Universidade Federal de Uberlândia
Faculdade de Computação
Programa de Pós-Graduação em Ciência da Computação

ANÁLISE DE SEGURANÇA DE ESQUEMAS DE PRIVACIDADE DE DADOS

Heveraldo Rodrigues de Oliveira

Uberlândia - MG
Setembro de 2007

ANÁLISE DE SEGURANÇA DE ESQUEMAS DE PRIVACIDADE DE DADOS

Por

Heveraldo Rodrigues de Oliveira

DISSERTAÇÃO APRESENTADA À UNIVERSIDADE FEDERAL DE UBERLÂNDIA, MINAS GERAIS, COMO PARTE DOS REQUISITOS EXIGIDOS PARA OBTENÇÃO DO TÍTULO DE MESTRE EM CIÊNCIA DA COMPUTAÇÃO.

Orientador: João Nunes de Souza - UFU
Co-Orientrador: Luís Fernando Faina - UFU

SETEMBRO DE 2007

©Todos os direitos reservados a Heveraldo Rodrigues de Oliveira

Dados Internacionais de Catalogação na Publicação (CIP)

O48a Oliveira, Heveraldo Rodrigues de, 1972-
Análise de segurança de esquemas de privacidade de dados / Heveraldo Rodrigues de Oliveira. - 2007.
153 f. : il.

Orientador: João Nunes de Souza.
Co-orientador: Luís Fernando Faina.
Dissertação (mestrado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Ciência da Computação.
Inclui bibliografia.

1. Criptografia de dados (Computação) - Teses. I. Souza, João Nunes de. II. Faina, Luís Fernando. III. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU: 681.3-78

Elaborado pelo Sistema de Bibliotecas da UFU / Setor de Catalogação e Classificação (CIP)

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE COMPUTAÇÃO

Os abaixo assinados, por meio deste, certificam que leram e recomendam para a Faculdade de Computação a aceitação da dissertação, intitulada "**Análise de Segurança de Esquemas de Privacidade de Dados**", por **Heveraldo Rodrigues de Oliveira**, como parte dos requisitos exigidos para a obtenção do título de **Mestre em Ciência da Computação**.

Uberlândia, 14 de setembro de 2007

Orientador: _____
Prof. Dr. João Nunes de Souza
Universidade Federal de Uberlândia UFU / MG

Co-Orientador: _____
Prof. Dr. Luís Fernando Faina
Universidade Federal de Uberlândia UFU / MG

Banca Examinadora: _____
Prof. Dr. Marco Aurélio Amaral Henriques
Universidade Estadual de Campinas UNICAMP / SP

_____
Prof. Dr. Ilmério Reis da Silva
Universidade Federal de Uberlândia UFU / MG

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Data: Setembro, 2007

Autor:	**Heveraldo Rodrigues de Oliveira**
Título:	**Análise de Segurança de Esquemas de Privacidade de Dados**
Faculdade:	**Faculdade de Computação**
Grau:	**Mestrado**

Fica garantido à Universidade Federal de Uberlândia o direito de circulação e impressão de cópias deste documento para propósitos exclusivamente acadêmicos, desde que o autor seja devidamente informado.

Autor

O AUTOR RESERVA PARA SI QUALQUER OUTRO DIREITO DE PUBLICAÇÃO DESTE DOCUMENTO, NÃO PODENDO O MESMO SER IMPRESSO OU REPRODUZIDO, SEJA NA TOTALIDADE OU EM PARTES, SEM A PERMISSÃO ESCRITA DO AUTOR.

Dedicatória

À minha esposa, Claudete, a minhas filhas, Amanda e Alanna, à minha mãe e irmãs e toda minha família. Aos meus colegas e alunos.

Agradecimentos

Agradeço primeiro a Deus por este momento.

Ao meu orientador e ao meu co-orientador, Profs. Drs. João N. Souza e Luís Fernando Faina, que me guiaram durante este trabalho.

Em especial, ao Prof. Dr. Ilmério Reis da Silva, por ter acreditado em minha capacidade para chegar aqui.

Aos colegas em Montes Claros, Renato, Christine, Renê e Expedito pelas sugestões.

À minha família, minha esposa, minhas filhas, minha mãe e minhas irmãs, que compreenderam a importância desse curso.

A todos os professores e amigos da Pós Graduação da Universidade Federal de Uberlândia, sou grato pelos ensinamentos, companheirismo e pela amizade.

Finalmente, agradeço a todos que contribuiram de alguma forma para a conclusão deste trabalho.

Resumo

Esta dissertação faz uma releitura do Modelo de Função Vantagem, de Goldwasser e Bellare, elucidando seus conceitos complexos e exemplificando sua aplicação na análise de segurança de esquemas reais de criptografia. São apresentadas várias formas de aplicação do modelo. A segurança dos esquemas utilizados no Padrão IEEE 802.11 são explanados e esses são usados como objeto de estudo para a aplicação do modelo. Os resultados podem ser usados para justificar a adoção de certas tecnologias de segurança em detrimento de outras. As análises apresentadas aqui demonstram a importância do estudo e uso de modelos matemáticos para provas de segurança e podem ajudar aos projetistas e desenvolvedores a maximizar a segurança de seus esquemas de criptografia, pela aplicação efetiva do modelo ou pelo simples entendimento dos conceitos de segurança abordados aqui.

Palavras-chave: Criptografia, esquemas criptográficos, privacidade de dados, redes locais sem fios.

Abstract

This dissertation walks through the Advantage Function Model, from Goldwasser and Bellare, explaining its complex concepts and exemplifying its application in the analysis of encryption security schemes currently used in practice. It is also shown ways to put this model into practice. The safety of the schemes used in the IEEE 802.11 Standard are explained and used as the object of study in the model application. The results can be used to justify the adoption of certain security technologies. The analysis presented demonstrates the great value of its study and of the use of mathematical models in safety demonstrations. It can help designers and developers to maximize the safety of their encryption schemes, by the effective application of the model or by the simple understanding of the concepts of security mentioned here.

