifica que o cliente conhece a chave privada que corresponde à chave pública do certificado e autentica-o. A mensagem *Client Key Exchange* contém um valor aleatório cifrado com a chave pública do servidor. Este valor aleatório é conhecido de *pre-*

master secret e será utilizado para criar as chaves de sessão. O envio desta informação por parte do cliente é enviada recorrendo à fragmentação.

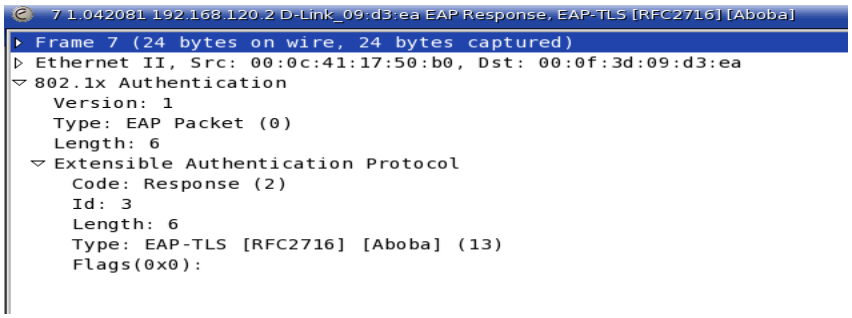


Figura 7.19 - Confirmação por parte do cliente de recepção do fragmento.

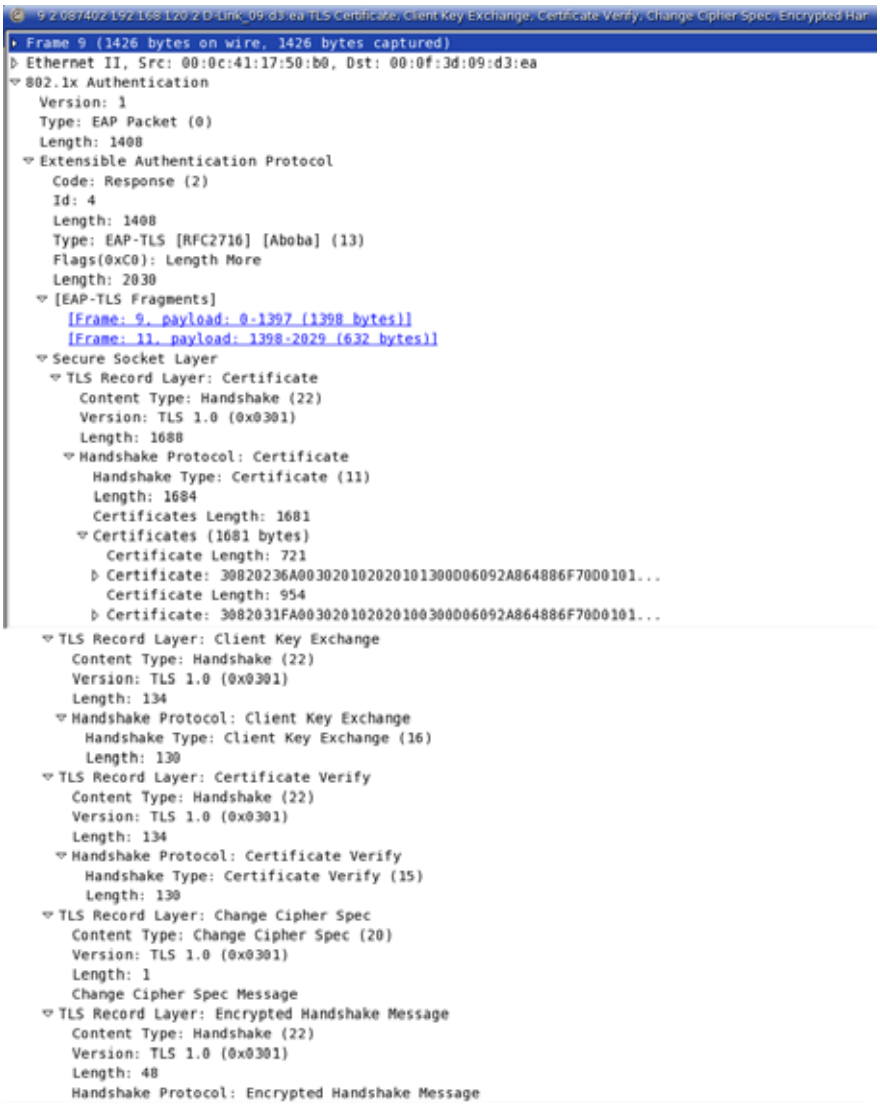


Figura 7.20 - Primeiro fragmento da mensagem do cliente com o seu certificado.

Como o servidor autentica o cliente com sucesso, determina a chave de sessão utilizando os números aleatórios anteriormente trocados e o *pre-master secret*, e envia uma nova mensagem de pedido EAP (pacote 15 – servidor convertido em pacote 12 – cliente). Esta mensagem (Figura 7.21) inclui o campo *Type* com o valor EAP-TLS e dois novos

registos TLS. Um registo contém a mensagem *Change Cipher Spec* e o outro contém mensagens de *handshake* cifradas. As mensagens de *handshake* são constituídas pelo código de *hash*, resultante da utilização da função de *hash* previamente negociada sobre a chave *pre-master secret* e sobre a resposta de autenticação do cliente. A mensagem *Change Cipher Spec* informa o cliente que o servidor também está pronto a utilizar a nova chave de sessão. O servidor utiliza o campo *State* para informar o cliente que, a partir do momento em que a PMK seja obtida, entram num novo estado, o estado de cifra dos dados.

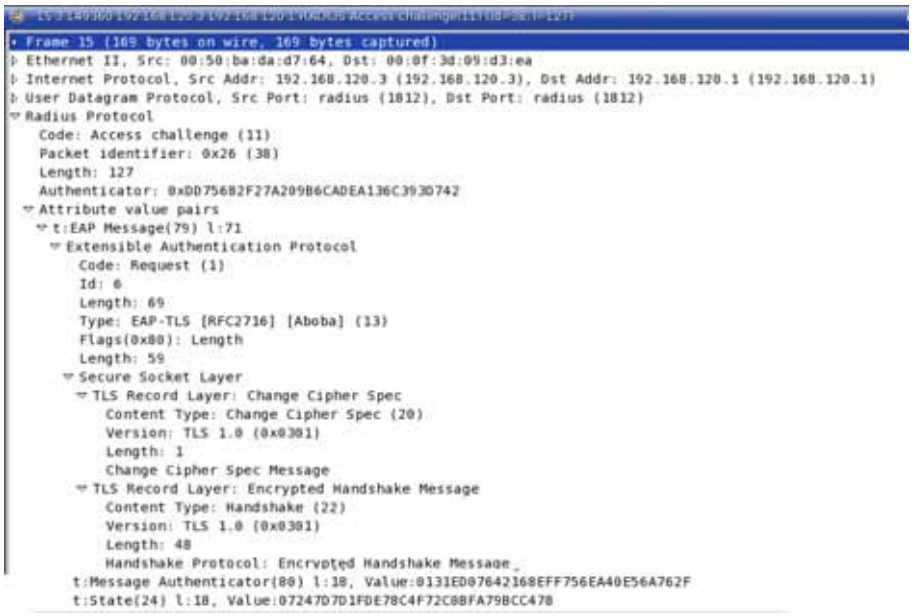


Figura 7.21 - Mensagem enviado pelo servidor ao cliente de definição dos parâmetros de cifra.

Após recepção da mensagem anterior, o cliente calcula um novo código de *hash*, utilizando a função de *hash* negociada. A função tem como entradas a chave *pre-master secret* e a resposta do servidor. O cliente compara o código de *hash* calculado com o recebido na mensagem; se forem iguais o cliente autentica o servidor. Para confirmar ao servidor que o cliente também o autenticou, este envia uma mensagem de resposta EAP (pacote 13 – cliente convertido em pacote 16 – servidor). Esta mensagem (Figura 7.22) não inclui dados, apenas indica no campo *Type* o valor EAP-TLS.

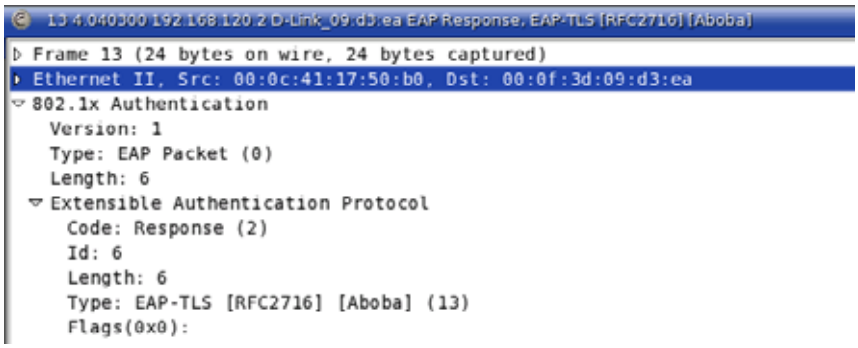


Figura 7.22 - Confirmação de autenticação do servidor por parte do cliente.

Para concluir o processo de negociação TLS, o servidor responde com uma mensagem EAP indicando o sucesso de autenticação (pacote 17 - servidor convertido em pacote 14 – cliente). A Figura 7.23 apresenta esta mensagem de resposta do servidor. Para assinalar o sucesso da autenticação é utilizado o campo *Code* com o valor *Success*. Como o AP não conhece a PMK, o servidor tem que a fornecer, e o seu envio deve ser efectuado de forma segura. O campo *MPPE Recv Key* tem como valor uma chave que permite cifrar os dados enviados para o AP, protegendo assim o envio da PMK do servidor para o AP. O campo MPPE *Send Key* é apenas utilizado em sistemas 802.1X para assinar digitalmente a chave WEP. Neste momento existe um canal seguro entre o cliente e o servidor.

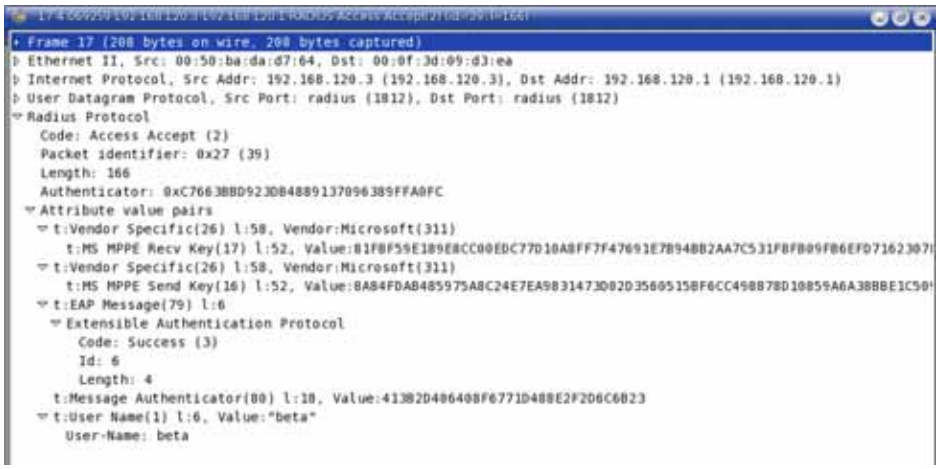


Figura 7.23 - Resposta de autenticação EAP bem sucedida do servidor.

Tendo concluído com sucesso o processo de autenticação EAP-TLS, inicia-se o processo de determinação das chaves WPA, a *Pairwaise Transient Key* (PTK) e a *Group Transient Key* (GTK). Para se gerar a chave WPA PTK, é utilizado o processo *four-way handshake* (ver secção 5.4) que faz uso da chave PMK. O processo *four-way handshake* está representado pelos pacotes 15 a 18 da Figura 7.10. O processo de obtenção da chave WPA GTK está representado pelos pacotes 19 e 20 da mesma figura. A primeira mensagem do processo *four-way handshake* (Figura 7.24) é enviada pelo AP ao cliente (pacote 15 EAPoL). Esta mensagem inclui apenas um *nonce* que será utilizado para gerar a PTK. Como ainda não se conhecem as chaves, os campos *WPA Key MIC* e *Key IV* têm o valor zero. Esta mensagem é enviada sem ser cifrada. Através do campo *Key Information*, o AP informa o cliente que o tipo de chave a calcular é a PTK (*Key Type: Pairwise Key*), que o processo *four-way handshake* ainda não concluiu *(Secure flag: Not Set),* que ainda não existe MIC *(Key MIC flag: Not Set)*, que o AP pretende uma resposta do cliente *(Key Ack flag: Set*), e que ainda não é possível instalar as chaves (*Install flag: Not Set*).

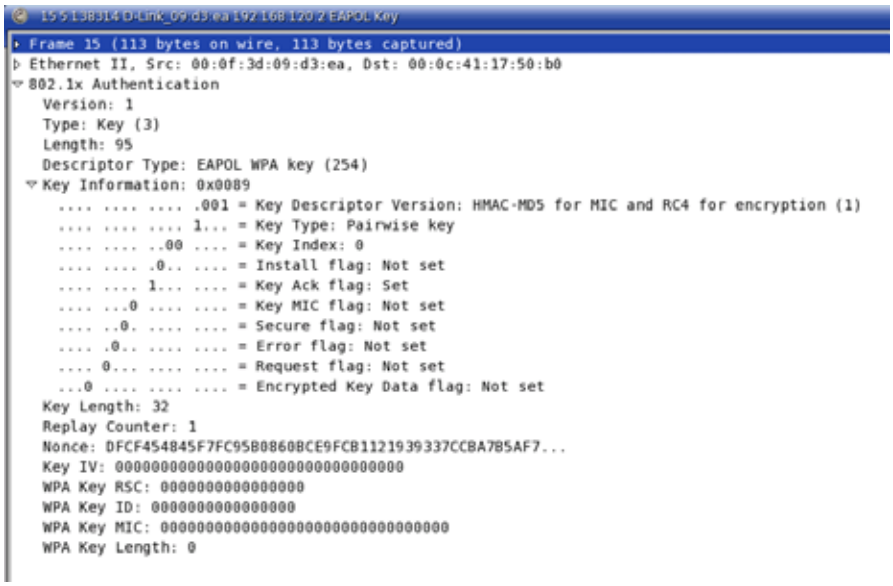


Figura 7.24 - Primeira mensagem do processo *four-way handshake* WPA-EAP, para determinação das chaves PTK.

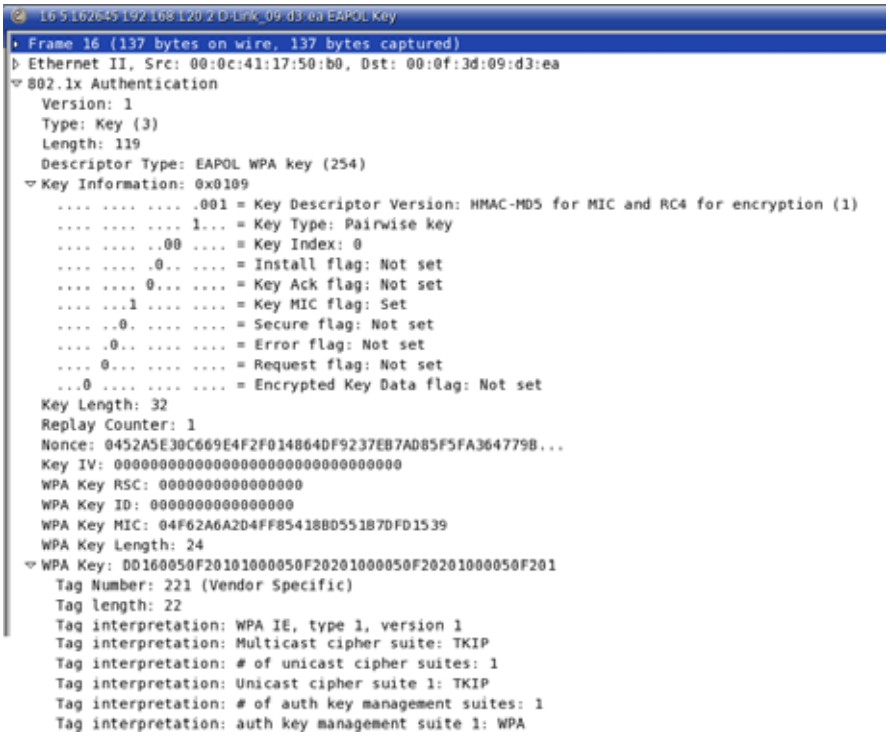


Figura 7.25 - Segunda mensagem do processo *four-way handshake* WPA-EAP.