Keywords: Cryptography, cryptography schemes, data privacy, wireless local networks

Sumário

Lista de Figuras **xii**

Lista de Acrônimos **xiv**

Lista de Símbolos **xvi**

1 Introdução **1**

1.1 Objetivos e Organização da Dissertação 2

2 Modelo de Segurança **3**

2.1 A Função Vantagem das PRFs e PRPs 3

2.1.1 Famílias de Funções 3

2.1.2 Funções Randômicas e Permutações Randômicas 4

2.1.3 Funções Pseudo-randômicas 5

2.1.4 Permutações Pseudo-randômicas 9

2.1.5 Seqüências de Famílias de PRFs e PRPs 12

2.1.6 Usando PRFs e PRPs 12

2.1.7 Exemplos de Ataques 14

2.1.8 Segurança Quanto à Recuperação da Chave 18

2.1.9 Ataque do Aniversário 24

2.1.10 PRFs *versus* PRPs 25

2.1.11 Construção de Famílias de PRFs 25

2.2 A Função Vantagem dos Esquemas de Criptografia de Chave Secreta 26

2.2.1 Esquema de Criptografia Simétrico 26

2.2.2 Alguns Esquemas de Criptografia 27

2.2.3 Pensando em Segurança 35
2.2.4 Segurança na Teoria da Informação 36
2.2.5 Indiscernibilidade Sob *Chosen-Plaintext Attack* (IND-CPA) 41
2.2.6 Exemplos de Ataques CPA (*Chosen-Plaintext Attack*) 45
2.2.7 Segurança Quanto à Recuperação da Mensagem 49
2.2.8 Segurança do CTR sob CPA (*Chosen-Plaintext Attack*) 52
2.2.9 Segurança do CBC sob CPA (*Chosen-Plaintext Attack*) 56
2.2.10 Indiscernibilidade sob CCA (*Chosen-Ciphertext Attack*) 57
2.2.11 Exemplos de CCA (*Chosen-Ciphertext Attack*) 58
2.3 Considerações Finais sobre o Modelo de Segurança 63

3 Segurança das Redes Locais sem Fios 64
3.1 *Wired Equivalent Privacy* - WEP 66
3.1.1 Encapsulamento do WEP 66
3.1.2 Desencapsulamento do WEP 69
3.1.3 Vulnerabilidades do WEP 70
3.2 *WI-FI Protected Access* - WPA 72
3.2.1 Encapsulamento do TKIP 73
3.2.2 Desencapsulamento do TKIP 79
3.2.3 Variantes do WPA 80
3.2.4 Vulnerabilidades do WPA 81
3.3 Padrão IEEE 802.11i ou WPA2 82
3.3.1 Encapsulamento do CCMP 82
3.3.2 Desencapsulamento do CCMP 87
3.3.3 Vulnerabilidades do WPA2 87
3.4 Padrão IEEE 802.1X 88
3.5 Ataques às Redes Locais sem Fios 89
3.5.1 Processo de Ataque às Redes Locais sem Fios 89
3.5.2 Ataques ao WEP 92
3.5.3 Ataques ao WPA 95
3.5.4 Ataques ao WPA2 96
3.5.5 Ataques ao Padrão IEEE 802.1X 96

3.5.6 Outros Ataques às Redes Locais sem Fios 99
3.6 Considerações Finais sobre a Segurança das Redes Locais sem Fios 101

4 Aplicação do Modelo às Redes Locais sem Fios 102
4.1 A Função Vantagem no Protocolo WEP 103
4.1.1 A Segurança do RC4 como uma PRP 103
4.1.2 Discernidor do RC4 segundo Mantin e Shamir 104
4.1.3 A Vantagem do Discernidor de Mantin e Shamir para o Algoritmo RC4 . . 107
4.1.4 Discernidor do RC4 segundo Paul e Preneel 111
4.1.5 A Vantagem do Discernidor de Paul e Preneel para o Algoritmo RC4 . . . 113
4.1.6 Recuperando a Chave do WEP 116
4.1.7 Avaliando os Modos do WEP 117
4.2 A Função Vantagem no Protocolo WPA 122
4.2.1 O Modo C-CTR com o Protocolo WPA 122
4.3 A Função Vantagem no Protocolo WPA2 123
4.3.1 Avaliando os Modos do Protocolo WPA2 123
4.4 Considerações Finais sobre a Aplicação do Modelo 124

5 Conclusões e Trabalhos Futuros 126
5.1 Conclusões 126
5.2 Trabalhos Futuros 127

Referências Bibliográficas 128

A Manutenção e Troca de Chaves nas Redes Locais sem Fios 131
A.1 Estabelecimento da Chave Mestra 131
A.2 Hierarquia das Chaves 132
A.2.1 Hierarquia das Chaves de *Pairwise* 132
A.2.2 Hierarquia das Chaves de *Group* 133
A.3 O *4-Way-Handshake* 134
A.4 O *Handshake* da Chave de Grupo 135

Lista de Figuras

2.1 Cifragem no modo ECB (*Electronic Codebook*) 29
2.2 Decifragem no modo ECB (*Electronic Codebook*) 30
2.3 Cifragem no modo CBC (*Chipher Block Chaining*) 31
2.4 Decifragem no modo CBC (*Chipher Block Chaining*) 31
2.5 Cifragem no modo CTR (*Counter*) 34
2.6 Decriptação no modo CTR (*Counter*) 35