O cliente responde com uma mensagem EAPoL (pacote 16) com o *nonce* por ele gerado, com a PTK e com o MIC da chave (Figura 7.25). A PTK é obtida pela combinação, através de uma função geradora de pseudoaleatórios dos *nonces* recebidos do AP e gerado pelo cliente, do endereço MAC do cliente e do AP, e da PMK. O valor do campo WPA *Key* MIC é agora diferente de zero, sendo utilizado como prova ao autenticador de que o cliente conhece a PMK. A mensagem indica também, através do campo *Key Information*, que o tipo de chave a negociar é a PTK (*Key Type: Pairwise Key*), que ainda não é possível instalar as

chaves (*Install flag: Not set*), que o processo *four-way handshake* ainda não terminou (*Secure flag: Not set*), e que foi calculado o MIC da chave (*Key MIC flag: Set*). Nesta mensagem, o cliente indica também ao AP que protocolos de gestão de chaves este suporta; neste caso específico é o *Temporal Key Integrity Protocol* (TKIP). Como a mensagem enviada pelo cliente não é cifrada, o AP consegue obter o *nonce* gerado pelo cliente. É então possível ao AP obter, da mesma forma que o cliente, a PTK. O AP calcula também o MIC da chave e compara-o com o MIC recebido; se forem iguais demonstra que a mensagem não foi alterada e que o cliente conhece a PMK. É agora possível instalar e começar a utilizar a PTK. A Figura 7.26 apresenta a resposta do AP ao cliente (pacote 17). Nesta mensagem o AP indica ao cliente que está pronto a utilizar a PTK e que o cliente também a pode utilizar (*Install flag: Set*). Esta mensagem tem também o propósito de indicar que o AP conhece a PMK. Tal como na mensagem anterior, os campos WPA *Key* MIC e WPA *Key* têm valores diferentes de zero. O campo WPA *Key* MIC prova ao cliente que o AP conhece a PMK e a WPA *Key* é a chave PTK. O AP também solicita ao cliente uma resposta (*Key Ack flag: Set*), e informa-o que o processo *four-way handshake* ainda não finalizou (*Secure flag: Not set*).

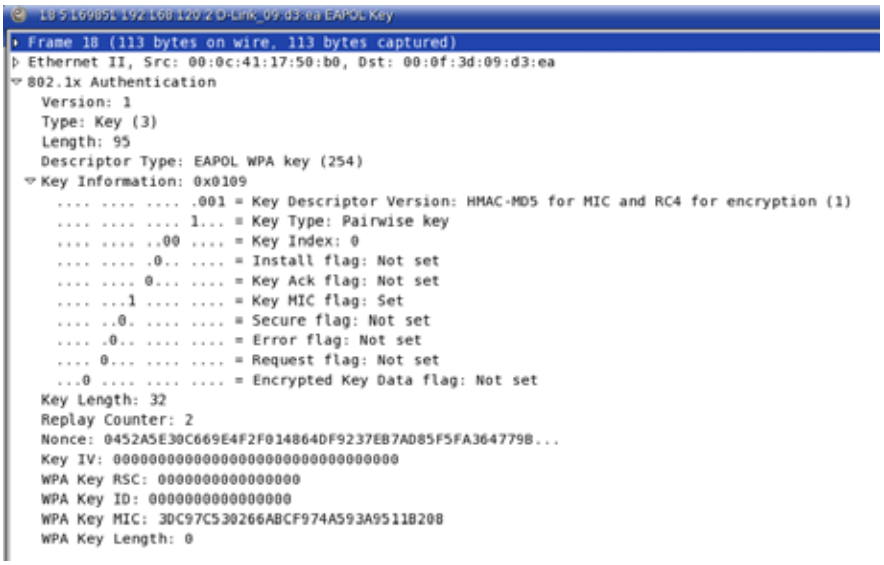


Figura 7.26 - Terceira mensagem do processo *four-way handshake* WPA-EAP.

Como resposta (pacote 18), o cliente envia ao AP a confirmação da instalação da chave PTK (Figura 7.27). Essa mensagem inclui apenas o MIC e um *nonce,* não incluindo a PTK. Como o processo *four-way handshake* só termina depois de o AP receber e decifrar esta mensagem, o cliente sinaliza tal facto (*Secure flag: Not set*).

Depois de recebida a confirmação por parte do cliente que a PTK foi instalada com sucesso, e antes de ser possível ao AP e ao cliente comunicarem de forma segura, é determinada a GTK. A GTK é determinada pelo AP e enviada ao cliente. Esta chave não é mais do que a combinação, através de uma função geradora de pseudoaleatórios, de um *nonce*

escolhido de forma aleatória pelo AP e do seu endereço MAC. A mensagem (pacote 19) que contém a GTK é enviada ao cliente e é cifrada com a PTK anteriormente instalada (Figura 7.28). Através do campo *Key Information*, o AP informa o cliente que a chave a negociar é a GTK (*Key Type: Group Key*), que o processo *four-way handshake* já terminou (*Secure flag: Set*), e que é solicitada ao cliente uma resposta (*Key Ack flag: Set*). Como a GTK é gerada apenas pelo AP, este informa o cliente que já é possível instalar e começar a utilizar essa chave (*Install flag: Set*). A mensagem inclui ainda o MIC da chave e um IV. O MIC permite ao cliente verificar a integridade da mensagem, e o IV é utilizado para cifrar o campo WPA-Key onde é colocada a GTK. A mensagem inclui, apenas para referência, o *nonce* utilizado para calcular a GTK.

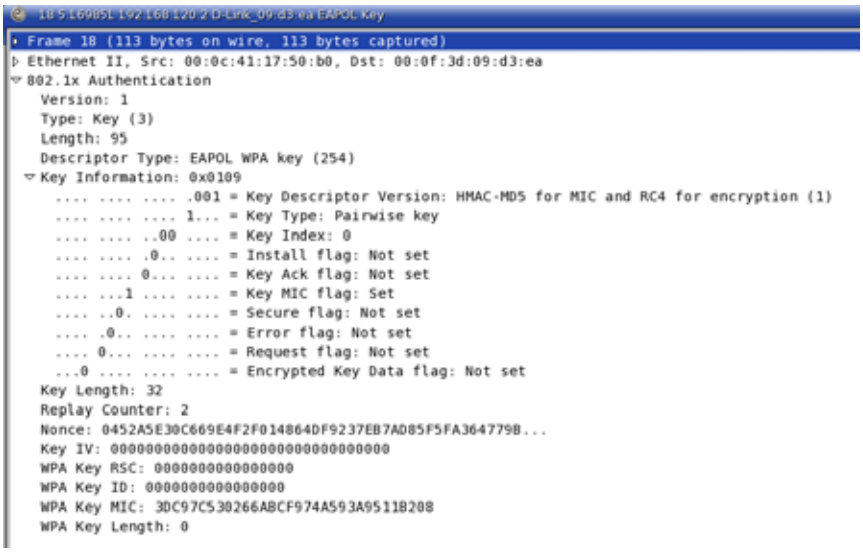


Figura 7.27 - Quarta mensagem do processo *four-way handshake* WPA-EAP.

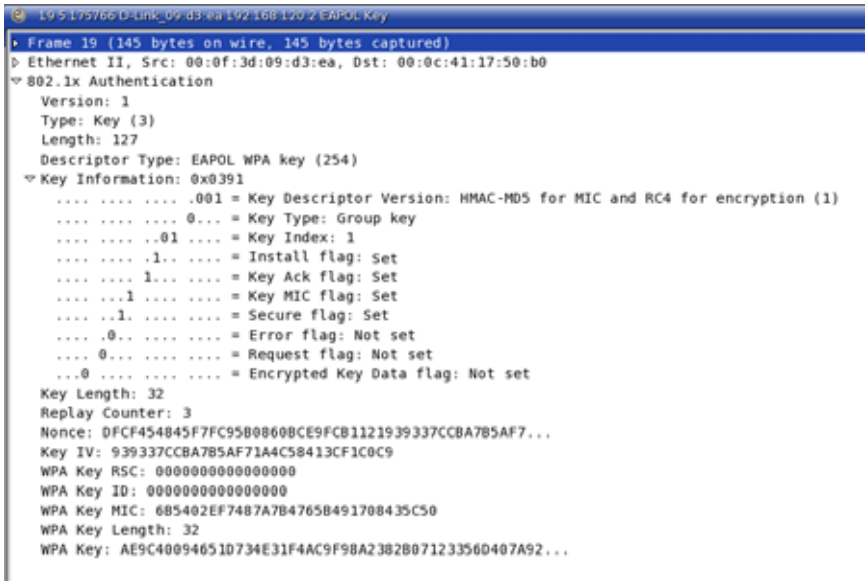


Figura 7.28 - Mensagem com a chave GTK WPA-EAP, enviada pelo AP ao cliente.

Este processo finaliza com o envio de uma mensagem EAPoL (pacote 20) por parte do cliente ao AP; esta mensagem serve apenas para confirmar ao AP a instalação da chave GTK (Figura 7.29). Tal como na confirmação da instalação da PTK, esta mensagem contém apenas o MIC. Finalizada a determinação destas chaves, o AP e o cliente utilizam a PTK e GTK para cifrarem o tráfego de informação gerado (Figura 7.30).

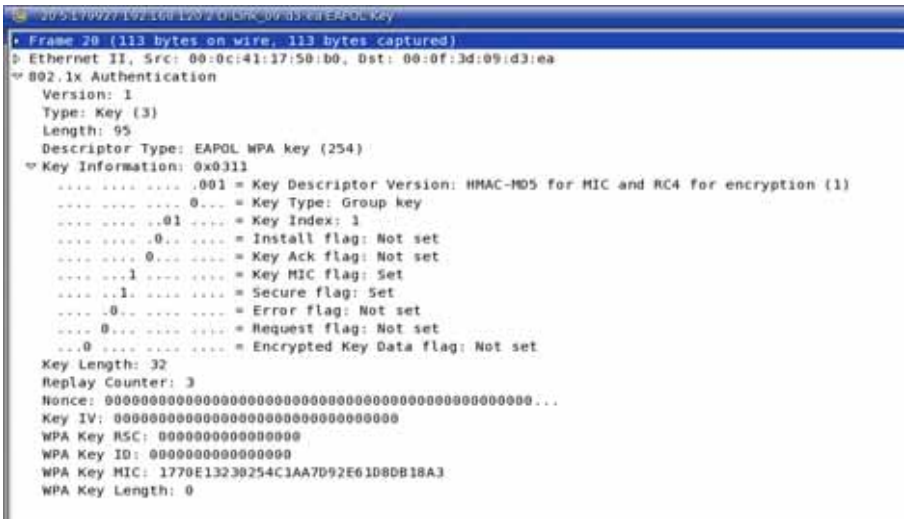


Figura 7.29 - Mensagem do cliente ao AP confirmando a instalação da chave GTK WPA-EAP.

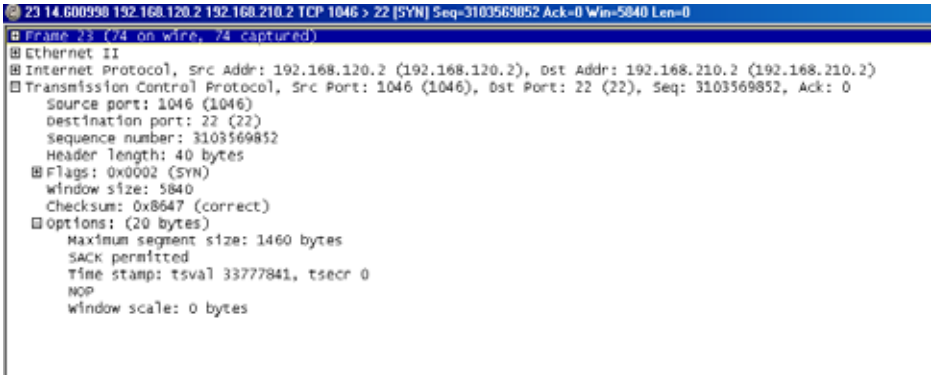


Figura 7.30 - Troca de informação protegida pelo WPA-EAP.

A análise à negociação do protocolo WPA-TLS confirma a especificação normalizada. Comparativamente à negociação IPsec, a negociação WPA-TLS é mais lenta devido ao maior número de intervenientes no processo, e consequentemente, ao aumento do número de mensagens necessárias.

7.1.3. Análise ao funcionamento do protocolo WPA-PSK

Para ser possível analisar a sequência de negociação do protocolo WPA-*Pre Shared Key* (PSK), utilizou-se um cenário experimental semelhante ao ilustrado na Figura 6.10. Para ser possível a visualização das mensagens necessárias à negociação WPA-PSK, instalou-se e configurou-se a ferramenta *ethereal* no cliente.

A principal diferença entra o funcionamento do WPA-PSK e do WPA-EAP é que o primeiro não inclui a fase de autenticação de nível de rede. A autenticação é baseada numa palavra-chave partilhada pelo AP e pelo cliente, sendo apenas efectuado o processo conhecido como *four-way handshake* para determinação das chaves utilizadas para cifrar os dados das comunicações. Tal como no WPA-EAP, no WPA-PSK também se determinam as chaves PTK e a GTK. A chave PMK é previamente configurada no AP e no cliente. A Figura 7.31 mostra as tramas utilizadas no processo *four-way handshake* de determinação das chaves WPA (pacotes 2 a 7), verificando-se que apenas são utilizadas tramas do tipo EAPoL-*key*. A negociação do protocolo WPA-PSK tem uma duração aproximada de 0.2 segundos.

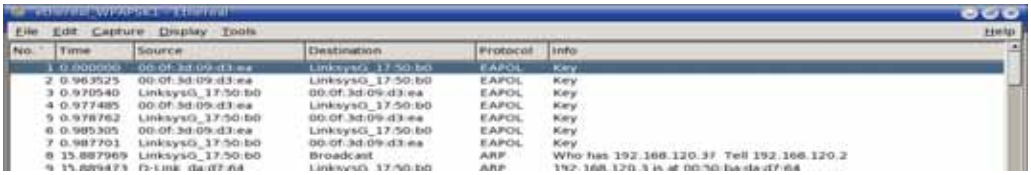


Figura 7.31 - Mensagens WPA-PSK.

O processo de negociação das chaves (processo *four-way handshake*) é idêntico ao processo de negociação de chaves do WPA-EAP. As mensagens apresentam apenas valores diferentes para os campos *Nonce*, *WPA Key*, *MIC*, *Key IV*. Neste processo é negociado como protocolo de cifra o TKIP, sendo o método de autenticação o PSK (Figura 7.32).

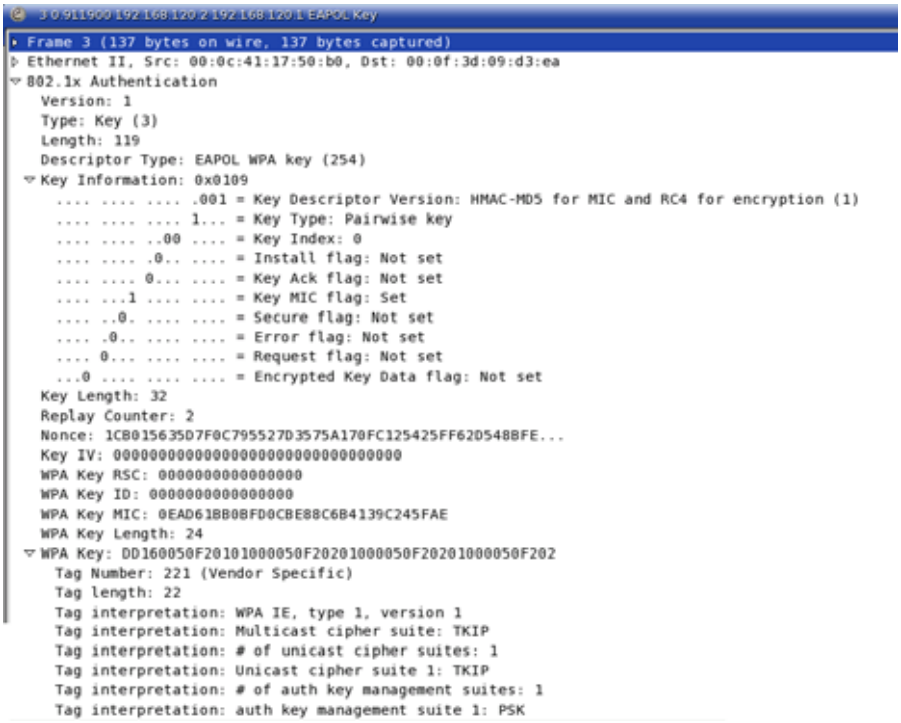


Figura 7.32 - Segunda mensagem do processo *four-way handshake* WPA-PSK, mensagem do cliente para o AP.

Depois de se obter a PTK e a GTK tem início a transferência de informação protegida entre o AP e o cliente. A Figura 7.33 apresenta uma mensagem protegida pelo protocolo WPA-PSK.

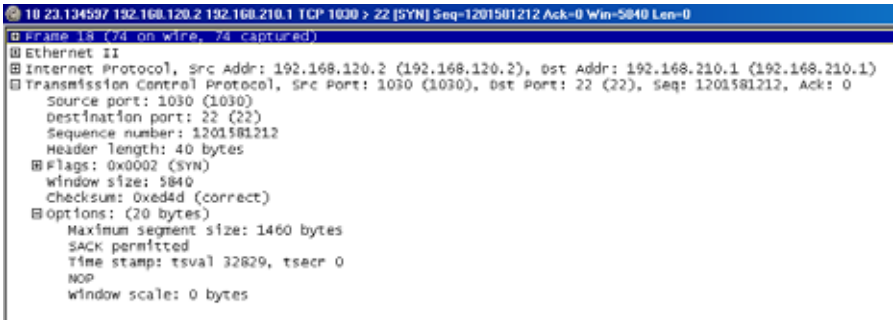


Figura 7.33 - Mensagem protegida pelo WPA-PSK.

A análise ao funcionamento deste protocolo permite verificar que o seu funcionamento é muito mais simples, já que apenas são necessários dois intervenientes. Ao nível do *overhead*, os valores apresentados pelo protocolo WPA-PSK são semelhantes ao do protocolo WPA-EAP e muito inferiores ao do IPsec. O IPsec apresenta mensagens que sofrem grandes alterações e que afectam o tamanho das mesmas, o mesmo não acontecendo nos protocolos WPA, em que as mensagens se mantêm praticamente inalteradas. Devido ao facto de ser um sub-conjunto do WPA-EAP, a negociação do WPA-PSK é mais rápida.

7.2. Ataques aos protocolos de segurança implementados

Nesta secção são realizadas algumas experiências de ataques à segurança da rede implementada com suporte das várias soluções apresentadas, e são analisados os resultados obtidos de eficiência das respectivas soluções.

Os ataques de segurança nesta dissertação são do tipo *Man-In-The-Middle* (MITM), negação de serviço, personificação, repetição e modificação da informação. As ferramentas utilizadas para os implementar são o *ettercap* [20] e o *Cain e Abel* [30]. Estas ferramentas permitem efectuar *arp spoofing*, *ip spoofing* e descodificar palavras-chave. Uma característica destas ferramentas é a de serem muito fáceis de utilizar. Esta facilidade de utilização poderá significar que qualquer utilizador, mesmo com poucos conhecimentos técnicos, pode efectuar ataques de segurança às redes. A ferramenta *Kismet* [19] foi também avaliada para implementar este tipo de ataques. No entanto, apresenta a desvantagem de apenas suportar um conjunto limitado de placas de redes sem fios em modo de monitorização.

Nas secções seguintes apresentam-se, para cada solução implementada, os testes efectuados à rede e os seus resultados.

7.2.1. Resultados IPSec

Para ser possível realizar ataques à segurança da implementação IPSec (devido a ter-se optado pela ferramenta *FreeS/Wan*, apenas se utiliza o protocolo ESP em modo túnel entre o cliente e o servidor RADIUS), é necessário utilizar um cliente da rede a funcionar como intruso. A localização do intruso e o *software* suportado apresentam-se na Figura 7.34. O intruso não dispõe de *software* cliente VPN, apenas tem instalado a ferramenta *ettercap*. A ferramenta *Cain e Abel* é apenas compatível com o sistema operativo *Windows*, e como o cliente a funcionar como intruso utiliza o sistema operativo *Linux*, não foi possível utilizar essa ferramenta.

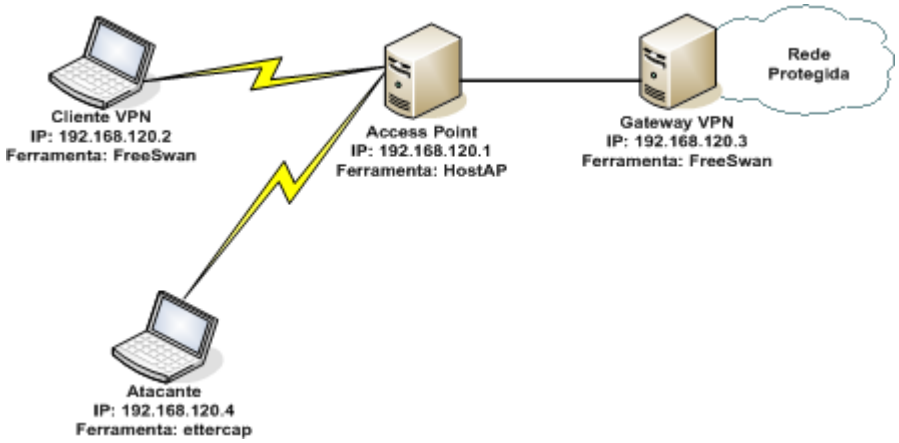


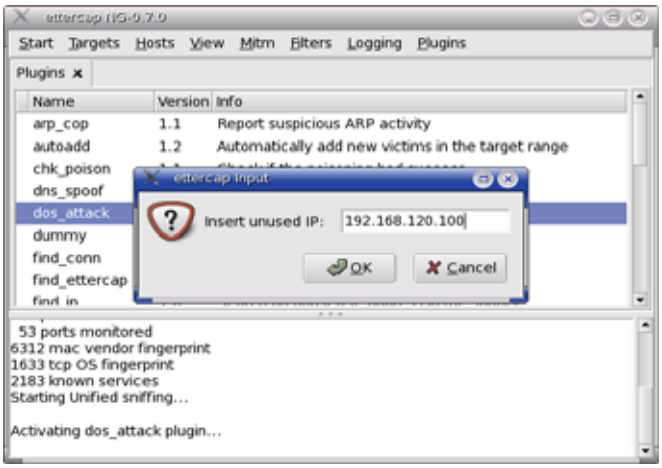
Figura 7.34 - Localização e ferramenta do atacante IPsec.

Para se realizar o ataque de negação de serviço, utiliza-se o *plug-in ISAKMP DoS* do *ettercap*. Os ataques de negação de serviço IPsec consistem em não permitir que o túnel IPsec se active entre um cliente VPN e a *gateway*. Para efectuar estes ataques é necessário (após seleccionar os sistemas aos quais se pretende efectuar ataques, representados pelos endereços TCP/IP da *gateway* VPN e de um cliente VPN), activar no *ettercap* o *plug-in* antes referido. Para se utilizar este *plug-in* tem que se seleccionar no menu *Plugins* do *ettercarp,* o *plugin dos attack*, introduzir o endereço TCP/IP da máquina a atacar (para esta situação o 192.168.20.3) e um endereço IP não utilizado na rede. A Figura 7.35 ilustra os passos necessários à realização do ataque DoS.

Os ataques de DoS apenas têm sucesso antes da criação do túnel *IPsec*; caso o túnel já esteja estabelecido não é possível realizar este tipo de ataque às máquinas intervenientes. A Figura 7.36 ilustra o resultado do ataque de negação de serviço. Este é realizado com sucesso antes da activação do túnel: ao se efectuar o comando que indica o estado do túnel IPsec (*ipsec eroute*) surge a indicação que o túnel está bloqueado (*hold*) e também a indicação que o número de pacotes a circularem no túnel é igual a zero.

a) Activação do *Plugin* DoS.

b) Endereço IP da vítima.



c) Endereço IP falso.

Figura 7.35 - Ataque de DoS IPsec.

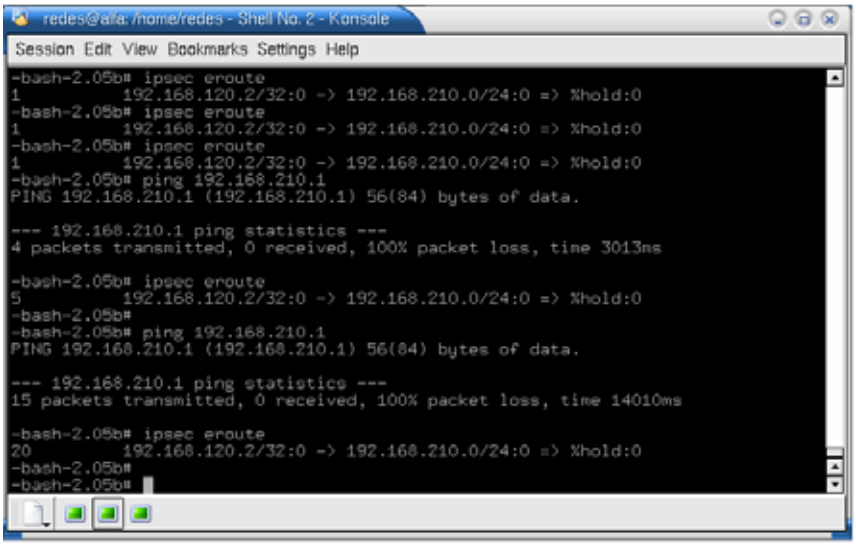


Figura 7.36 - Resultado do ataque de negação de serviço IPsec.

Efectuando o ataque do tipo MITM, utilizando o *arp spoofing*, apenas é possível detectar a activação do protocolo UDP IKE (porta 500) entre o cliente VPN e a *gateway* VPN (Figura 7.37). A partir desse instante a ferramenta apenas detecta como actividade um conjunto de caracteres sem ligação lógica entre eles. Para se efectuar o ataque MITM,

selecciona-se no menu MITM do *ettercap* a opção *arp poisoning*. Surge depois a possibilidade de activar parâmetros adicionais, nos quais se escolhe a opção *Sniff remote connections* (Figura 7.38). A Figura 7.39 mostra que o *ettercap* altera com sucesso a *cache* de arp do cliente VPN, fazendo-lhe crer que tanto a *gateway* VPN como o atacante têm o mesmo endereço MAC. No entanto, verifica-se que o *arp poisoning* não tem qualquer efeito na comunicação, tal como indica a Figura 7.40. Pode-se concluir que o ataque MITM efectuado a uma rede com suporte de segurança IPsec não é bem sucedido.

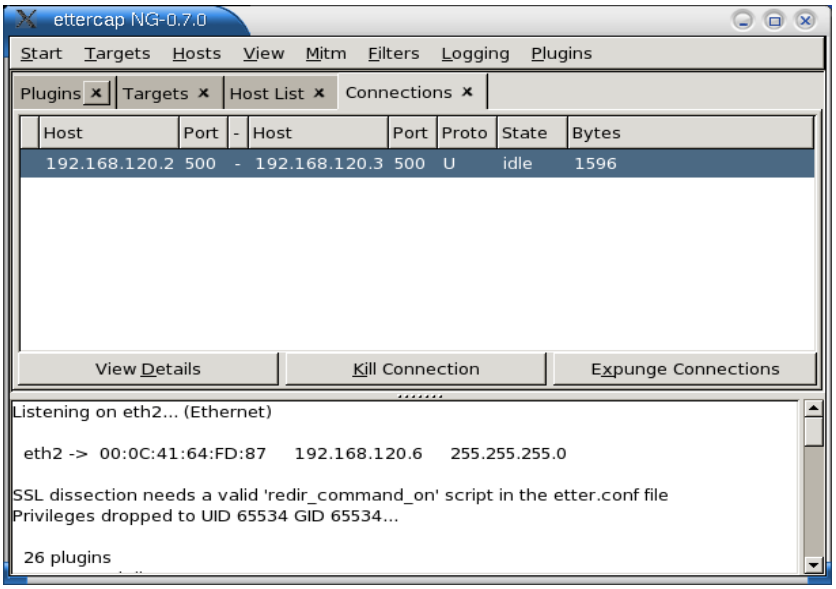


Figura 7.37 - Início do ataque MITM.

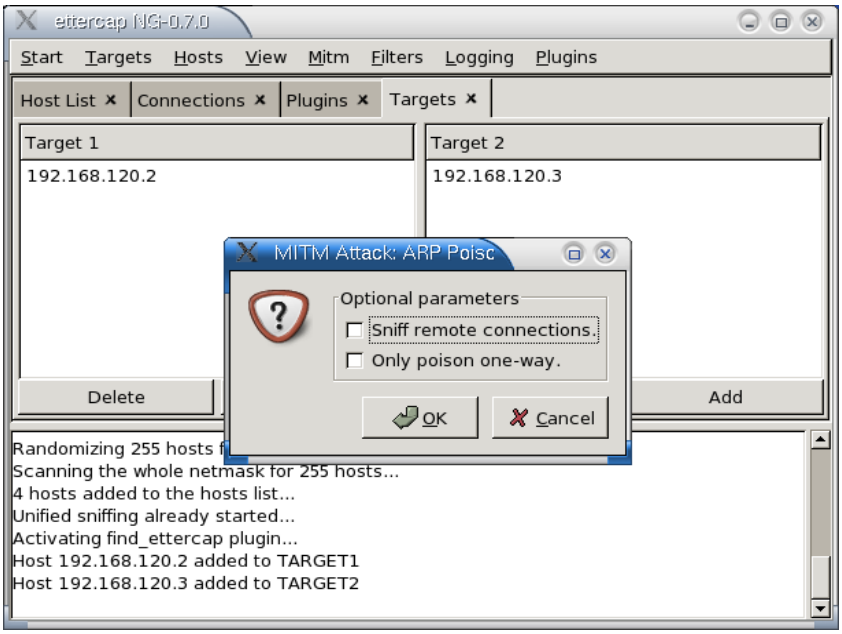


Figura 7.38 - Activação do ataque MITM.

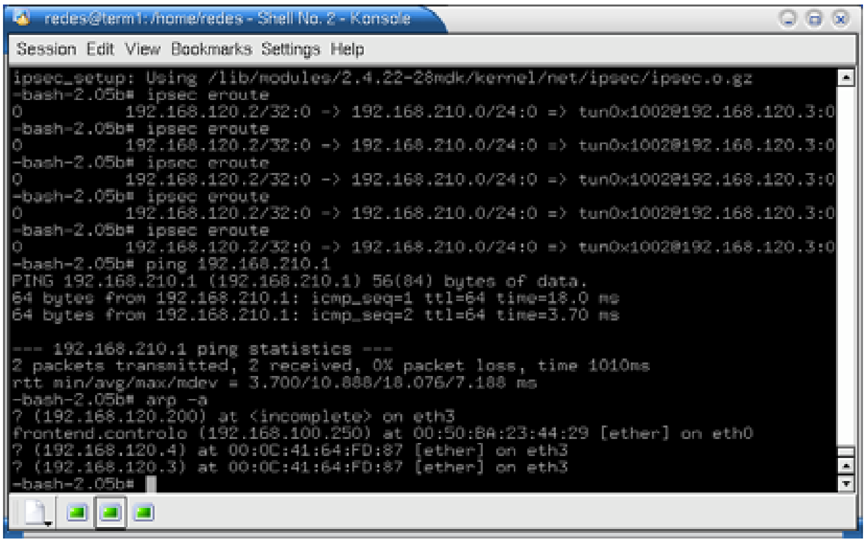


Figura 7.39 - *Cache* de ARP do cliente VPN.

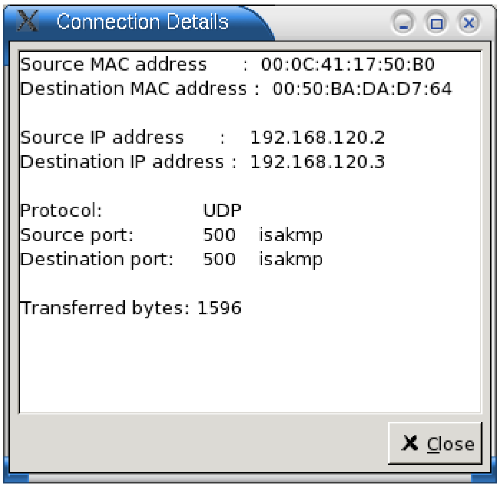


Figura 7.40 - Resultado do ataque *Man-In-The-Middle*.

Os ataques de personificação, modificação e repetição também falharam. Para realizar estes ataques utiliza-se o *ethereal* como ferramenta auxiliar. O *ethereal* permite injectar numa rede pacotes previamente guardados. Ao se injectar na rede um pacote previamente alterado – ataque de modificação – o IPsec não responde ao pedido do pacote, rejeitando-o e eliminando-o. No ataque de repetição, o intruso tenta activar um túnel IPsec. Neste ataque o intruso injecta na rede um pacote previamente capturado e anteriormente utilizado no processo de activação do túnel IPsec; no entanto, não se estabelece nenhum túnel IPsec entre a *gateway* VPN e o intruso.

Para realizar o ataque de personificação o intruso utiliza o mesmo endereço IP e o mesmo *hostname* que o cliente, e tenta desta forma activar um túnel IPsec. Como a troca ISAKMP do IPsec utiliza certificados digitais para autenticação mútua entre o cliente e o servidor, o intruso não é autenticado pelo servidor e não consegue estabelecer um túnel IPsec.

7.2.2. Resultados WPA

Como as implementações WPA-PSK e WPA-EAP utilizam, na parte da rede sem fios, o mesmo mecanismo de protecção, os resultados dos ataques efectuados são apresentados conjuntamente. Para a realização de ataques ao WPA foram instaladas as ferramentas *ettercap* [20] e *Cain e Abel* [30] numa máquina cliente tal como a Figura 7.41 ilustra. O intruso dispõe também do *wpa_supplicant* na sua máquina.

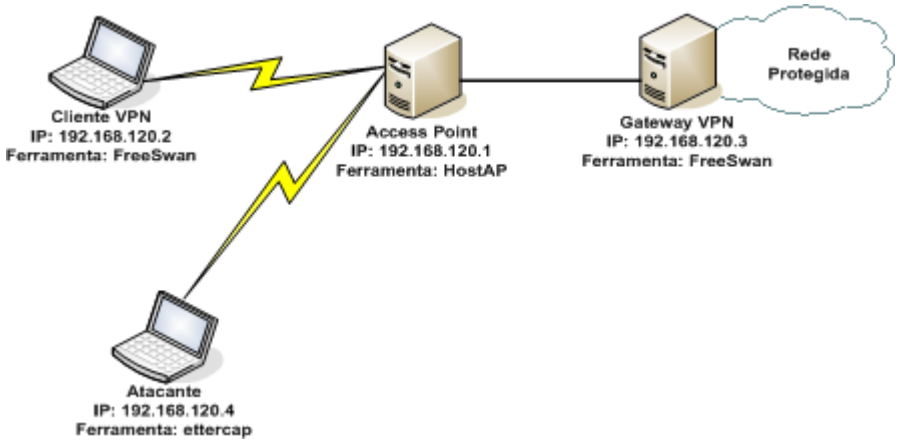


Figura 7.41 - Localização e ferramentas do atacante WPA.