3.1 Esquema de encapsulamento do protocolo WEP 67
3.2 Formato do quadro do WEP 67
3.3 Algoritmo interno do RC4 68
3.4 Esquema de desencapsulamento do protocolo WEP 69
3.5 Esquema do protocolo TKIP 74
3.6 Formato do quadro MAC do TKIP 75
3.7 Algoritmo de mistura da chave do TKIP 76
3.8 MSDU ajustado do Michael 78
3.9 Esquema do desencapsulamento do TKIP 80
3.10 Esquema de encapsulamento do protocolo CCMP 83
3.11 Formato do MPDU do CCMP 84
3.12 Esquema CBC-MAC do AES para calcular o MIC 85
3.13 A entrada do CBC-MAC do AES para calcular o MIC. 85
3.14 Bloco inicial de 128 bits para o cálculo do MIC. 85
3.15 Esquema do modo contador com o AES para cifragem dos dados 86
3.16 Valor inicial do contador de 128 bits para o CCTR-AES. 87
3.17 Desencapsulamento do CCMP 87

3.18 Controle de portas do 802.1X. 88
3.19 Fluxo de mensagens do 802.1X. 89

4.1 Os dois primeiros *rounds* do RC4 com $S_0[2] = 0$ e $S_0[1] \neq 2$. (a) posição inicial da geração das saídas, (b) a primeira saída e (c) a segunda saída. 104
4.2 Esquema criptográfico de exemplo de uso do RC4. 107
4.3 Os dois primeiros *rounds* do RC4 com $S_0[1] = 2$. (a) Posição inicial da geração das saídas. (b) A primeira saída. (c) A segunda saída. 112
4.4 O modo ECB com o protocolo WEP 118
4.5 O bloco do ECB do protocolo WEP formado por 2^{24} pacotes. 119
4.6 O modo C-CTR com o protocolo WEP 120
4.7 O modo C-CTR com o protocolo WPA. 122

A.1 Hierarquia das chaves de *Pairwise* 134
A.2 Hierarquia das chaves de *Group* 134
A.3 Esquema do *4-Way-Handshake* 135
A.4 Esquema do *Group Key Handshake* 136

Lista de Acrônimos

AAD	-	*Additional Authentication Data*
AES	-	*Advanced Encryption Standard*
AP	-	*Access Point*
CBC	-	*Cipher Block Chaining*
CCA	-	*Chosen-Ciphertext Attack*
CCPM	-	*Counter mode/CBC-MAC Protocol*
CHAP	-	*Challenge-Handshake Authentication Protocol*
CPA	-	*Chosen-Plaintext Attack*
CRC	-	*Cyclic redundancy check*
CTR	-	*Counter Mode*
DES	-	*Data Encryption Standard*
DHCP	-	*Dynamic Host Configuration Protocol*
DNS	-	*Domain Name System*
DoS	-	*Denial of Service*
EAP	-	*Extensible Authentication Protocol*
EAPOL	-	*EAP over LAN*
ECB	-	*Electronic Code Book*
GMK	-	*Group Master Key*
HTTP	-	*HyperText Transfer Protocol*
ICV	-	*Integrity Check Value*
IEEE	-	*Institute of Electrical and Electronics Engineers, Inc.*
IESG	-	*Internet Engineering Steering Group*
IETF	-	*Internet Engineering Task Force*
IND-CPA	-	Indiscernibilidade Sob *Chosen-Plaintext Attack*
IP	-	*Internet Protocol*
IV	-	*Initialization Vector*
KCK	-	EAPOL *Key Confirmation Key*
KEK	-	*EAPOL Key Encryption Key*
KSA	-	*Key Scheduling Algorithm*
LAN	-	*Local Area Network*
LEAP	-	*Lightweight Extensible Authentication Protocol*
MAC	-	*Medium Access Control*
MD5	-	*Message-Digest algorithm 5*

MIC	-	*Message Integrity Check*
MPDU	-	*MAC Protocol Data Unit*
MSDU	-	*MAC Service Data Unit*
NCAT	-	*Network Config Audit Tool*
OSA	-	*Open System Authentication*
PKI	-	*Public Key Infrastructures*
PMK	-	*Pairwise Master Key*
PMK	-	*Pairwise Master Key*
PN	-	*Packed Number*
PRF	-	*Pseudo-Random Function*
PRGA	-	*Pseudo Random Generation Algorithm*
PRGN	-	*Pseudo Random Number Generator*
PRP	-	*Pseudo-Random Permutation*
PSK	-	*Preshared Key*
PTK	-	*Pairwise Transient Key*
RADIUS	-	*Remote Authentication Dial In User Service*
RC4	-	*Rivest Cipher 4*
RNS	-	*Robust Security Network*
RSC	-	*Receive Sequence Counter*
SA ou AS	-	Servidor de Autenticação
SHA	-	*Secure Hash Algorithm*
SNAP	-	*SubNetwork Access Protocol*
SNMP	-	*Simple Network Management Protocol*
SOHO	-	*Small Office and Home Office*
SSID	-	*Service Set IDentifier*
SSL	-	*Secure Sockets Layer*
TA	-	*Transmitter Address*
TEK	-	*Temporal Encryption Key*
TG	-	*Task Group*
TI	-	Tecnologias da Informação
TK	-	*Temporal Key*
TKIP	-	*Temporal Key Integrity Protocol*
TLS	-	*Transport Layer Security*
TSC	-	TKIP *Sequence Counter*
TSN	-	*Transition Security Network*
TTAK	-	TKIP *mixed Transmit Address and Key*
VNC	-	*Virtual Network Computing*
VPN	-	*Virtual Private Network*
WEP	-	*Wired Equivalent Privacy*
WLAN	-	*Wireless Local Area NetWork*
WPA	-	*Wi-Fi Protected Access*