O protocolo WPA é um protocolo de segurança da camada de ligação de dados. As ferramentas utilizadas para realizar os ataques ao WPA funcionam na camada de rede; como consequência não é possível realizar nenhum tipo de ataque ao WPA com estas ferramentas. Através de uma pesquisa exaustiva no sentido de encontrar ferramentas passíveis de implementar ataques de segurança ao WPA, verifica-se que as ferramentas que permitem ataques aos protocolos que funcionam na camada de ligação de dados apenas funcionam no modo *b* (mais restritivo ainda, apenas funcionam para um determinado *chipset* das placas de rede sem fios), limitando a possibilidade de realizar ataques de segurança ao WPA.

No entanto, é possível efectuar com sucesso ataques de negação de serviço ao WPA-EAP e WPA-PSK. Estes ataques foram conseguidos, enviando mensagens falsas de pedido de acesso à rede sem fios. Estas mensagens foram obtidas pela utilização do *software* cliente WPA, o *wpa_supplicant*. O WPA define que, se chegarem ao AP mensagens de pedido de acesso à rede sem a PTK correcta, ele deve bloquear o acesso à rede. Para verificar tal situação, o intruso activa o seu cliente WPA e efectua vários pedidos de acesso à rede. De seguida, activa-se um cliente WPA devidamente configurado para ter acesso aos recursos da rede. Como o AP está no estado bloqueado, o cliente não consegue aceder aos recursos da rede.

Ao WPA-PSK também é possível efectuar ataques passivos de dicionário. Para a realização destes ataques de segurança utilizam-se ferramentas como o *coWPAtty* [52] e

ptcrack [53] que permitem, mediante a sua utilização, determinar a palavra-chave partilhada entre o AP e o cliente em aproximadamente cinco minutos. A utilização destas ferramentas é muito simples e divide-se em duas fases. Numa primeira fase é necessário recolher as mensagens do processo *four-way handshake* (ferramenta *ethereal*); na segunda fase utiliza-se uma das ferramentas (na situação aqui descrita optou-se pela *coWPAtty*) para se obter a chave PMK. Para efectuar este tipo de ataque, e depois de instalada a ferramenta *coWPAtty,* utiliza-se a ferramenta *ethereal* para se obter um registo das mensagens da negociação do processo *four-way handshake* (ficheiro *eap-psk.dump*) do WPA-PSK. Activa-se a seguir a ferramenta *coWPAtty* da seguinte forma:

```
# ./cowpatty -r eap-psk.dump -f dict -s wpa
```

Cerca de 5 minutos após a activação da ferramenta surge a mensagem:

```
The PSK is "thewindinthewillows".
```

Deste modo obtém-se a PMK, sendo agora possível configurar qualquer cliente com essa PMK e aceder à rede sem fios protegida pelo WPA-PSK. Os ataques possíveis de ser efectuados com a utilização da PMK são ataques aos serviços de integridade, confidencialidade, autenticação e não-repúdio. Depois de concluída a configuração do intruso com a PMK, o intruso autentica-se na rede. De seguida, utiliza-se a ferramenta *ethereal*, tentando obter a informação trocada entre um cliente válido e a rede. A Figura 7.42 ilustra a captura de um intruso obtida com a ferramenta *ethereal*. O intruso consegue obter o nome de utilizador (*USER*) e a palavra-chave (*PASS*) de um cliente quando este tenta estabelecer uma sessão com um servidor FTP.

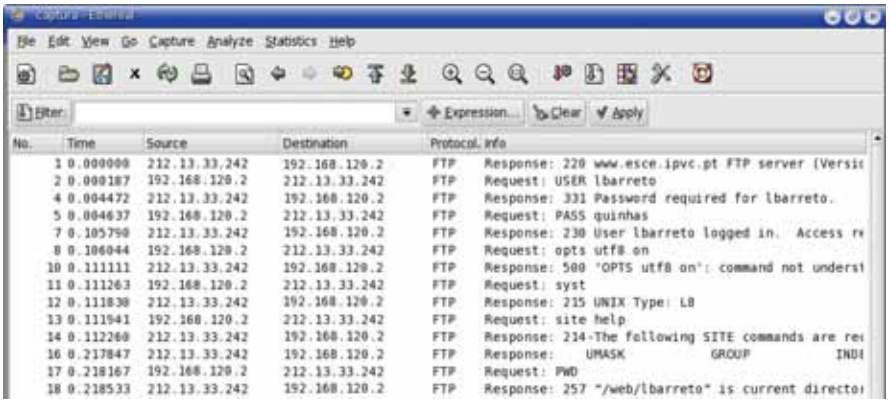


Figura 7.42 -Captura *ethereal* depois de obtida a PMK.

A utilização da ferramenta *ptcrack* processa-se de forma semelhante à utilização da ferramenta *coWPAtty*.

7.3. Testes de desempenho aos protocolos de segurança implementados

Esta secção apresenta testes de desempenho e complexidade das diferentes soluções implementadas. Estes testes dividem-se em duas partes: numa primeira parte apresentam-se testes que permitem avaliar o desempenho da rede; numa segunda fase efectuam-se testes de desempenho aos equipamentos (servidores e clientes). Para a realização dos testes de desempenho da rede utilizam-se as ferramentas *IPERF* [42] e *(C)RUDE* [43]. As ferramentas utilizadas para avaliar o desempenho dos equipamentos são o *sysstat* [44] e o comando do *Linux vmstat*.

Os resultados obtidos com as ferramentas *IPERF* e *(C)RUDE* são apresentadas na secção 7.3.1 e 7.3.2. respectivamente. A secção 7.3.3 apresenta e discute os resultados de desempenho dos equipamentos. Os resultados apresentados representam o valor médio obtido pela realização de 8 experiências.

7.3.1. Resultados *IPERF*

O *IPERF* é uma ferramenta que permite determinar o desempenho de uma rede, tais como a ocupação da largura de banda e variação do atraso da rede. O *IPERF* gera tráfego TCP e UDP, em que uma das máquinas funciona como cliente que gera tráfego para a rede, e a outra funciona como servidor que recolhe os dados e efectua o registo da informação. As estatísticas registadas no modo TCP são o tráfego total entre o cliente e o servidor e a largura de banda média utilizada num determinado período de tempo (por defeito 10 segundos). No modo UDP, além destas estatísticas inclui-se a variação do atraso (*jitter*) e a perda de datagramas.

Para se conseguir uma avaliação eficiente entre as situações implementadas, são criadas as mesmas condições de rede e equipamentos entre todas as implementações. Assim, como o *HostAP* da implementação IPsec apenas funciona em *interfaces* de rede do modo *b*, este é substituído pelo AP da *D-Link*, criando-se assim uma rede IPsec a funcionar em modo *g*, tal como nas implementações WPA. Como a implementação IPsec permite a utilização da opção de compressão de dados nas comunicações, são também realizados testes de avaliação de desempenho com e sem a opção de compressão. Para melhor se compreender qual o custo acrescido ao desempenho da rede pela introdução dos protocolos de segurança implementados, realizam-se também testes de desempenho da rede numa situação em que esta se encontra sem a introdução de nenhum mecanismo de segurança (rede plana). As Figura 7.43 e Figura 7.44 indicam quais os equipamentos utilizados para os testes de desempenho

IPERF, bem como a localização do cliente *IPERF* e do servidor *IPERF* para as situações de rede plana (sem a introdução de nenhum mecanismo de segurança), IPsec, WPA-EAP e WPA-PSK.

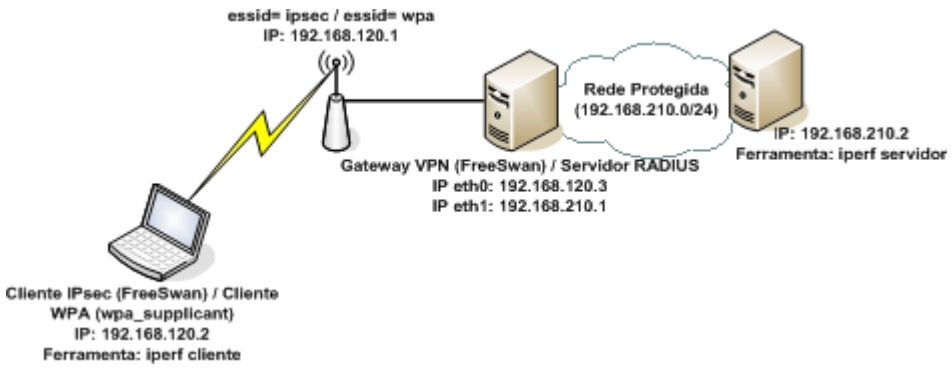


Figura 7.43 - Utilização IPERF implementação IPsec /WPA-EAP.

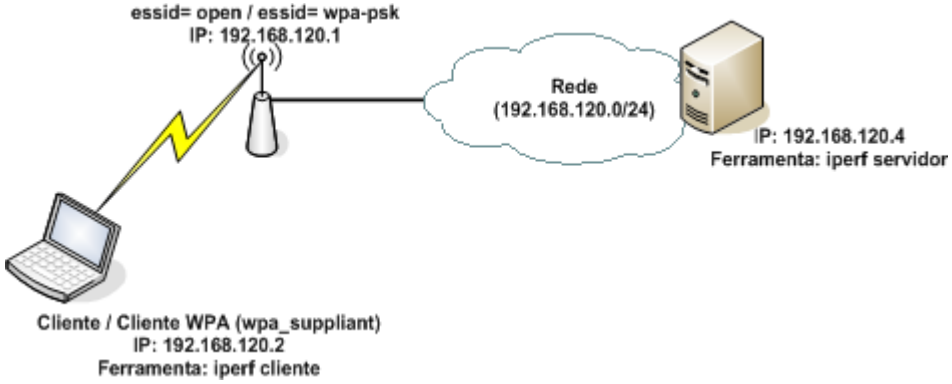


Figura 7.44 - Utilização IPERF implementação rede plana/ WPA-PSK.

Os primeiros testes de avaliação do desempenho da rede consistem na determinação do *throughput* (quantidade de informação enviada, em *bytes*, num determinado intervalo de tempo). O primeiro fluxo de tráfego TCP gerado tem uma janela de 85.3 *Kbytes* (este valor é definido por defeito pelo *IPERF*), o que significa que por cada 85.3 *Kbytes* enviados, o receptor deve confirmar a recepção dos mesmos, enviando um pacote de confirmação. Todos os fluxos aqui utilizados têm uma duração de 20 segundos. O fluxo é gerado após a negociação dos diferentes protocolos (IPSec, WPA-EAP e WPA-PSK) ter concluído com êxito. O gráfico da Figura 7.45 mostra o resultado obtido pelo *IPERF* para esse fluxo de tráfego. Os resultados presentes no gráfico permitem concluir que o IPsec é o mecanismo com menor *throughput* e informação enviada, enquanto que as implementações WPA apresentam um desempenho mais próximo do sistema plano, sendo o desempenho do sistema WPA-EAP um pouco melhor que o do WPA-PSK. Os resultados obtidos devem-se ao facto de, para criar um túnel que proteja os dados, o IPsec cifra todos os pacotes relativos a um determinado fluxo de dados, encapsula todo o pacote IP original, e introduz novos campos específicos do IPsec (cabeçalho IPsec ESP, campo de autenticação IPsec). As implementações WPA apenas

cifram com uma determinada chave de cifra o pacote IP e acrescentam-lhe um IV, não alterando assim significativamente o tamanho nem o formato do mesmo. Outro factor que influencia o desempenho do sistema WPA é o facto de utilizar um algoritmo de cifra, nomeadamente o RC4, que é de processamento mais fácil do que o utilizado pelos sistemas IPsec. Como o processo IPsec altera e aumenta de forma significativa o pacote IP, no mesmo espaço de tempo consegue enviar um menor número de *bytes.* Para este tamanho da janela, o desempenho da implementação IPsec com compressão e sem compressão é semelhante. Este resultado deve-se ao facto de, para este tamanho da janela, a taxa de compressão conseguida ser muito pequena mantendo-se o tamanho das mensagens praticamente inalterado. Relativamente às implementações WPA, o melhor desempenho da implementação WPA-EAP pode estar relacionado com a forma de funcionamento do protocolo TCP. Sempre que ocorrer uma perda de pacotes, o TCP interpreta esta indicação como congestionamento da rede e diminui a taxa de transmissão da informação. As perdas de pacotes podem ser originadas pela falta de capacidade do *buffer* do AP em armazenar todos os pacotes recebidos, pela impossibilidade deste processar todos os pacotes e pelo maior número de pacotes a circular na rede. Como os resultados das implementações WPA são obtidos para janelas TCP e pacotes de igual tamanho, e como todo o processo de protecção de dados (cifra e decifra) nas duas implementações é exactamente igual, o melhor desempenho da implementação WPA-EAP poderá ser consequência, para aquele momento, das condições da rede.

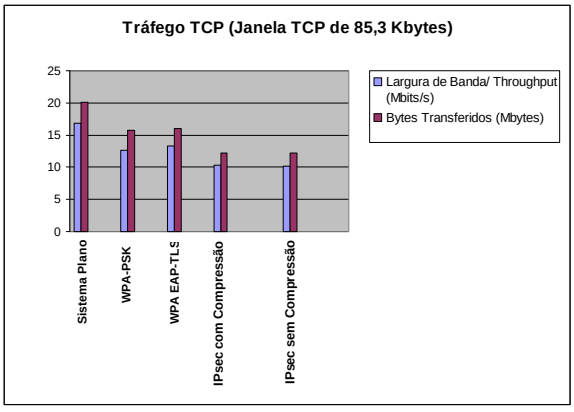
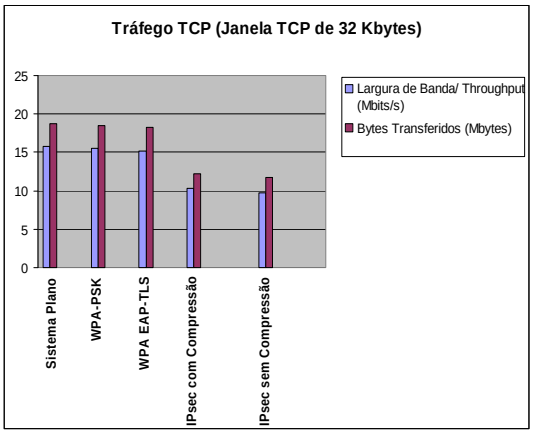
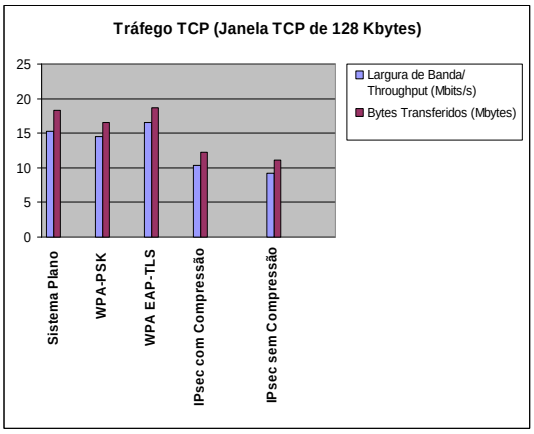


Figura 7.45 - *Troughput* e informação enviada – fluxo TCP de 85,3 *Kbytes*.

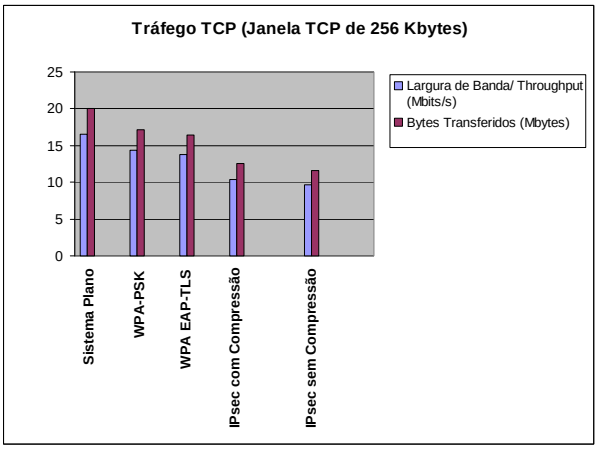
A seguir realizaram-se as mesmas experiências alterando-se o valor da janela do tráfego TCP para valores de 32, 128 e 256 *Kbytes*. A Figura 7.46 apresenta o resultado do *throughput* e *bytes* enviados para cada um dos fluxos de tráfego anteriormente referidos.



a) Janela TCP de 32 *Kbytes.*



b) Janela TCP de 128 *Kbytes.*



c) Janela TCP de 256 *Kbytes.*

Figura 7.46 - *Troughput* e informação enviada

A análise aos gráficos da Figura 7.46 permite concluir que o protocolo IPsec, quer na sua variante com compressão quer na variante sem compressão, é aquele que introduz sempre mais custos de desempenho na rede, o que vem de encontro aos resultados apresentados na Figura 7.45. Uma outra particularidade da observação dos gráficos é que para um fluxo de tráfego TCP com uma janela de 128 *Kbytes* o desempenho da situação WPA-EAP é maior que a do sistema plano. No sistema plano, o cliente não efectua nenhuma alteração aos pacotes a enviar, o que significa que no mesmo intervalo de tempo pode enviar um maior número de mensagens, preenchendo assim, mais rapidamente a janela TCP. Como é possível enviar um maior número de mensagens, e como as redes sem fios são redes do tipo *broadcast*, existe a possibilidade de ocorrer um maior número de colisões. O maior número de colisões implica uma maior perda de pacotes. O TCP interpreta esta perda de pacotes como congestionamento da rede, e diminui a taxa de transmissão da informação. Este é um dos motivos com influência na diminuição do desempenho do sistema. Como são utilizadas janelas de fluxo de tráfego relativamente pequenas, e mensagens de tamanho relativamente pequeno e constante, o mecanismo de compressão utilizado pelo sistema IPsec não acrescenta

melhorias significativas ao nível do desempenho: a compressão é tanto mais difícil quanto mais pequeno for o tamanho da mensagem a enviar (é mais difícil encontrar uma sequência com *bits* repetidos numa mensagem de tamanho pequeno). Os sistemas IPsec, relativamente aos restantes sistemas, apresentam sempre o menor desempenho da rede.

Para simular a utilização da rede por aplicações diferentes, definem-se tamanhos de mensagens com valores compreendidos entre 8 e 4096 *bytes*. Os resultados obtidos encontram-se apresentados na Figura 7.47.

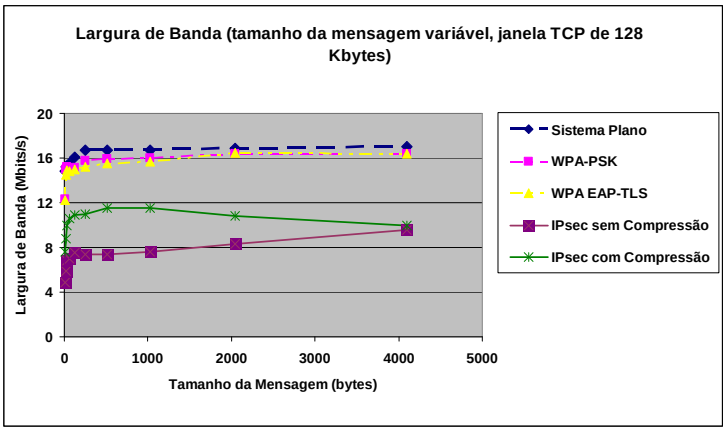
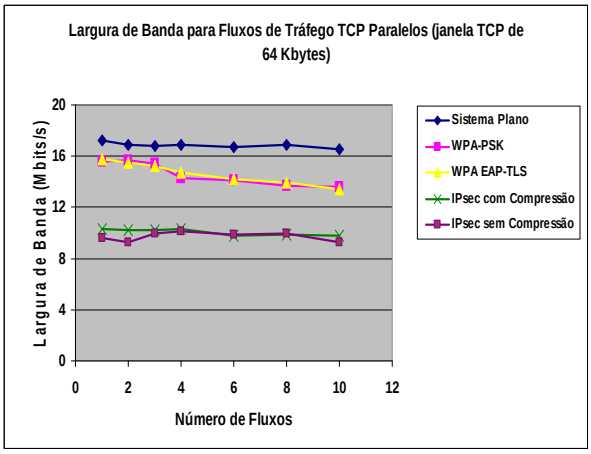


Figura 7.47 - *Throughput* para mensagens de tamanho variável

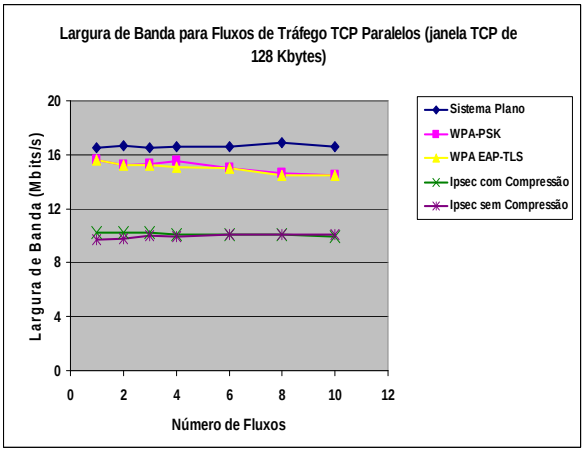
Verifica-se que, para os diversos valores de tamanho das mensagens, as implementações WPA apresentam melhor desempenho que as implementações IPsec, em que o ganho é de cerca de 5 *Mbits/s*. O desempenho da implementação IPsec com compressão, à medida que o tamanho das mensagens aumenta, aproxima-se do desempenho da implementação IPsec sem compressão. Tal situação deve-se ao facto de, quanto maior a mensagem, mais fácil será encontrar um padrão que permita comprimir de forma significativa a mensagem, e assim poder enviá-la de forma mais rápida. Com o aumento do tamanho das mensagens, as implementações WPA apresentam um comportamento estável, já que estes sistemas não alteram de forma significativa o tamanho das mensagens. Também se pode verificar que o desempenho das implementações WPA é melhor para mensagens de tamanho superior a 2000 *bytes*. Este resultado está relacionado com o facto de, sendo o tamanho da janela TCP um valor fixo, quanto maior o tamanho das mensagens, menor o número de mensagens a enviar para a mesma janela TCP, o que reduz o número de colisões, aumentando o desempenho da rede.

Para simular a utilização da rede por vários clientes em simultâneo e a utilização de diversas aplicações, utilizaram-se fluxos de tráfego TCP com janelas de 64, 128 e 256 *kbytes*, com 1 a 10 fluxos de tráfego em simultâneo. Esta experiência tem a duração de 10 segundos para cada fluxo. A Figura 7.48 apresenta os resultados obtidos. A análise às figuras permite

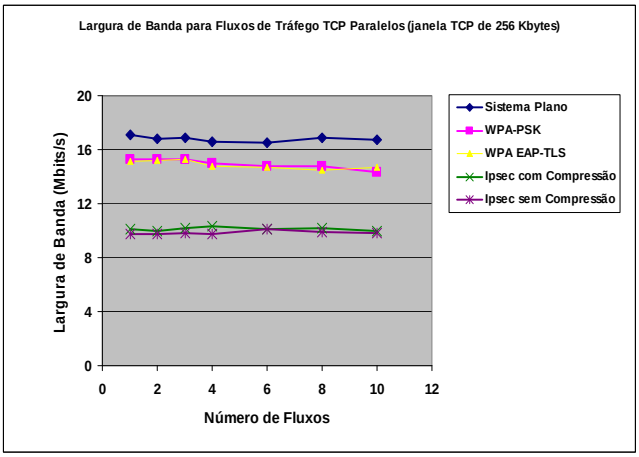
concluir que à medida que aumenta o número de clientes diminui o desempenho do sistema, sendo mais evidente esta degradação do desempenho nos sistemas WPA. O aumento do número de clientes implica o aumento do número de mensagens a circular na rede, e consequentemente o aumento do número de colisões. Outro factor que contribui para a diminuição da taxa de transmissão é o aumento do número de mensagens recebidas pelo AP. Como o AP não tem capacidade para processar todas as mensagens recebidas, rejeita um maior número de mensagens, o que leva o TCP a activar o mecanismo de controlo de congestionamento da rede. Comparativamente aos sistemas WPA, os sistemas IPsec apresentam um desempenho da rede mais constante. Isto deve-se ao facto de, para os sistemas IPsec o processo de cifra e de alteração dos pacotes IP ser mais moroso, o que implica um menor número de mensagens a circular na rede. Outro factor que contribui para este resultado é também a maior capacidade de processamento da *gateway* VPN e do cliente relativamente ao AP (mais limitado ao nível do processamento).



a) Janela TCP de 64 *Kbytes.*

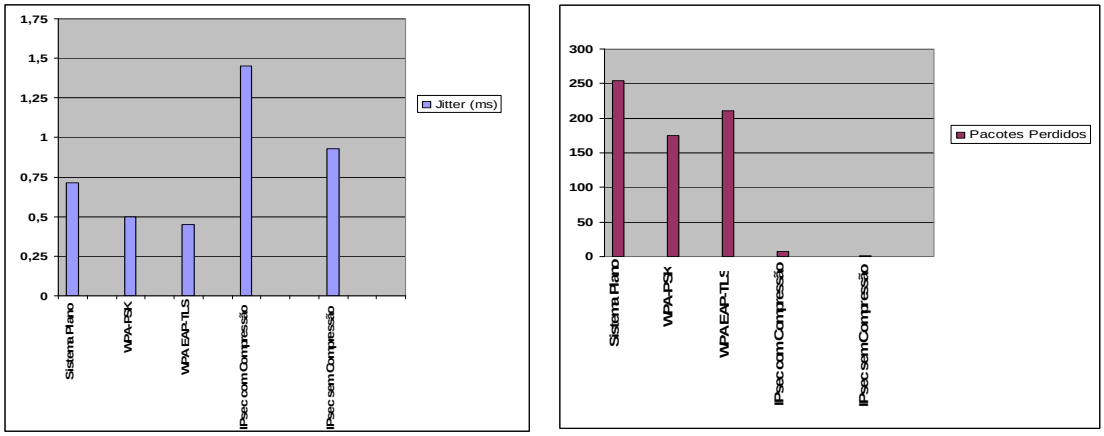


b) Janela TCP de 128 *Kbytes*



c) Janela TCP de 256 *Kbytes*

Figura 7.48 - *Throughput* para fluxo de tráfego TCP.



a) *Jitter* b) Pacotes perdidos.

Figura 7.49 - *Jitter* e número de pacotes perdidos – largura de banda=10*Mbits/s*.

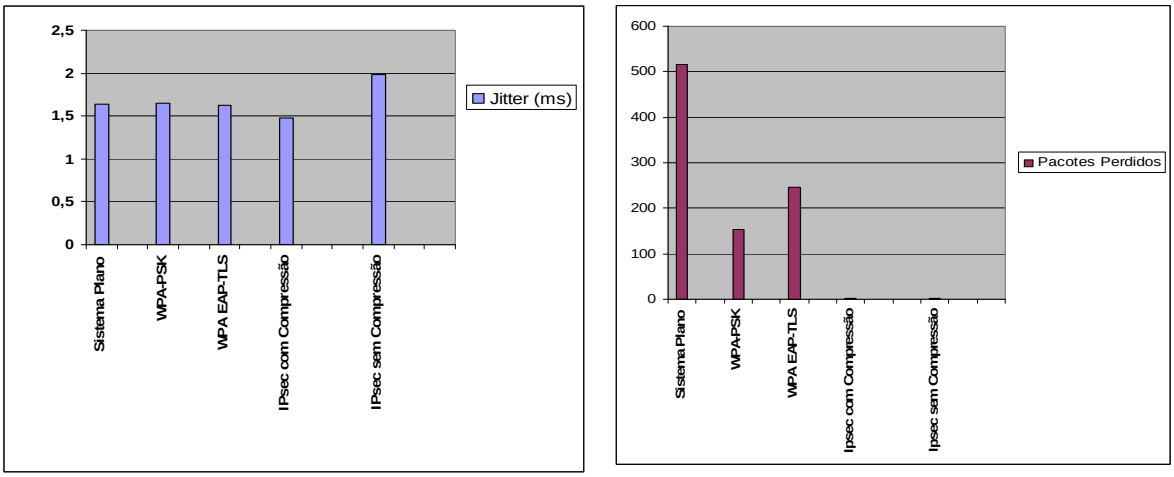
De seguida apresentam-se os parâmetros de desempenho variação do atraso (*jitter*) e o número de pacotes perdidos. Para estas experiências são utilizados fluxos de dados UDP. As experiências realizadas utilizam um fluxo de dados UDP com datagramas de 1500 *bytes*, *buffer* UDP de 64 *Kbytes* e largura de banda de 10 *Mbits/s*. A Figura 7.49 apresenta os resultados. Nesta experiência são gerados 10 fluxos de dados UDP com a duração de 10 segundos.

Os resultados obtidos indicam que os valores da variação do atraso para as implementações IPsec são bastante superiores ao das outras implementações. Um sistema IPsec para proteger os dados encapsula um pacote IP sobre um outro pacote IP, o que exige maior tempo de processamento introduzindo de forma notória valores de variação de atraso superiores. Para além disso o algoritmo de cifra utilizado pelo IPsec é bastante mais lento que o utilizado pelo WPA, o que também contribui para o valor obtido para o *jitter*. Outro factor que introduz maior variação de atraso é a compressão. O tempo e processamento dispendidos na compressão de uma mensagem também têm como consequência uma maior variação do atraso final. Como as implementações WPA não alteram de forma significativa o pacote, a variação do atraso é significativamente menor. A maior variação do atraso no sistema plano em relação ao WPA pode ser explicada pelo facto de o AP estar mais congestionado no sistema plano (maior envio de pacotes).

Em relação ao número de pacotes perdidos, as implementações IPsec apresentam os melhores resultados. Nas implementações WPA, o processo de protecção dos pacotes é efectuado entre o cliente e o AP, enquanto que nas implementações IPsec esse processo é efectuado entre o cliente e a *gateway* VPN. A capacidade de processamento e o tamanho do *buffer* do AP são significativamente inferiores aos da *gateway*. No processo de decifra dos pacotes, e devido à menor capacidade de processamento do AP, o *buffer* deste é rapidamente

preenchido. Quando o AP não tem disponibilidade no *buffer* para armazenar pacotes, estes são eliminados. O sistema plano apresenta o maior valor de pacotes perdidos. Como no sistema plano, os pacotes não são alterados, é possível ao cliente enviar uma quantidade de informação superior para a rede, o que provoca maior perda de pacotes. Este número elevado de mensagens enviadas tem também como consequência um acréscimo de colisões de pacotes na rede, o que contribui para o aumento do número de pacotes perdidos.

A Figura 7.50 apresenta os valores da mesma experiência, mas para uma largura de banda disponível de 54 *Mbits/s*. Este valor foi escolhido pelo facto de uma rede sem fios a funcionar em modo *g* disponibilizar uma largura de banda de 54 *Mbits/s*.



a) *Jitter*. b) Pacotes perdidos.

Figura 7.50 - *Jitter* e número de pacotes perdidos – largura de banda=54*Mbits/s*.

A observação da Figura 7.50 permite concluir novamente que o número de pacotes perdidos pelas implementações IPsec é muito inferior aos valores obtidos pelas implementações WPA. Os valores da variação do atraso para uma largura de banda disponível de 54 *Mbits/s* são muito semelhantes entre as implementações WPA e a implementação em sistema plano. As implementações IPsec, uma vez mais, praticamente não apresentam perdas.

Para se determinar o valor da variação do atraso, criado pela utilização de várias aplicações (variação do atraso instantâneo), utiliza-se um fluxo de dados UDP com uma largura de banda disponível de 54 *Mbits/s*, alterando-se o valor do tamanho das mensagens que constituem o fluxo de dados para valores compreendidos entre os 32 *bytes* e os 8 *Kbytes*. A Figura 7.51 apresenta os resultados obtidos com esta experiência, em que para as situações em que o tamanho das mensagens é pequeno, os resultados obtidos pelas implementações WPA aproximam-se dos valores da variação de atraso obtidos pelas implementações IPsec. Nestes casos o processo de compressão das mensagens não introduz ganhos significativos. Os resultados obtidos por esta experiência indicam que as implementações IPsec funcionam de

forma mais eficaz nas situações em que existe transferência de grandes quantidades de informação.

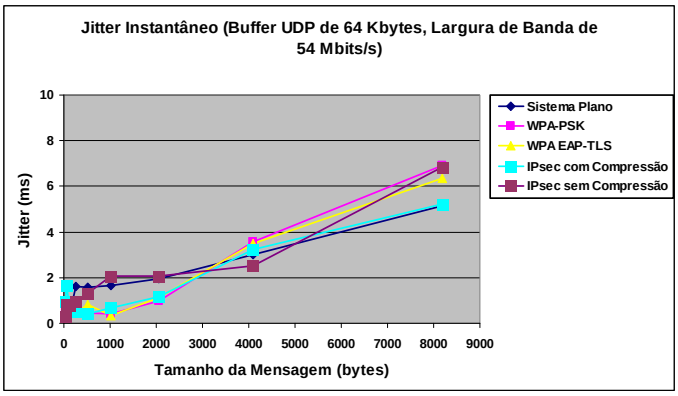


Figura 7.51 - *Jitter* instantâneo.

As experiências realizadas com o *IPERF* permitem-nos concluir que para situações em que a rapidez na transferência da informação e a disponibilidade da mesma são factores a ter em conta deve-se, como nas transferências de ficheiros ou nos acessos a uma base de dados, optar pelas implementações WPA. No entanto, a nível de perdas verifica-se que as implementações IPsec obtêm os melhores resultados. Outro factor a ter em conta é o atraso no início de uma sessão. Para se estabelecer uma sessão IPsec são necessários cerca de 1.5 segundos, enquanto que uma sessão WPA-EAP precisa de aproximadamente 5 segundos. O tempo necessário ao estabelecimento de uma sessão WPA-EAP pode ser um inconveniente, nomeadamente quando é utilizado um serviço de rede como o DHCP. Neste caso, pode ocorrer um *timeout* não sendo possível ao cliente obter um endereço IP válido, não conseguindo assim aceder aos recursos da rede.