Lista de Símbolos

F	-	Uma família de funções
$Keys(F)$	-	O conjunto de chaves de F
$Dom(F)$	-	O domínio de F
$Range(F)$	-	O contradomínio de F
$\{0,1\}^L$	-	Uma *string* binária de tamanhos L
$K \xleftarrow{R} Keys(F)$	-	Denota a operação de selecionar uma *string* randômica de $Keys(F)$
$f \xleftarrow{R} f$	-	Faz de F a função F_K onde K é uma chave randômica
$Rand^{D \to R}$	-	A família de todas as funções de D para R
$Perm^D$	-	A família de todas as permutações em D
$P[X]$	-	Probabilidade de X
$P[X\|Y]$	-	Probabilidade de X dado Y
$Adv^{prf}_{F,A}$	-	A função vantagem prf de um algoritmo A contra a família de funções F
$Adv^{prf}_{F}(t,q,\mu)$	-	A função vantagem de qualquer algoritmo que seja limitado aos recursos indicados
g^{-1}	-	Se g é uma função então g^{-1} é a função inversa
$\|\|$	-	Significa uma operação de concatenação
$\oplus$	-	Significa uma operação binária XOR (OU Exclusivo)
1^L	-	Uma *string* binária de tamanho L com todos os bits setados em 1
0^L	-	Uma *string* binária de tamanho L com todos os bits setados em 0
$\mid Y \mid$	-	O tamanho de Y em bits
$\perp$	-	Símbolo retornado da função quando restrições são capturadas
$NtS_l(i)$	-	Denota a *string* de tamanho l-bits que é a representação binária do inteiro i (leia "*number to string*")
$StN(s)$	-	Denota o inteiro não negativo cuja representação é a *string* de bits s (leia "*string to number*").

Capítulo 1

Introdução

Nos padrões de comunicação de dados, como por exemplo o Padrão IEEE 802.11, a segurança é um tópico indispensável. Dados são transmitidos por algum meio e neste cenário algumas das principais preocupações dos usuários do padrão estão relacionadas à privacidade[1] e à autenticidade[2] dos dados.

O modelo de criptografia de chave privada, também conhecido como modelo simétrico, considera dois participantes que compartilham uma chave e usam esta chave para viabilizar a comunicação de dados com vários atributos de segurança. Os principais objetivos dessa segurança são a privacidade e a autenticidade dos dados transmitidos. Um esquema de criptografia simétrico (também conhecido como esquema de criptografia de chave privada) permite que os participantes, com posse de uma chave secreta, alcancem o objetivo de privacidade do dados. Um esquema de criptografia simétrico diz como os dados serão produzidos para a transmissão, como o receptor recuperará os dados e como será gerada a chave que deverá ser compartilhada [Goldwasser and Bellare 2001].

O único esquema de criptografia de dados considerado seguro chama-se *one-time-pad*, mas é impraticável, pois se utiliza de uma chave secreta de mesmo tamanho da mensagem a ser enviada, o que leva ao problema de como compartilhar uma chave tão grande.

Para esquemas criptográficos implementáveis, não é possível provar na prática que são realmente seguros. Pode-se, no entanto, levar em conta o fato de que um atacante é computacionalmente restrito e encontrar um conceito de segurança que, apesar de não ser perfeito, seja bom na prática.

[1] José Afonso da Silva define privacidade como o conjunto de informação acerca do indivíduo que ele pode decidir manter sob seu exclusivo controle, ou comunicar, decidindo a quem, quando, onde e em que condições, sem a isso poder ser legalmente sujeito. Define intimidade como a esfera secreta da vida do indivíduo na qual este tem o poder legal de evitar os demais. Define vida privada como o direito de o indivíduo viver sua própria vida, pois o segredo da vida privada é condição de expansão da personalidade [da Silva 2003].

[2] Assegura que a origem da mensagem ou documento eletrônico é identificada corretamente, com a certeza de que a identidade não é falsa [Stallings 1998].

Um modelo pode ajudar em tal tarefa. A modelagem permite moldar, analisar e avaliar as possibilidades, antes de investir na construção de algo real. O modelo não sofre influências de leis físicas e de limitações financeiras, que poderiam invalidar um projeto. Por outro lado, nem tudo o que é modelado é possível de ser construído, mas os modelos podem ter valor mesmo nestes casos [Goldwasser and Bellare 2001].

O modelo da função vantagem quanto a IND-CPA (Indiscernibilidade Sob *Chosen-Plaintext Attack*), de Goldwasser e Bellare, aplicado aqui ao Padrão IEEE 802.11, tem o poder de analisar a segurança de esquemas de criptografia e dizer se um esquema adiciona fraquezas ao algoritmo cifrador utilizado, ou se o esquema é seguro. Os modelos utilizados aqui não são dependentes de limitações da implementação ou da praticidade e por isso são bem eficazes e poderosos nas demonstrações de segurança. Isso significa que os modelos não consideram uma ou outra implementação, eles são genéricos no esquema, independente de como são implementados.

1.1 Objetivos e Organização da Dissertação

O objetivo do trabalho é fazer uma releitura do modelo da função vantagem e verificar sua aplicação em um padrão de segurança utilizando na prática, mostrando assim, que é possível garantir através deste modelo, aos usuários, o nível de segurança que o padrão propõe.

Esta dissertação está organizada da seguinte maneira:

- O Capítulo 2 define os conceitos de segurança que queremos alcançar, apresenta o modelo de Função Vantagem e como o modelo aborda tais conceitos.
- O Capítulo 3 apresenta os algoritmos de criptografia e integridade usados no Padrão IEEE 802.11, suas propriedades, vulnerabilidades e ataques conhecidos.
- O Capítulo 4 mostra como aplicar o modelo definido no Capítulo 2 aos padrões apresentados no Capítulo 3.
- O Capítulo 5 resume as contribuições e propõe tópicos para trabalhos futuros.

Capítulo 2

Modelo de Segurança

A abordagem que será apresentada neste capítulo tem a força de permitir-nos a análise de segurança de protocolos baseados em cifradores de bloco. Na seção 2.1, será apresentada a função vantagem das funções e permutações pseudo-randômicas usadas em cifradores de bloco. Na Seção 2.2, será discutida a função vantagem dos cifradores de bloco de chave privada. Este capítulo faz uma releitura do modelo da função vantagem e está fortemente baseado em [Goldwasser and Bellare 2001].