7.3.2. Resultados *(C)RUDE – RUDE* e *CRUDE*

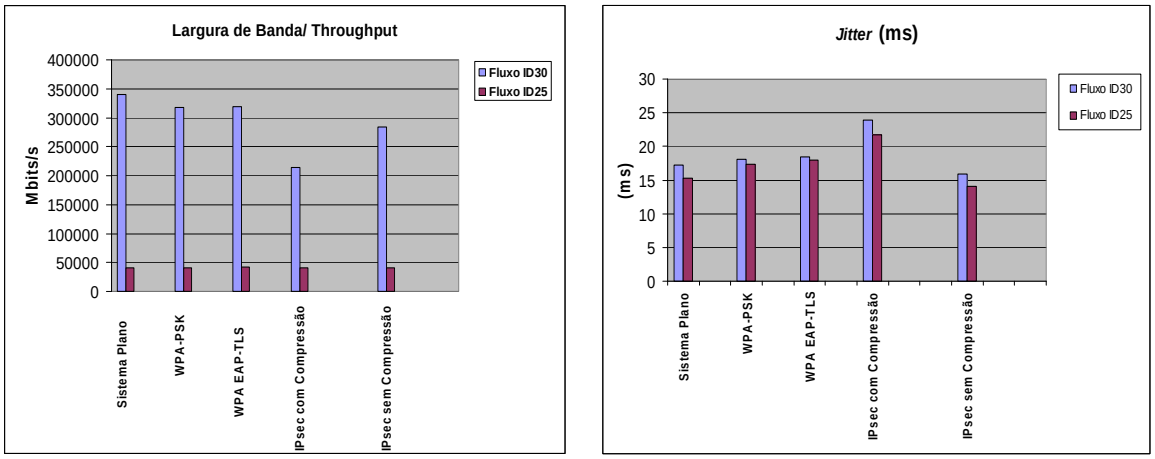
O RUDE é um emissor de tráfego UDP em tempo real e o *CRUDE* é um programa que recolhe as informações obtidas da rede; em conjunto permitem determinar o *throughput* e a variação do atraso (*jitter*) do tráfego numa rede, apenas baseados em tráfego UDP. Tal como o *IPERF*, é necessário efectuar a instalação em duas máquinas, uma a gerar tráfego (*RUDE*) e outra a efectuar a recolha da informação (*CRUDE*). Para a realização dos testes são criados dois tipos de fluxos de dados: *flow ID* 30 e *flow ID* 25. O *flow ID* 30 é um fluxo de 200 pacotes/s tendo cada pacote 250 *bytes*, o que perfaz um total de 50 *Kbytes*/s. Dois segundos após o início do fluxo, este é modificado para um fluxo de 200 *Kbytes*/s. Um segundo depois a largura de banda do fluxo aumenta para 1000 *Kbytes*/s. Este fluxo tenta simular uma aplicação multimédia. O segundo fluxo, o *flow ID* 25, é um fluxo 40 *Kbytes/s*; este fluxo tem também definido o tipo de serviço (*TOS*) de atraso reduzido (*LOW_DELAY*) para simular voz

sobre IP. Para ser possível obter os valores do *Jitter* e do *throughput* para as diversas implementações, instala-se o *CRUDE* numa máquina da rede com fios, e o *RUDE* num cliente da rede sem fios.

Os resultados obtidos pela utilização do *(C)RUDE* estão apresentados na Figura 7.52. A análise desta figura, e para uma quantidade de circulação de dados relativamente elevada (*flow id*=30), permite concluir que os valores de *throughput* são um pouco melhores nas situações WPA, reforçando assim os resultados obtidos pelo *IPERF*. Na situação em que a quantidade de dados em circulação não é muito elevado (*flow id*=25), o valor do *throughput* é semelhante em todas as implementações. Estes resultados estão de acordo com o esperado. A implementação IPsec com compressão é a que obtém o pior desempenho para a protecção de tráfego representativo da aplicação multimédia. Esta implementação introduz novos cabeçalhos nos pacotes IP, encapsula os pacotes IP sobre outros pacotes IP, cifra todos os dados e para além disso efectua a compressão dos pacotes. A realização de todos estes mecanismos tem como consequência o aumento significativo do tamanho de todos os pacotes, a necessidade de maior processamento por parte dos equipamentos envolvidos e, consequentemente, a necessidade de mais tempo para realizar todo o processo. A conjugação de todos estes factores faz com que esta implementação tenha o pior desempenho de entre todas as implementações. A implementação IPsec sem compressão consegue obter um desempenho muito semelhante às implementações WPA e ao sistema plano, e muito melhor que a implementação IPsec com compressão, pelo facto de reduzir a carga de processamento dos equipamentos e permitir reduzir de forma significativa o tempo dispendido na protecção dos pacotes. Comparativamente às implementações WPA, a implementação IPsec sem compressão obtém um resultado muito semelhante. Embora, o aumento de tamanho dos pacotes seja consideravelmente superior na implementação IPsec, a *gateway* tem, comparativamente com o AP, maior capacidade de processamento e um *buffer* (memória RAM) de maior capacidade.

Relativamente ao *jitter*, o melhor desempenho da implementação IPsec sem compressão está relacionada com a capacidade de processamento e o tamanho do *buffer* dos PCs. Os valores obtidos para o *jitter* na implementação IPsec com compressão permitem concluir que os equipamentos precisam de uma quantidade de tempo considerável para realizarem o processo de compressão, o que reforça os resultados obtidos pelo *IPERF* para a variação do atraso. O melhor desempenho da implementação IPsec sem compressão relativamente ao sistema plano e WPA está relacionada com o limite, em termos de capacidade de processamento e de *buffer* do AP. Pode-se concluir que a escolha do AP a

utilizar numa implementação WPA é um factor decisivo para o bom desempenho da mesma. Estes resultados reforçam de forma clara os resultados obtidos com a utilização do *IPERF*.



a) *Throughput* b) *Jitter*

Figura 7.52 - Resultados *(C)RUDE*.

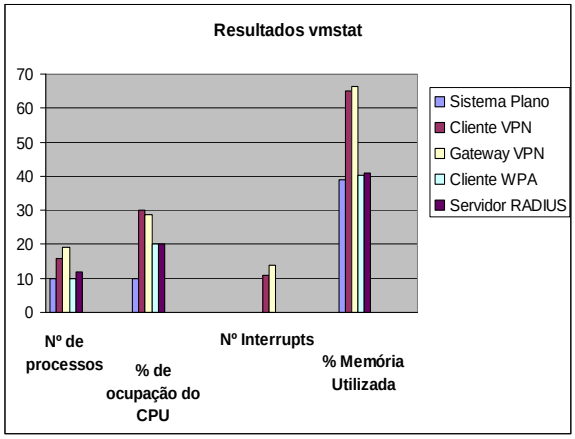
7.3.3. Resultados *sysstat* e *vmstat*

Para avaliar o desempenho dos equipamentos, nomeadamente dos servidores e dos clientes da rede sem fios, utilizam-se as ferramentas *sysstat* [44] e *vmstat* incluído no *Linux*. O *sysstat* é um conjunto de ferramentas que permitem medir o desempenho de uma máquina *Linux*, ao nível do processamento, IO, memória e de actividade do sistema. O *vmstat* é um comando *Linux* que reporta estatísticas sobre *kernel threads*, memória, discos, actividade do processador e *traps*. Convém mencionar que a sua utilização acarreta custos consideráveis no sistema. Para se obterem os resultados, é necessário incluir as ferramentas nos seguintes sistemas: cliente VPN, cliente WPA, servidor RADIUS e *gateway* VPN.

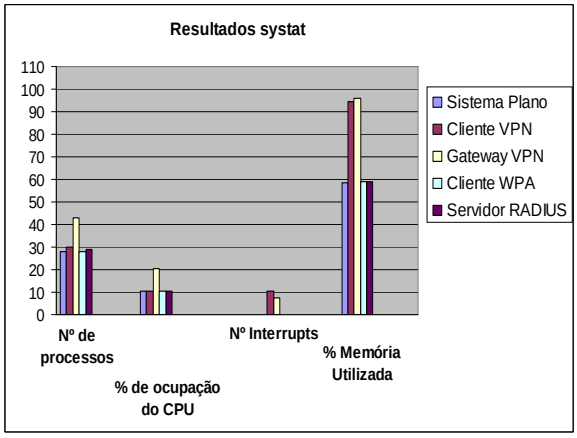
As ferramentas *sysstat* e *vmstat* na *gateway* e no cliente VPN permitem verificar que a existência do túnel implica um aumento de processamento do CPU, maior número de processos, uma maior percentagem de ocupação do CPU e um maior número de *interrupts* recebidos no sistema (Figura 7.53). Ao nível do sub-sistema de memória da *gateway* e do cliente VPN, a existência do túnel implica também um acréscimo significativo das suas necessidades.

Em relação à implementação WPA-EAP, activam-se as ferramentas *sysstat* e *vmstat* no servidor RADIUS e no cliente WPA, permitindo verificar que a utilização do WPA não implica um aumento das necessidades do sistema, mantendo-se quase sempre constantes todos os valores, nomeadamente a ocupação do CPU, o número de processos, os *interrupts* e a memória utilizada. As principais diferenças de valores indicados por cada uma das ferramentas devem-se aos custos específicos introduzidos por cada uma das ferramentas. Estes estão directamente relacionados com a forma de implementação, de interacção com o

sistema e de obtenção dos resultados das ferramentas. Podemos concluir que a ferramenta *sysstat* é aquela que, no seu funcionamento, utiliza mais memória e também utiliza mais processos para obter os resultados. A ferramenta *vmstat*, para recolher a informação, necessita de um maior número de *Interrupts* e de maior percentagem de ocupação do CPU.



a) Ferramenta *vmstat.*



b) Ferramenta *sysstat.*

Figura 7.53 - Desempenho dos equipamentos.

7.4. Conclusão

Neste capítulo efectuou-se uma análise ao funcionamento dos protocolos de segurança, cuja descrição da implementação se encontra no capítulo 6. A análise ao funcionamento das implementações permitiu verificar as diversas fases de negociação dos protocolos. Foi possível verificar que a negociação do protocolo IPsec utiliza duas fases: a *main mode* e a *quick mode*. Na fase *main mode* verificou-se o processo necessário para a criação do IKE SA, que é depois utilizado para proteger a negociação do IPsec SA da fase *quick mode*. Só depois de estabelecido o IPsec SA é que os dados são enviados de forma protegida entre o emissor e o receptor utilizando o protocolo ESP. A análise ao funcionamento da implementação WPA-EAP permitiu verificar a existência de duas fases no seu processo de negociação. Na primeira fase foi efectuada a autenticação mútua entre o cliente e o servidor de autenticação, e foi obtida uma chave de sessão temporária (PMK). Na segunda fase foram negociadas as chaves PTK e GTK necessárias à protecção da troca de informação entre o AP e o cliente. Relativamente à implementação WPA-PSK, a sua análise permitiu identificar apenas uma fase de negociação, a negociação das chaves PTK e GTK, já que a chave PMK foi previamente configurada no AP e no cliente.

Para ser possível concluir sobre o nível de protecção que cada implementação garante, efectuaram-se um conjunto de ataques de segurança aos mesmos. Os ataques realizados foram do tipo *Man-In-The-Middle*, negação de serviço, repetição e modificação da informação. Nas implementações WPA-EAP e IPsec, os ataques *Man-In-The-Middle,* repetição e modificação

da informação falharam. No entanto, foi possível realizar com sucesso ataques do tipo de negação de serviço. Estes resultados permitem concluir que estas implementações garantem os serviços de autenticação, confidencialidade, integridade e não-repúdio. Na implementação WPA-PSK, para além do ataque de negação de serviço, foi possível também realizar ataques passivos de dicionário e obter a PMK. Depois de obtida a PMK bastou configurá-la num cliente e aceder à rede de forma não-autorizada. Conclui-se assim que a implementação WPA-PSK não protege de forma eficaz uma rede, e não garante os serviços de autenticação, confidencialidade e não-repúdio.

Para finalizar este capítulo, realizaram-se um conjunto de experiências que permitiram analisar o desempenho da rede quando cada uma das implementações se encontra activa. Os resultados obtidos permitem concluir que, ao nível do desempenho da rede para comunicações TCP (transferências de ficheiros), aquelas que apresentam mais vantagens são as implementações WPA. Para comunicações sensíveis a perdas de pacotes, a que apresenta melhores resultados é a implementação IPsec. Foram também realizadas experiências ao nível de desempenho dos equipamentos, tendo a implementação IPsec obtido sempre os piores resultados, nomeadamente ao nível da percentagem de utilização do CPU, memória, *Interrupts* e número de processos.

Capítulo 8

Conclusão e trabalho futuro

Nesta dissertação apresentou-se um estudo sobre as principais características de alguns dos mecanismos de segurança de uma rede sem fios. Os mecanismos de protecção de redes sem fios estudados foram os protocolos WEP, IPsec, 802.1X, WPA e IEEE802.11i. Para ser possível avaliar as capacidades de segurança dos protocolos implementaram-se em cenário experimental o protocolo IPsec e o WPA. Escolheram-se estes dois protocolos por serem aqueles que mais garantias de segurança introduzem numa rede sem fios. Não foi possível efectuar um estudo prático do protocolo IEEE802.11i, pelo facto de não existirem ainda equipamentos compatíveis com a norma.

O WEP foi a primeira tentativa de criar um mecanismo de segurança em redes sem fios. Este protocolo utiliza, para garantir a confidencialidade da informação, o algoritmo de cifra RC4 com uma chave de 40 *bits*. A integridade da informação é conseguida através de um *checksum* calculado de forma linear. Para garantir a utilização de chaves de cifra diferentes por pacote, o WEP utiliza um IV de 24 *bits*. Rapidamente se mostrou que este protocolo era incapaz de proteger, de uma forma robusta e eficaz, as redes sem fios. O IV utilizado tem um tamanho demasiado pequeno, o que permite efectuar ataques de colisão ao IV, permitindo assim obter a chave de cifra. Dado o *checksum* ser um mecanismo obtido de forma linear, o WEP também não protege contra falsificações e ataques de repetição.

O 802.1X é um mecanismo que garante o controlo de acesso, não tendo sido especificamente desenvolvido para as redes sem fios. Inicialmente, o 802.1X foi aplicado às redes sem fios para ser utilizado conjuntamente com o WEP. A combinação destes dois protocolos permitia, de certa forma, contornar os problemas do WEP de falta de geração de chaves por sessão. Quando utilizado em associação com o WEP, o 802.1X gera uma chave diferente por cada sessão iniciada, que depois é utilizada pelo WEP para efectuar a cifra dos pacotes. Como o 802.1X não definia claramente que tipo de esquema se deveria utilizar para a cifra dos dados, é considerado um mecanismo de transição. Outro dos problemas apontados ao 802.1X é o facto de apenas garantir a autenticação mútua entre o AP e o servidor de autenticação. Este é um problema muito importante, já que o que se pretende é garantir uma comunicação segura entre o AP e os clientes da rede sem fios.

Devido a todos estes problemas, o IEEE criou um grupo de trabalho, IEEE 802.11i, com a finalidade de criar um protocolo que garantisse, de forma eficaz e robusta, a protecção

das redes sem fios. Como a normalização demora sempre algum tempo, e dada a necessidade imediata de se implementar um mecanismo capaz de proteger as redes sem fios, a *Wi-Fi Alliance* desenvolveu o protocolo WPA. Este protocolo implementa algumas das directrizes do 802.11i. Ambos os protocolos definem como mecanismo de controlo de acesso à rede o 802.1X; também definem um IV de 48 *bits*, o que elimina todos os problemas apontados ao IV do WEP. A integridade das mensagens é obtida pelo MIC. Este valor é calculado utilizando rotações, substituições e operações XOR; o MIC garante de forma eficiente a integridade das mensagens. A principal diferença entre estes dois protocolos é a protecção da transferência das mensagens (confidencialidade). O WPA define como mecanismo a utilizar na transferência das mensagens o TKIP. O TKIP utiliza como algoritmo de cifra o RC4 e, para garantir a utilização de chaves por pacote, define um mecanismo de hierarquia de chaves. Pelo seu lado, o IEEE802.11i define como protocolos de protecção da transferência de informação o TKIP e o AES. O algoritmo AES é um algoritmo de chave simétrica ao qual ainda não foi possível realizar com sucesso ataques.

Uma outra solução que permite o reforço da segurança numa rede sem fios é a utilização de um mecanismos de protecção ao nível da rede, o IPsec. O IPsec é um conjunto de protocolos de segurança que permitem criar túneis para a protecção da transferência dos dados. O IPsec define dois modos para efectuar a protecção da informação, o modo túnel e o modo transporte. No modo túnel, o mais seguro, para além de ser introduzido um novo cabeçalho IP, toda a informação é cifrada. O IPsec utiliza como algoritmo de cifra o 3DES. Este algoritmo é um algoritmo de chave simétrica que exige, no entanto, uma capacidade superior de processamento dos equipamentos comparativamente ao RC4.

A implementação dos cenários experimentais IPsec e WPA teve como principais objectivos a verificação das dificuldades ao nível da instalação, configuração e utilização, análise da sua capacidade em resistir contra ataques à confidencialidade, integridade e negação de serviço, análise do custo de desempenho numa rede, associado à introdução de cada um destes mecanismos para efectuar a protecção de uma rede sem fios, e da carga de processamento nos elementos da rede.

Ao nível da instalação, configuração e utilização pode-se concluir que, embora com algum esforço, é possível a implementação desses sistemas recorrendo essencialmente a ferramentas em código aberto. Outra conclusão a retirar é que para ser possível a implementação dos sistemas WPA, é por vezes necessário, e sobretudo em equipamentos mais antigos, realizar uma actualização de *firmware*, ou adquirir equipamentos novos e compatíveis com a norma 802.11g. O WPA é apenas compatível com equipamentos 802.11g.

Relativamente ao IPsec, este é fácil de implementar com equipamentos compatíveis com quaisquer das normas 802.11, nomeadamente a 802.11b e a 802.11g.

A realização dos ataques à segurança dos protocolos implementados permite concluir que, ao nível da protecção contra ataques MITM, de modificação de informação, falsificação e de controlo de sessão, os protocolos WPA e IPsec são eficazes. Ambos os mecanismos não protegem contra ataques de negação de serviço, sendo extremamente fácil realizar com sucesso ataques desse tipo aos dois protocolos. Relativamente à variante WPA-PSK, foi possível através da utilização de ferramentas de *software*, realizar ataques de dicionário à PMK. Depois de obtida a PMK, foi fácil aceder aos recursos da rede e realizar ataques à confidencialidade da informação.

A avaliação do custo introduzido no desempenho de uma rede em cada um dos cenários implementados foi efectuada medindo valores como o *throughput*, o número de *bytes* enviados, o *jitter* e o número de pacotes perdidos. As implementações WPA apresentam sempre melhores valores quando comparadas com as implementações IPsec relativamente ao *throughput*, ao número de bytes enviados e ao *jitter*. Estes resultados devem-se a que, no processo de protecção dos dados, as implementações IPsec introduzem alterações mais significativas nos pacotes IP, nomeadamente pela introdução de novos cabeçalhos, pelo encapsulamento IP dos pacotes, e pela utilização de um algoritmo de cifra que precisa de mais tempo para concluir o seu processo. A soma de todos estes factores contribui para o menor desempenho nas métricas antes referidas. As implementações IPsec apresentam valores muito menores que as implementações WPA relativamente ao número de pacotes perdidos. Um aspecto que permite compreender estes resultados é o facto de no IPsec o processo de protecção das mensagens ser efectuado entre um cliente da rede sem fios e uma *gateway* VPN. No caso do WPA esse processo é efectuado entre o AP e o cliente. Como no mesmo espaço de tempo, o cliente WPA envia mais mensagens para o AP, e como as suas capacidades de *buffer* e de processamento são inferiores ao do PC da *gateway* VPN, o número de pacotes perdidos pelo sistemas WPA é muito superior aos do IPsec. Pode-se então concluir que as implementações WPA são mais adequadas para as situações onde a disponibilidade imediata da informação é o factor principal, mas em que a perda de informação não é relevante.

Ao nível do custo no desempenho dos equipamentos, mediu-se a percentagem de ocupação de CPU, a percentagem de memória utilizada, o número de *interrupts*, e o número de processos. Os resultados obtidos revelam que existe sempre uma maior sobrecarga de CPU

e de memória utilizada nas implementações IPsec. Estes resultados são consequência das maiores alterações introduzidas pelo IPsec nas mensagens.

Este trabalho representa um começo do estudo de mecanismos de segurança em redes sem fios. No entanto, um aprofundamento deste estudo deve ser realizado. Exemplos de propostas de trabalho futuro incluem o estudo da segurança da norma 802.11i, estudo destes mecanismos em cenários de mobilidade dos clientes (entre redes diferentes), e estudo de melhorias a efectuar aos mecanismos estudados no sentido de melhorar o seu desempenho ao nível da protecção e segurança, assim como ao nível do impacto que introduzem na rede.

Referências

[1] Schneier, B., *Applied Cryptography – Protocols, Algorithms, and Source Code in C*. 2nd ed. 1996: John Wileye & Sons, Inc. 755.

[2] Schneier, B., *Secrets & Lies – Digital Security in a Networked World*. 2000: John Wiley & Sons. 432.

[3] Amoroso, E., *Fundamentals of Computer Security Technology*: Prentice-Hall International Editions. 403.

[4] Randall Nichols, P.L., *Wireless Security*. 2002: McGraw-Hill Telecom Professional – Models, Threats, and Solutions. 657.

[5] Christian Barnes, T.B., Donald Lloyd, Eric Ouellet, Jeffrey Posluns, David M. Zendzian, Neal O´Farrell, *Hack Proofing Your Wireless Network*. 2nd ed. 2002: Syngress. 483.

[6] Naganand Doraswamy, D.H., *IPSec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks*. 2nd ed. 2003: Prentice Hall. 165.

[7] Edd Dunbill, B.J., Roger Weeks, *Linux Unwired*. 2004: O´Reilly. 312.

[8] Andrew Vladimirov, K.V.G., Andrei A. Mikhailovsky, *Wi-Foo: The Secrets of Wireless Hacking*. 2004: Addison Wesley Professional. 592.

[9] Evi Nemeth, G.S., Scott Seebas, Trent R. Hein, *Unix System Administration Handbook*. 3rd ed. 2001: Prentice Hall. 853.

[10] N. Borisov, I. Goldberg and D. Wagner, *Intercepting Mobile Communications:The Insecurity of 802.11*. 2001.http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf, Julho de 2001.

[11] Página da implementação *OpenSSL*, http://www.openssl.org. Junho 2005

[12] Página da imlementação *FreeRADIUS*, http://www.freeradius.org. Junho 2005

[13] Página da ferramenta de análise de redes *Netstumbler*, http://www.netstumbler.com. Junho 2005

[14] Página da ferramenta *Airsnort*, http://airsnort.shmoo.com. Junho 2005

[15] Página da ferramenta de análise de pacotes *Airopeek*, http://www.wildpackets.com/products/airopeek. Junho 2005

[16] Página da ferramenta de auditoria de redes sem fios *Sniffer Wireless*, http://www.networkgeneral.com/. Junho 2005

[17] Página do analisador de protocolos *Ethereal*, http://www.ethereal.com. Junho 2005

[18] Página das ferramentas de auditoria e de teste de segurança *Dsniff*, http://naughty.monkey.org/~dugsong/dsniff. Junho 2005

[19] Página da ferramenta de auditoria *Kismet*, http://www.kismetwireless.net. Junho 2005

[20] Página da ferramenta de realização de ataques MITM *Ettercap*, http://ettercap.sourceforge.net/. Junho 2005

[21] Página da ferramenta de monitoração de redes *Etherape*, http://etherape.sourceforge.net. Junho 2005

[22] Página da ferramenta de monitoração de redes *Snips*, http://www.navya.com/software/snips. Junho 2005

[23] Página da implementação *FreeS/Wan*, http://www.freeswan.org. Junho 2005

[24] Página de configuração de uma VPN IPsec numa máquina *Linux*, http://www.tldp.org/HOWTO/VPN-Masquerade-HOWTO.html. Junho 2005

[25] Página de configuração de uma VPN numa máquina *Linux* http://tldp.org/HOWTO/VPN-HOWTO/index.html. Junho 2005

[26] Página com exemplos de configuração do *FreeS/Wan*, http://www.natecarlson.com/linux. Junho 2005

[27] Página com exemplos de configuração do *FreeS/Wan*, http://www.jacco2.dds.nl/networking/freeswan-l2tp.html. Junho 2005

[28] Página com informação dos sistemas compatíveis com o *FreeS/Wan*, http://www.freeswan.org/freeswan_trees/freeswan-2.03/doc/interop.html. Junho 2005

[29] Página da ferramenta de protecção de intrusos *Airmagnet*, http://www.airmagnet.com. Junho 2005

[30] Página da ferramenta *Cain e Abel*, http://www.oxid.it. Junho 2005

[31] Jesse Walker, *802.11 Security Séries Part II: The Temporal Key Integrity Protocol*. 2002. http://cedar.intel.com/media/pdf/security/80211_part2.pdf

[32] Página da *Meetinghouse*, http://www.mtghouse.com. Junho 2005

[33] Página da *Funk*, http://www.funk.com. Junho 2005

[34] Página da implementação *Cistron*, http://www.radius.cistron.nl. Junho 2005

[35] Página da imlementação *Open1x*, http://www.open1x.org. Junho 2005

[36] Página da implementação *HostAP*, http://hostap.epitest.fi. Junho 2005

[37] Página da implementação *AVS AP*, http://www.linux-wlan.org. Junho 2005

[38] Página da implementação *OpenCA*, http://www.openca.org. Junho 2005

[39] Página da implementação *wpa_supplicant*, http://hostap.epitest.fi/wpa_supplicant/. Junho 2005

[40] Ferramenta *FreeS/Wan* para *Windows*, http://vpn.ebootis.de/package.zip. Junho 2005

[41] Página de descrição de uma implementação EAP TLS, http://www.impossiblereflex.com/8021x/eap-tls-HOWTO.htm. Dezembro 2004

[42] Página da implementação *IPERF*, http://dast.nlanr.net/Projects/Iperf/. Junho 2005

[43] Página da implementação *(C)RUDE*, http://rude.sourceforge.net/. Junho 2005

[44] Página da implementação *Sysstat*, http://perso.wanadoo.fr/sebastien.godard. junho 2005

[45] Página da *DLink*, http://www.dlink.com. Junho 2005

[46] Página do WPA, http://www.wi-fi.org/OpenSection/protected_access_archive.asp. Junho 2005

[47] Página da associação *wireless lan*, http://www.wlana.org. Junho 2005

[48] Página de exemplos de configuração para redes sem fios, http://www.bawug.org/howto/. Junho 2005

[49] Página da implementação *linux wlan*, http://www.linux-wlan.org/. Junho 2005

[50] Página com boas práticas para protecção das redes sem fios, http://www.wardriving.com/. Junho 2005

[51] Página da associação *Westchester*, http://www.weca.org/. Junho 2005

[52] Página da implementação *coWPAtty*, http://sourceforge.net/projects/cowpatty. Junho 2005

[53] Página da implementação *Ptcrack*, http://sourceforge.net/projects/ptcrack. Junho 2005

[54] William A. Arbaugh, N.S., Y.C. Justin Wan, *Your 802.11 Wireless Network has No Clothes*. 2001. http://www.cs.umd.edu/~waa/wireless.pdf

[55] Página com boas práticas de segurança para *Linux*, http://www.linux-sec.net/. Junho 2005

[56] Página do fabricante de placas-mãe *Soekris*, http://www.soekris.com/. Junho 2005

[57] Fred Tanzella, *WIRELESS LAN SECURITY – HOW TO PROTECT WLAN. 2004* http://airdefense.net/wirelesslansecurity/wlan_security_whitepaper.html

[58] Página com informação sobre segurança de sistemas informáticos, http://www.securiteam.com/. Junho 2005

[59] Página com informação de segurança em redes sem fios, http://www.wardrive.net/. Junho 2005

[60] Página de implementação de uma AC com o *OpenSSL*, http://www.pseudonym.org/ssl/ssl_ca.html. Junho 2005

[61] Página do gestor de redes sem fios para KDE, http://kwifimanager.sourceforge.net/. Junho 2005

[62] Página da implementação *Snort*, http://www.snort.org/. Junho 2005

[63] Página do plugin *Snortsam* para o *Snort*, http://www.snortsam.net/. Junho 2005

[64] Página do driver para placas de rede *Atheros*, http://madwifiwiki.thewebhost.de/wiki/MadWifi. Junho 2005

[65] Página do CERT, http://www.cert.org/. Junho 2005

[66] Página de inúmeros recursos de segurança para redes, http://packetstormsecurity.nl/. Junho 2005

[67] Página da implementação *Wavesec*, http://www.wavesec.org/. Junho 2005

[68] Página de instalação e configuração *SuperFreeS/Wan*, http://www.strongsec.com/freeswan/install.htm. Junho 2005

[69] Página de configuração de um túnel Ipsec entre um sistema *FreeS/Wan* e um sistema *Windows*, http://www.natecarlson.com/linux/ipsec-x509.php. Junho 2005

[70] Adam Sulmicki, *HOWTO on EAP/TLS authentication between FreeRADIUS and XSupplicant.* http://www.missl.cs.umd.edu/wireless/eaptls/. Junho 2005

[71] Andreas Steffen, *Virtual Private Networks Coping with Complexity*. 2003. Security Group Zurcher Hochshule Wintherthur.

[72] Thomas Walpuski, *IPsec in Tunnel Mode between Windows XP Professional and OpenBSD with X.509v3 Certificate Authentication*.2002

[73] SSH Communications Security Corp. *VPN Connection to FreeS/WAN IPsec Gateway*. 2002. http://www.ssh.com. Junho 2005

[74] Jon Harrison, *VPN Technologies- A Comparison*. Data Connection, http://www.dataconnection.com, 2003

[75] *Proceedings of the 9th USENIX Security Symposium*. Denver, Colorado, 14-17 Agosto 2000

[76] Cisco Systems, *SAFE VPN IPsec Virtual Private Networks in Depth*. http://www.cisco.com/warp/public/ cc/so/cuso/epso/sqfr/safev_wp.pdf

[77] TimeStep, *Understanding the IPsec protocol suite*. Outubro 2004. http://www.firstvpn.com/papers/timestep/ipsecv2-Dec98.pdf

[78] Jean Tourrilhes, *Wireless Extensions for Linux*. 1997. http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Wireless.html

[79] Atheros Communications, *Building Scure Wireless Networks.* http://www.atheros.com/pt/atheros_security_whitepaper.pdf

[80] NetScreen Technologies, *Securing Wireless LANs*. http://www.netscreen.com/auth/register.jsp?_returnurl=http%3A%2F%2Fwww.juniper.

net%2Fsolutions%2Fliterature%2Fwhite_papers%2Fwp_wirelesslan.pdf&_id=www.whitepapers, Janeiro de 2003

[81] Cisco Systems, *Cisco SAFE: Wireless LAN Security in Depth*. 2003. http://www.cisco.com/warp/public/ cc/so/cuso/epso/sqfr/safwl_wp.pdf

[82] Cisco Systems, *Extensible Authentication Protocol Transport Layer Security Deployment Guide for Wireless LAN Networks*. 2002. http://www.cisco.com/warp/public/ cc/pd/sqsw/sq/tech/acstl_wp.pdf

[83] Jesse Walker, *802.11 Security Séries: Part I: The Wired Equivalent Privacy (WEP).* http://www.intel.com

[84] Jesse Walker, *802.11 Key management Séries: Part I: Key Management for WEP and TKIP*. http://www.intel.com

[85] Jesse Walker, *Overview of 802.11 Security*.http://www.intel.com

[86] Hassell, J., *RADIUS*. 1st ed. 2002: O'Reilly. 206.

[87] Página da *Wi-Fi Alliance*, http://www.wi-fi.org. Junho 2005

[88] Página da implementação *Surf*, http://www.ks.uiuc.edu/Research/vmd/doxygen/Surf_8C-source.html. Junho 2005

[89] Página da implementação *PingFlood*, http://packetstormsecurity.org/Exploit_Code_Archive/pingflood.c. Junho 2005

[90] Página da implementação *Fraggle*, http://packetstormsecurity.org/exploits/DoS/fraggle.c. Junho 2005

[91] Página da implementação *Neptune*, http://packetstormsecurity.org/Exploit_Code_Archive/neptune.c. Junho 2005

[92] Página da implementação *Synk4*, http://packetstormsecurity.org/Exploit_Code_Archive/synk4.c. Junho 2005

[93] Wash, R., *Lecture Notes on Stream Ciphers and RC4*. 2000.

[94] Xiaoyun Wang, D.F., Xuejia Lai, Hongbo Yu, *Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD*. 2004. p. 4.

[95] B. Aboba, D.S., *PPP EAP TLS Authentication Protocol, RFC2716*. 1999.

[96] Página do ITU, http://www.itu.int/publications/online/index.html. Junho 2005

[97] Funk, P., *EAP Tunneled TLS Authentication Protocol (EAP-TTLS), draft-ietf-pppext-eap-ttls-05.txt*. 2004, Funk Software, Inc.

[98] Carl Rigney, A.C.R., William Allen Simpson, Steve Willens, *Remote Authentication Dial In User Service (RADIUS), RFC2138*. 1997.

[99] J. Hodges, R.M., *Lightweight Directory Access Protocol (v3): Technical Specification, RFC3377*. 2002.

[100] Página do *Active Directory*, http://www.microsoft.com/windows2000/technologies/directory/ad/default.asp. Junho 2005

[101] B. Lloyd, W.S., *PPP Authentication Protocols, RFC1334*. 1992.

[102] Simpson, W., *PPP Challenge Handshake Authentication Protocol (CHAP), RFC1994*. 1996.

[103] G. Zorn, S.C., *Microsoft PPP CHAP Extensions, RFC2433*. 1998, Microsoft Corporation.

[104] Zorn, G., *Microsoft PPP CHAP Extensions, Version 2, RFC2759*. 2000, Microsoft Corporation.

[105] Ashwin Palekar, J.S., Glen Zorn, S. Josefsson, *Protected EAP Protocol (PEAP) Version 2, draft-josefsson-pppext-eap-tls-eap-10.txt*. 2004.

[106] Rivest, R., *The MD5 Message-Digest Algorithm, RFC1321*. 1992.

[107] Rivest, R., *The MD4 Message-Digest Algorithm, RFC1320*. 1992.

[108] C. Rigney, W.W., P. Calhoun, *RADIUS Extensions, RFC2869*. 2000.

[109] N. Haller, C.M., P. Nesser, M. Straw, *A One-Time Password System, RFC2289*. 1998.

[110] Simpson, W., *The Point-to-Point Protocol (PPP), RFC1661*. 1994.