2.1 A Função Vantagem das PRFs e PRPs

As funções pseudo-randômicas (PRFs) e as permutações pseudo-randômicas (PRPs) são muito utilizadas no desenvolvimento de protocolos, principalmente para criptografia de chave privada (criptografia simétrica), sendo assim sua abordagem aqui contribui efetivamente no presente estudo. As propriedades básicas das PRFs e PRPs são apresentadas a seguir, sendo que em [Ostrovsky 1995], podem ser encontradas mais informações sobre funções pseudo-randômicas e como construí-las.

2.1.1 Famílias de Funções

Uma família de funções é um mapeamento $F : Keys(F) \times Dom(F) \rightarrow Range(F)$, onde:

- $Keys(F)$ é o conjunto de chaves de F;
- $Dom(F)$ é o domínio de F;
- e $Range(F)$ é o contradomínio de F.

Sejam $Keys(F) = \{0,1\}^k$, $Dom(F) = \{0,1\}^l$ e $Range(F) = \{0,1\}^L$, isto é, $Keys(F)$, $Dom(F)$ e $Range(F)$ são *strings* binárias de tamanhos k, l e L respectivamente. Considere F_K cada instância da família de funções F que é definida pela escolha randômica de uma chave $K \in Keys(F)$. Em outras palavras, cada chave K identifica ou aponta um membro dessa família. Assim, dado um K e um x como entrada, F_K retorna algo que é o mapeamento de cada valor de $x \in Dom(F)$, para um $y \in Range(F)$, tal que $y = F_K(x) = F(K, x)$. A expressão $K \xleftarrow{R} Keys(F)$ denota a operação de selecionar uma *string* randômica de $Keys(F)$. De forma análoga, $f \xleftarrow{R} F$ faz de f a função F_K onde K é uma chave randômica.

Se o domínio e o contradomínio são o mesmo conjunto, isto é, $l = L$, e a função é do tipo um-para-um dentro do mesmo conjunto, a função $F_K(x)$ é chamada de permutação. Assim, o *Data Encryption Standard* (DES) é uma família de permutações com $Keys(DES) = \{0,1\}^{56}$, $Dom(DES) = \{0,1\}^{64}$ e $Ranger(DES) = \{0,1\}^{64}$. O *Advanced Encryption Standard* (AES) com chaves de 128 bits é, também, uma família de permutações com $Keys(AES) = \{0,1\}^{128}$, $Dom(AES) = \{0,1\}^{128}$ e $Ranger(AES) = \{0,1\}^{128}$.

2.1.2 Funções Randômicas e Permutações Randômicas

Se $D = \{0,1\}^l$ e $R = \{0,1\}^L$, com l e L inteiros ≥ 1, então é possível definir duas famílias de funções:

- $Rand^{D \to R}$, a família de todas as funções de D para R;
- $Perm^D$, a família de todas as permutações em D.

O tipo de função randômica considerada aqui, de l-bits para L-bits, pode ser imaginada como uma caixa preta, a qual, dada uma entrada $x \in \{0,1\}^l$, retorna um número randômico. No entanto, se se repetir uma entrada anterior ela retorna o mesmo resultado da vez anterior. O algoritmo a seguir denota essa funcionalidade [Goldwasser and Bellare 2001]:

Se $T[x]$ não foi definido então
Jogue moedas para determinar uma *string* $y \in \{0,1\}^L$ e faça $T[x] \leftarrow y$
Retorne $T[x]$

O termo "função randômica", aqui, se refere à forma como a função é escolhida, e não que uma instância selecionada é randômica. Depois de selecionada de forma randômica, a função é constante

como outra qualquer. Os cálculos probabilísticos a seguir ajudam a entender as funções randômicas [Goldwasser and Bellare 2001]:

Exemplo 2. 1. *Dado* $x \in \{0,1\}^l$ *e* $y \in \{0,1\}^L$*. Então:*

$$P\left[f(x) = y : f \xleftarrow{R} Rand^{l \to L}\right] = 2^{-L}.$$

Neste caso, como a saída de $f(x)$ *é randômica, a probabilidade de essa saída ser igual a um determinado* y *é de* 1 *(um) sobre* 2^L *(o total de possibilidades).*

Agora, dado $x1, x2 \in \{0,1\}^l$ *e* $y \in \{0,1\}^L$*. Então:*

$$P\left[f(x1) = f(x2) = y : f \xleftarrow{R} Rand^{l \to L}\right] = \begin{cases} 2^{-2L} & \textit{se } x1 \neq x2 \\ 2^{-L} & \textit{se } x1 = x2 \end{cases}$$

Aqui, tanto a saída de $f(x1)$ *como a saída de* $f(x2)$ *são randômicas, então a probabilidade de essas saídas serem iguais a um determinado* y *depende da relação entre as duas saídas. Se as duas saídas são iguais então a probabilidade é a mesma do primeiro caso. Se as duas saídas são diferentes então é* $2^{-L} \times 2^{-L}$*.*

Finalmente, dado $x1, x2 \in \{0,1\}^l$*,* $y \in \{0,1\}^L$ *e sendo* $\oplus$ *uma operação XOR. Então:*

$$P\left[f(x1) \oplus f(x2) = y : f \xleftarrow{R} Rand^{l \to L}\right] = \begin{cases} 2^{-L} & \textit{se } x1 \neq x2 \\ 0 & \textit{se } x1 = x2 \textit{ e } y \neq 0^L \\ 1 & \textit{se } x1 = x2 \textit{ e } y = 0^L \end{cases}$$

Note que a probabilidade é independente de l, x ou y. A probabilidade é em função do contradomínio L, porque esse é o número de possibilidade para o mapeamento de um x qualquer, quando feito por uma função randômica, com uma distribuição uniforme.