[111] Página do motor de pesquisa *RpmSeek*, http://www.rpmseek.com. Junho 2005

[112] Página da implementação *ipsecmd*, http://www.microsoft.com/resources/documentation/windows/ xp/all/proddocs/en-us/ipsecmd.mspx. Junho 2005

[113] Página da implementação *Bridge-Linux Ethernet Bridging*, http://bridge.sourceforge.net. Junho 2005

[114] Schneier, B., *Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish)*. 1994. http://www.schneier.com/paper-blowfish-fse.html

[115] Ross Anderson, E.B., Lars Knudsen, *Serpent: A Proposal for the Advanced Encryption Standard*. 1998. http://www.tropsoft.com/strongenc/serpent.pdf

[116] IEEE Std. 802.11b (1999), Supplement to ANSI/IEEE Std. 802.11, 1999 Edition, *Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher Speed Physical Layer (PHY) Extension in the 2.4 GHz band*. 1999. http://standards.ieee.org/getieee802/

[117] IEEE Std. 802.11i (2004), *Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)Specifications Amendment 6: Medium Access Control (MAC) security Enhancements.*2004. http://standards.ieee.org/getieee802/

[118] R. Thayer, N.D., R. Glenn, *IP Security Document Roadmap, RFC2411.* 1998.

[119] S. Kent, R.A., *IP Authentication Header, RFC2402*. 1998.

[120] S. Kent, R.A., *IP Encapsulating Security Payload (ESP), RFC2406.* 1998.

[121] D. Harkins, D.C., *The Internet Key Exchange (IKE), RFC2409*. 1998.

[122] D. Maughan, M.S., M. Schneider, J. Turner, *Internet Security Association and Key Management Protocol (ISAKMP), RFC2408.* 1998.

[123] IEEE Std. 802.1x (2001), *Port-Based Network Access Control*, 2001. http://standards.ieee.org/getieee802/

[124] Página das normas IEEE802.11, http://grouper.ieee.org/groups/802/11/. Junho 2005

[125] Página dos equipamentos certificados *Wi-Fi*, http://www.wi-fi.org/OpenSection/certified_products.asp. Junho 2005

[126] Pat R. Calhoun, G.Z., *Roadmaps Authentication/Authorization Requirements, draft-ietf-aaa-roamops-auth-req-00.txt*. 1999.

[127] IEEE Std. 802.2 (1998), *Part 2: Logical Link Control*, 1998. http://standards.ieee.org/getieee802/

[128] Página das normas IEEE 802.10, http://grouper.ieee.org/groups/802/10/

[129] Página da implementação *Offset Code Book*, http://www.cs.ucdavis.edu/~rogaway/ocb/ocb-back.htm. Junho 2005

[130] Página da implementação WRAP, http://www.tech-faq.com/wireless-networks/wrap-wireless-robust-authenticated-protocol.shtml. Junho 2005

[131] B. Aboba, L.B., J. Vollbrecht, J. Carlson, H. Levkowetz, *Extensible Authentication Protocol (EAP), RFC3748*. 2004.

[132] J. Kohl, C.N., *The Kerberos Network Authentication Service (V5), RFC1510*. 1993.

[133] *ISO/IEC 7816-1, Identification cards – Integrated circuit(s) cards with contacts - Part 1: Physical characteristics*. 1998.

[134] W. Diffie and M.E. Hellman, *New directions in cryptography*, *IEEE Transactions on Information Theory* 22**.** 1976.

[135] W. Diffie, P.C. van Oorschot, and M.J. Wiener, *Authentication and authenticated key exchanges*, *Designs, Codes and Cryptography.* 1992.

[136] Standards, N.B.o., *Data Encryption Standard, FIPS-Pub.46*, U.S.D.o.C. National Bureau of Standards, Washington D.C., Editor. January 1977

[137] Página da RSA, http://www.rsasecurity.com. Junho 2005

[138] (NIST), N.I.o.S.a.T., *FIPS-197: Advanced Encryption Standard*. 2001, National Institute of Standards and Technology (NIST)

[139] Jakimoski et al, *Related-Key Differential Cryptanalysis of 192-bit Key AES Variants*, in *SAC 2003*. 2003, LNCS. p. 208-221.

[140] D. Whiting, R.H., N. Ferguson, *Counter with CBC-MAC (CCM), RFC3610*. 2003, IETF

[141] R. Rivest, A.Shamir.a.L.Adleman., *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Communications of the ACM, 1978.

[142] D. Eastlake, P.J., *US Secure Hash Algorithm 1 (SHA1), RFC3174*. 2001, IETF

[143] NIST, *Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques*. 2001.http://csrc.nist.gov/publications/.

[144] Página da Certicom, http://www.certicom.org. Junho 2005

Anexo A

Ficheiros de configuração IPsec

A.1 Ficheiro de configuração OpenSSL

A.1.1 openssl.cnf

```
HOME = .
RANDFILE = $ENV::HOME/.rnd
oid_section = new_oids
 [ new_oids ]
#####################################################################
[ ca ]
default_ca = CA_default # The default ca section
#####################################################################
[ CA_default ]
dir = ./demoCA # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir/newcerts # default place for new certs.
certificate = $dir/cacert.pem # The CA certificate
serial = $dir/serial # The current serial number
crl = $dir/crl.pem # The current CRL
private_key = $dir/private/cakey.pem# The private key
RANDFILE = $dir/private/.rand # private random number file
x509_extensions = usr_cert # The extentions to add to the cert
name_opt = ca_default # Subject Name options
cert_opt = ca_default # Certificate field options
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = md5 # which md to use.
preserve = no # keep passed DN ordering
policy = policy_match
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
```

```
commonName = supplied
emailAddress = optional
 [ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
#################################################################
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert
string_mask = nombstr
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = PT
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Portugal
localityName = Locality Name (eg, city)
localityName_default = Porto
0.organizationName = Organization Name (eg, company)
0.organizationName_default = FCUP
organizationalUnitName = Organizational Unit Name (eg, section)
organizationalUnitName_default = DCC
commonName = Common Name (eg, YOUR name)
commonName_max = 64
commonName_default =  IPsec CA
emailAddress = Email Address
emailAddress_max = 64
emailAddress_default = lbarreto@esce.ipvc.pt
[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20
```

```
unstructuredName = An optional company name
[ usr_cert ]
basicConstraints=CA:FALSE
nsComment = "OpenSSL Generated Certificate"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
[ v3_ca ]
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
basicConstraints = CA:true
[ crl_ext ]
authorityKeyIdentifier=keyid:always,issuer:always
```

A.2 Ficheiros de configuração do FreeS/Wan

A.2.1 *ipsec.conf* do *roadwarrior*

```
config setup
        interfaces=%defaultroute
        klipsdebug=none
        plutodebug=none
        plutostart=%search
        plutoload=%search
        uniqueids=yes
conn %default )
        keyintries=3
        disablearrivalcheck=no
        compress=yes
        authby=rsasig
        pfs=yes
        rightrsasigkey=%cert
        leftrsasigkey=%cert
conn wlan
        right=192.168.120.3
        rightsubnet=192.168.210.0/24
        rightcert=<gwCert.pem>
        left=%defaultroute
        leftcert=betaCert.pem
        auto=start
```

A.2.2 *ipsec.conf* da *gateway* VPN

```
config setup
        interfaces=”IPsec0=eth2”
        forwardcontrol=yes
        klipsdebug=none
        plutodebug=none
        plutoload=%search
        plutostart=%search
        uniqueids=yes
conn %default
        keyintries=3
        disablearrivalcheck=no
        authby=rsasig
        pfs=yes
        compress=yes
        rightrsasigkey=%cert
        leftrsasigkey=%cert
conn wlan
        right=192.168.123.3
        rightsubnet=192.168.210.0/24
        rightcert=gwCert.pem
        left=%any
        auto=add
```

A.2.3 *ipsec.secrets* do *roadwarrior*

```
:RSA        /etc/IPsec.d/private/betaKey.pem  “palavra-chave da chave privada
```

A.2.4 *ipsec.secrets* da *gateway* VPN

```
:RSA        /etc/IPsec.d/private/gwKey.pem  “palavra-chave da chave privada”
```

A.3 Script de configuração da Firewall

```
# NAT and FORWARD
iptables –t nat –A POSTROUTING –o eth1 –j MASQUERADE
iptables –A FORWARD –m state –state ESTABLISHED, RELATED –j ACCEPT
iptables –A FORWARD –i ipsec0 –j ACCEPT
# Permitir negociações IKE, autenticação e cifra ESP
iptables –A INPUT –p udp - -sport 500 - -dport 500 –j ACCEPT
iptables –A INPUT –p 50 –j ACCEPT
iptables –A INPUT –p esp –j ACCEPT
iptables –A OUTPUT –p udp --sport 500 –dport 500 –j ACCEPT
```

```
iptables –A OUTPUT –p 50 –j ACCEPT
iptables –A OUTPUT –p esp –j ACCEPT
iptables –A OUTPUT –p 51 –j ACCEPT
# Permitir apenas tráfego de e para a interface IPsec
iptables –A FORWARD –i eth1 –o eth2 –j DROP
iptables –A FORWARD –i eth2 –o eth1 –j DROP
iptables –A FORWARD –i ipsec0 –o eth1 –j ACCEPT
iptables –A FORWARD –i eth1 –o ipsec0 –j ACCEPT
iptables –A INPUT –i eth2 –j DROP
iptables –A OUTPUT –o eth2 –j DROP
```

Anexo B

Ficheiros de configuração WPA-EAP

B.1 Ficheiros de configuração OpenSSL

B.1.1 *Openssl.cnf*

```
HOME = .
RANDFILE = $ENV::HOME/.rnd
oid_section = new_oids
[ new_oids ]
#####################################################################
[ ca ]
default_ca = CA_default # The default ca section
#####################################################################
[ CA_default ]
dir = ./demoCA # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/index.txt # database index file.
new_certs_dir = $dir/newcerts # default place for new certs.
certificate = $dir/cacert.pem # The CA certificate
serial = $dir/serial # The current serial number
crl = $dir/crl.pem # The current CRL
private_key = $dir/private/cakey.pem# The private key
RANDFILE = $dir/private/.rand # private random number file
x509_extensions = usr_cert # The extentions to add to the cert
name_opt = ca_default # Subject Name options
cert_opt = ca_default # Certificate field options
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = md5 # which md to use.
preserve = no # keep passed DN ordering
policy = policy_match
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
```

```
commonName = supplied
emailAddress = optional
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
####################################################################
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert
string_mask = nombstr
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = PT
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Portugal
localityName = Locality Name (eg, city)
localityName_default = Porto
0.organizationName = Organization Name (eg, company)
0.organizationName_default = FCUP
organizationalUnitName = Organizational Unit Name (eg, section)
organizationalUnitName_default = DCC
commonName = Common Name (eg, YOUR name)
commonName_max = 64
commonName_default =  Wireless CA
emailAddress = Email Address
emailAddress_max = 64
emailAddress_default = lbarreto@esce.ipvc.pt
[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20
```

```
unstructuredName = An optional company name
[ usr_cert ]
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
[ v3_ca ]
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
basicConstraints = CA:true
[ crl_ext ]
authorityKeyIdentifier=keyid:always,issuer:always
```

B.1.2 *CA.root – Script* de geração da Autoridade de Certificação

```
#!/bin/sh
export PATH=/usr/local/openssl/misc:${PATH}
export LD_LIBRARY_PATH=/usr/local/openssl/lib
rm -rf demoCA
echo "**********************************************************************"
echo "Gera a chave privada e o certificado auto-assinados"
echo "Quando  solicitado altere o valor por defeito do campo Common Name"
echo "**********************************************************************"
echo
openssl req -new -x509 -keyout newreq.pem -out newreq.pem -passin pass:wpawpa -passout pass:wpawpa
echo "**********************************************************************"
echo "Gera a hierarquia da nova CA"
echo "**********************************************************************"
echo
echo "newreq.pem" | CA.pl -newca>/dev/null
echo "**********************************************************************"
echo "Gera a ROOT CA"
echo "**********************************************************************"
echo
openssl pkcs12 -export -in demoCA/cacert.pem -inkey newreq.pem -out root.p12 -cacerts -passin pass:wpawpa -
passout pass:wpawpa
openssl pkcs12 -in root.p12 -out root.pem -passin pass:wpawpa -passout pass:wpawpa
#Converte o formato do certificado PEM no formato DER
openssl x509 -inform PEM -outform DER -in root.pem -out root.der
#Remove o que não é necessário
rm -rf newreq.pem
```

B.1.3 *CA.svr – Script* de geração do certificado do servidor

```
#!/bin/sh
export PATH=/usr/local/openssl/misc:${PATH}
export LD_LIBRARY_PATH=/usr/local/openssl/lib
echo "************************************************************"
echo "Gera a chave privada do servidor"
echo "Quando  solicitado escreva o nome do servidor no campo Common Name"
echo "************************************************************"
echo
#Solicita um novo certificadoPKCS#10
openssl req -new -keyout newreq.pem -out newreq.pem -passin pass:wpawpa -passout pass:wpawpa
openssl ca -policy policy_anything -out newcert.pem -passin pass:wpawpa -key wpawpa –infiles newreq.pem
#Gera um ficheiro PKCS#12 a partir do novo certificado e da sua chave privada disponíveis em
#newreq.pem
openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12 -clcerts -passin pass:wpawpa -passout
pass:wpawpa
openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:wpawpa -passout pass:wpawpa
# Converte o formato do certificado PEM no formato DER
openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der
# Remove o que não é necessário
rm -rf newcert.pem newreq.pem
```

B.1.4 *CA.clt – Script* de geração do certificado dos clientes

```
#!/bin/sh
export PATH=/usr/local/openssl/misc:${PATH}
export LD_LIBRARY_PATH=/usr/local/openssl/lib
echo "************************************************************"
echo "Gera o certificado e a chave privada do cliente"
echo " Quando  solicitado escreva o nome do cliente no campo Common Name"
echo "Deverá ser igual ao USERNAME no FreeRADIUS"
echo "************************************************************"
echo
#Solicita um novo certificadoPKCS#10
openssl req -new -keyout newreq.pem -out newreq.pem -passin pass:wpawpa -passout pass:wpawpa
openssl ca -policy policy_anything -out newcert.pem -passin pass:wpawpa -key whatever –infiles newreq.pem
#Create a PKCS#12 file from the new certificate and its private key found in
#newreq.pem and place in file specified on the command line
openssl pkcs12 -export -in newcert.pem -inkey newreq.pem -out $1.p12 -clcerts -passin pass:wpawpa -passout
pass:wpawpa
#Gera um ficheiro PKCS#12 a partir do novo certificado e da sua chave privada disponíveis em
#newreq.pem
```

```
openssl pkcs12 -in $1.p12 -out $1.pem -passin pass:wpawpa -passout pass:wpawpa
# Converte o formato do certificado PEM no formato DER
openssl x509 -inform PEM -outform DER -in $1.pem -out $1.der
# Remove o que não é necessário
rm -rf newcert.pem newreq.pem
```

B.2 Ficheiros de configuração do FreeRadius

B.2.1 *radiusd.conf*

Neste ficheiro é apenas necessário alterar duas secções a *Authorization* e a *Authentication* de forma a possibilitar o *eap*.

```
authorize {
preprocess
eap
suffix
files
}
authenticate {
eap
}
```

B.2.2 *eap.conf*

Neste ficheiro definem-se os parâmetros da autenticação eap**.**

```
eap {
        default_eap_type = tls
        timer_expire     = 60
        ignore_unknown_eap_types = no
        cisco_accounting_username_bug = no
        md5 {
        }
        leap {
        }
        gtc {
                auth_type = PAP
        }
        tls {
                private_key_password = <palavra-chave utilizada para criar o certificado do servidor>
                private_key_file = /etc/1x/cert-srv.pem
                certificate_file = /etc/1x/cert-srv.pem
                CA_file = /etc/1x/demoCA/root.pem
```

```
        dh_file = /etc/1x/dh
        random_file = /etc/1x/random
        fragment_size = 1024
        include_length = yes
    }
```

B.2.3 *clients.conf*

```
client 127.0.0.1 {
secret = teste
shortname = localhost
nastype = other # localhost isn't usually a NAS...
}
client 192.168.120.1 {
secret = wpawpa
shortname = DLink-AP
}
client 127.0.0.1 {
secret = teste
shortname = localhost
}
```

B.2.4 *users*

```
"beta” Auth-Type := EAP
"teste" Auth-Type := Local, User-Password == "teste"
```

B.3 Ficheiro de configuração do wpa_supplicant (wpa_supplicant.conf)

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
eapol_version=1
ap_scan=1
network={
        ssid="wpa"
        #proto=RSN
        key_mgmt=WPA-EAP
        pairwise=TKIP
        group=TKIP
        eap=TLS
        identity="beta"
        ca_cert="/usr/local/certs/root.pem"
        client_cert="/usr/local/certs/ cert-clt.der"
        private_key="/usr/local/certs/ cert-clt.pem"
```

```
        private_key_passwd="palavra-chave utilizada na criação do certificado do cliente"
        priority=1
}
```

Anexo C

Ficheiros de configuração WPA-PSK

C.1 Ficheiro de configuração do wpa_supplicant (wpa_supplicant.conf)

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
eapol_version=1
ap_scan=1
network={
        ssid="wpa-psk”
        proto=WPA
        key_mgmt=WPA-PSK
        pairwise=CCMP TKIP
        group=CCMP TKIP WEP104 WEP40
        psk=<resultado obtido pelo wpa_passphrase>
}
```