2.1.3 Funções Pseudo-randômicas

Uma função pseudo-randômica é uma família de funções cujo comportamento de entrada e saída, de uma instância randômica da família, é "computacionalmente indistinguível" de uma função randômica. O cenário descrito a seguir representa essa definição.

Considere que $F : Keys(F) \times D \rightarrow R$ é uma família de funções. Imagine então alguém em uma sala com um computador conectado a um computador fora da sala. Ele pode teclar alguma coisa no seu computador e enviar para fora, e uma resposta é retornada. Ele pode teclar *strings* do domínio D e a resposta é uma *string* do contradomínio R. O computador fora da sala implementa uma função $g : D \rightarrow R$, de forma que a pessoa tecla x e recebe como resposta $g(x)$. Ele não pode ver o conteúdo de g, apenas o comportamento das entradas e saídas. Existem duas formas diferentes pelas quais g pode ser escolhido. Considere a escolha de g sendo dois mundos diferentes e possíveis [Goldwasser and Bellare 2001]:

- **Mundo 0**: a função g é randômica na forma $g \xleftarrow{R} Rand^{D \rightarrow R}$, isto é, g é somente uma função randômica de D para R, como descrito na seção anterior;
- **Mundo 1**: a função g é randômica na forma $g \xleftarrow{R} F$, isto significa que uma chave é escolhida via $K \xleftarrow{R} Keys(F)$ e então g é um conjunto para F_K.

Para diferenciar bem esses dois mundos, considere que funções randômicas, nas quais é baseado o **mundo 0**, são muito grandes para serem armazenadas e não são utilizadas na prática. O **mundo 1** propõe uma família de funções com aplicabilidade prática, mas limitadas quanto ao comportamento randômico. Alguém que tentar adivinhar em que mundo está, nesse cenário, dirá quanto a família de funções proposta se aproxima de uma função randômica verdadeira. A escolha do mundo é feita antes de a pessoa entrar na sala, antes de começar a digitar, e não muda durante uma sessão. O trabalho dela é descobrir em qual mundo foi colocada, isto é, descobrir se g é apenas randômica ou é um conjunto de funções, e seu único recurso é teclar valores de x e receber g(x).

Como resultado, a qualidade da família pseudo-randômica F é a medida da dificuldade de dizer quando está no **Mundo 0** ou no **Mundo 1**. Quanto mais dificuldade a pessoa tiver em fazer tal distinção, mais a família F se aproxima de uma verdadeira função randômica. Se não for possível distinguir o comportamento das entradas e saídas de uma instância randômica da família F, do comportamento das entradas e saídas de uma função randômica verdadeira, então, a função pseudo-randômica pode ser usada no lugar de uma função verdadeiramente randômica.

Para formalizar a tentativa de dizer em qual mundo a pessoa foi colocada, nós usamos um discernidor. Um discernidor é um algoritmo que provê um acesso do tipo oráculo a uma função g e tenta decidir se g é randômica (**mundo 0**) ou pseudo-randômica (**mundo 1**). Assim, A^g é um discernidor em que A tem um acesso do tipo oráculo à função g. Esse acesso é a única forma permitida

de interagir com a função g. O discernidor pode enviar entradas x e o oráculo, detentor da função g, devolve a saída correspondente a $g(x)$. Não é permitido, em hipótese alguma, ao discernidor, examinar a função diretamente. A definição a seguir captura matematicamente o uso do discernidor [Goldwasser and Bellare 2001]:

Definição 2. 2. *Seja $F : Keys(F) \times D \rightarrow R$ uma família de funções e A um algoritmo que tem um acesso do tipo oráculo para a função $g : D \rightarrow R$ e retorna um bit. Considere dois experimentos:*

Experimento Exp $^{prf-1}_{F,A}$	*Experimento Exp* $^{prf-0}_{F,A}$
$K \xleftarrow{R} Keys(F)$	$g \xleftarrow{R} Rand^{D \rightarrow R}$
$d \leftarrow A^{F_K}$	$d \leftarrow A^g$
Responda d	*Responda* d

*A **vantagem-prf** de A é:*

$$Adv^{prf}_{F,A} = P[Exp^{prf-1}_{F,A} = 1] - P[Exp^{prf-0}_{F,A} = 1] \tag{2.1}$$

Para qualquer t, q, μ a vantagem-prf de F é:

$$Adv^{prf}_{F}(t, q, \mu) = max_A\{Adv^{prf}_{F,A}\}, \tag{2.2}$$

ou seja, o máximo sobre todos os algoritmos A que tenham complexidade de tempo = t, façam no máximo q consultas ao oráculo e a soma dos tamanhos dessas consultas seja μ bits.

Na definição acima o primeiro experimento executa A com o oráculo $g = F_K$, e isso representa o **mundo 1**. Já o segundo experimento executa A como o oráculo $g : D \rightarrow R$, sendo g uma função randômica, e isso representa o **mundo 0**. O bit d retornado por A significa a opção de A sobre em qual mundo ele está. O algoritmo A é a representação da pessoa que está dentro da sala, enviando entradas x para o computador que está fora da sala (o oráculo), tentando adivinhar qual g o oráculo está usando para calcular $g(x)$.

A **vantagem-prf** mede a vantagem que o adversário A alcança na tentativa de adivinhar em qual mundo está. Essa vantagem é a probabilidade de A acertar. Os dois experimentos são completamente diferentes, por isso são avaliados separadamente. A equação 2.1 avalia a diferença entre as probabilidades de cada um dos experimentos retornar 1 e significa a probabilidade de A quebrar o esquema F. Se A está fazendo um bom trabalho ele irá retornar 1 mais freqüentemente no primeiro

experimento do que no segundo. Observe que no primeiro experimento é avaliada a probabilidade de A retornar 1, isto é, a probabilidade de A acertar. No segundo experimento, é avaliada também a probabilidade de A retornar 1, mas nesse caso significa a probabilidade de A errar.

A diferença entre as probabilidades dos experimentos retornarem 1 é a vantagem-prf de A. Em outras palavras, a probabilidade de A estar no mundo 1 e dizer que está no mundo 1 (A acertar) menos a probabilidade de A estar no mundo 0 e dizer que está no mundo 1 (A errar). A Equação 2.2 diz que o que interessa é a vantagem do discernidor mais "inteligente" de todos os que são restritos às limitações de recursos. Informalmente, uma família F é uma PRF segura se $Adv_F^{prf}(t, q, \mu)$ é pequena para valores práticos dos parâmetros dos recursos.

Um discernidor pode conseguir uma maior vantagem do que outro por duas razões:

- um discernidor é mais inteligente do que outro,
- um discernidor faz mais consultas e gasta mais tempo processando as respostas, ou seja, quanto mais entradas e saídas ele vê, maior será sua habilidade em dizer em qual mundo está.

Por causa do último motivo, a segurança da família F deve ser medida em função dos recursos permitidos para o atacante. O objetivo é saber, para certas limitações de recursos, qual a vantagem do discernidor mais "inteligente" sobre todos os que são restritos a essas limitações de recursos. Em outras palavras, qual a probabilidade máxima de "quebrar" o esquema F se um atacante está restrito a certos recursos.

A equação 2.2 atribui à família F uma função vantagem-prf que tem como parâmetros de entrada os valores dos recursos permitidos e retorna a vantagem-prf máxima que um adversário (o mais inteligente) limitado a tais recursos consegue obter. E essa **função vantagem** representa a segurança da família F quando usada como uma PRF. Aqui foram considerados três recursos: a complexidade de tempo t de A, um máximo q de consultas permitidas ao oráculo e a soma dos tamanhos dessa consultas μ medida em bits. No entanto, a escolha dos recursos a serem considerados podem variar.

A complexidade de tempo de A refere-se ao máximo do tempo de execução dos dois experimentos mais o tamanho do código de A. Considere um modelo de *Random Access Machine* (RAM) de computação, como os modelos usados nas disciplinas de algoritmos para medir o tempo de execução de um algoritmo. No primeiro experimento, o tempo de escolha da chave K como randômica é o tempo de calcular o valor de $F_K(x)$ para qualquer consulta feita por A ao oráculo. No segundo ex-

perimento, deve-se contar o tempo para escolher a função randômica g, incluindo o custo de manter a tabela de valores na forma $(x, g(x))$. O número de consultas feitas por A é o número de entradas e saídas de exemplo que ele vê. Como nem todas as *strings* no domínio precisam ter o mesmo tamanho, soma-se os tamanhos de todas as consultas feitas.

O tamanho da chave não aparece na função da vantagem porque ela não faz parte do objetivo que é só a vantagem que um discernidor pode obter. É verdade que o tamanho da chave influencia na determinação da segurança de F e a função vantagem é em função de k, mas aqui nesse momento não se sabe quem pode ser F, então não se pode determinar. Segundo [Goldwasser and Bellare 2001], "cifradores de blocos bem desenvolvidos deveriam ter funções vantagem com valores próximos de $t/2^k$, mas na prática os cifradores não são tão bons assim".

Essa definição é forte porque não faz qualquer referência as estratégias usadas pelo discernidor, isto é, o discernidor pode fazer o que achar necessário para discernir entre os dois mundos, desde que respeite as limitações de recursos. Tanto aqui como na vida real, a segurança é definida em função dos recursos investidos por um adversário. Todos os sistemas modernos de criptografia são quebráveis em princípio, a questão é então quanto tempo será gasto para tal tarefa. Sendo assim, intuitivamente, considera-se que F é segura, se o valor da função vantagem for "baixo" para valores práticos dos parâmetros de entrada [Goldwasser and Bellare 2001].

2.1.4 Permutações Pseudo-randômicas

As considerações da seção anterior são as mesmas para permutações pseudo-randômicas. Então pode-se proceder exatamente como em funções pseudo-randômicas, mas trocando $Rand^{D \to R}$ para $Perm^D$.

Como já foi dito nas seções anteriores, um cifrador de blocos F é uma família de permutações, isto é, cada instância F_K da família é uma permutação. Assim, dois tipos de ataque podem ser considerados. No primeiro, *chosen-plaintext attack* (CPA), o adversário tem acesso a um oráculo para a função g. No segundo, *chosen-ciphertext attack* (CCA), o adversário tem acesso além do oráculo anterior, a um oráculo para a função g^{-1}.

A Função Vantagem de Ataques tipo CPA contra uma PRP

Dada uma família de funções $F : Keys(F) \times D \to D$, cujo caso mais comum é $Keys(F) = \{0,1\}^k$, $D = \{0,1\}^l$ e F é uma família de permutações. Considere um adversário A em uma sala com acesso

a um oráculo para a função g escolhida de uma das duas formas [Goldwasser and Bellare 2001]:

- **Mundo 0**: a função g é randômica da forma $g \xleftarrow{R} Perm^D$, isto é, g é uma permutação randômica em D.
- **Mundo 1**: a função g é randômica da forma $g \xleftarrow{R} F$, isto significa que uma chave é escolhida via $K \xleftarrow{R} Keys(F)$ e então g é o F_K.

Considere dois experimentos:

Experimento Exp $_{F,A}^{prp-cpa-1}$	Experimento Exp $_{F,A}^{prp-cpa-0}$
$K \xleftarrow{R} Keys(F)$	$g \xleftarrow{R} Perm^D$
$d \leftarrow A^{F_K}$	$d \leftarrow A^g$
Responda d	Responda d

A **vantagem-cpa-prp** de A será definida assim:

$$Adv_{F,A}^{prp-cpa} = P[Exp_{F,A}^{prp-cpa-1} = 1] - P[Exp_{F,A}^{prp-cpa-0} = 1]$$

Para qualquer t, q, μ a vantagem-cpa-prp de F é:

$$Adv_F^{prp-cpa}(t, q, \mu) = max_A\{Adv_{F,A}^{prp-cpa}\}, \tag{2.3}$$

ou seja, o máximo sobre todos os algoritmos A que tenham complexidade de tempo = t, façam no máximo q consultas ao oráculo e a soma dos tamanhos dessas consultas seja μ bits.

Como já foi dito, a diferença entre as probabilidades dos experimentos retornarem 1 é a vantagem-cpa-prp de A. Em outras palavras, a probabilidade de A estar no mundo 1 e dizer que está no mundo 1 (A acertar) menos a probabilidade de A estar no mundo 0 e dizer que está no mundo 1 (A errar). A Equação 2.3 diz que o que interessa é a vantagem do discernidor mais "inteligente" de todos os que são restritos às limitações de recursos. Informalmente, uma família F é uma PRP segura sob CPA se $Adv_F^{prp-cpa}(t, q, \mu)$ é pequena para valores práticos dos parâmetros dos recurso.

A Função Vantagem de Ataques tipo CCA contra uma PRP

Dada uma família de funções $F : Keys(F) \times D \rightarrow D$, cujo caso mais comum é $Keys(F) = \{0, 1\}^k$ e $D = \{0, 1\}^l$ e F é uma família de permutações, vamos considerar um adversário A em uma sala

com acesso a dois oráculos, um para a função g e outro para a seu inverso, a função g^{-1}. A função g é escolhida de uma das duas formas [Goldwasser and Bellare 2001]:

- **Mundo 0**: a função g é randômica da forma $g \stackrel{R}{\leftarrow} Perm^D$, isto é, g é uma permutação randômica em D.
- **Mundo 1**: a função g é randômica da forma $g \stackrel{R}{\leftarrow} F$, isto significa que uma chave é escolhida via $K \stackrel{R}{\leftarrow} Keys(F)$ e então g é o F_K.

Uma vez definida g, então g^{-1} é automaticamente definida como sendo seu inverso. Prover ao atacante acesso a esses oráculos, tanto de cifragem como de decifragem, modela um ataque do tipo *Chosen-Ciphertext Attack* (CCA), no qual o adversário pode alterar textos cifrados e enviá-los para decifrar e então observar o seu comportamento [Bellare 1998]. Os dois experimentos a considerar são [Goldwasser and Bellare 2001]:

Experimento Exp $_{F,A}^{prp-cca-1}$	Experimento Exp $_{F,A}^{prp-cca-0}$
$K \stackrel{R}{\leftarrow} Keys(F)$	$g \stackrel{R}{\leftarrow} Perm^D$
$d \leftarrow A^{F_K, F_K^{-1}}$	$d \leftarrow A^{g, g^{-1}}$
Responda d	Responda d

A **vantagem-cca-prp** de A é:

$$Adv_{F,A}^{prp-cca} = P[Exp_{F,A}^{prp-cca-1} = 1] - P[Exp_{F,A}^{prp-cca-0} = 1]$$

Para qualquer $t, q_e, \mu_e, q_d, \mu_d$ a vantagem-cca-prp de F é:

$$Adv_F^{prp-cca}(t, q_e, \mu_e, q_d, \mu_d) = max_A\{Adv_{F,A}^{prp-cca}\},$$

ou seja, o máximo sobre todos os algoritmos A que tenham complexidade de tempo = t, façam no máximo q_e consultas ao oráculo g e a soma dos tamanhos dessas consultas seja μ_e bits. Eles também fazem no máximo q_d ao oráculo g^{-1} e a soma dos tamanhos dessas consultas deve ser μ_d bits. Nesses experimentos, A recebe dois oráculos e pode consultá-los enviando mensagens não cifradas e recebendo textos cifrados, ou enviando textos cifrados adaptados e recebendo mensagens decifradas como respostas. Baseado no comportamento observado, A toma uma decisão de em qual mundo está. A diferença entre a probabilidade de A acertar estando no mundo 1 menos a probabilidade de A errar estando no mundo 0 é a função vantagem de A contra a PRP sob CCA.

Informalmente, uma família F é uma PRP segura sob CCA se $Adv_F^{prp-cca}(t, q_e, \mu_e, q_d, \mu_d)$ é pequena para valores práticos dos parâmetros dos recursos.

Relações entre as definições

Sem consultar o último oráculo, o adversário está efetivamente montando um ataque CPA. Considere $F : Keys(F) \times D \rightarrow D$, uma família de permutações. Então [Goldwasser and Bellare 2001]:

$$Adv_F^{prp-cpa}(t, q, \mu) = Adv_F^{prp-cca}(t, q, \mu, 0, 0)$$

2.1.5 Seqüências de Famílias de PRFs e PRPs

Uma seqüência de famílias de funções é uma seqüência F^1, F^2,..., isto é, $\{F^n\}_{n\geq 1}$, onde cada F^n é uma família de funções com tamanhos de entrada $l(n)$, de saída $L(n)$ e de chave $k(n)$ em função do parâmetro de segurança n, ou seja, as famílias se diferenciam umas das outras por possuírem diferentes tamanhos de entrada, saída e chave.

Para modelar cifradores de blocos, as famílias já são consideradas a abstração apropriada. Uma das motivações para considerar seqüências de famílias é que a segurança pode ser definida assintoticamente, sendo mais conveniente, particularmente, porque neste caso podemos definir uma melhor noção de segurança, ao invés de medidas de insegurança [Goldwasser and Bellare 2001].

2.1.6 Usando PRFs e PRPs

Aqui são apresentadas as motivações para estudar os conceitos de PRFs e PRPs.

O Modelo de Função Randômica Compartilhada

Na criptografia simétrica, Alice e Bob compartilham uma chave secreta K. Eles querem criptografar e autenticar os dados enviados de um para o outro. Se a chave é muito grande, como uma função randômica f de l bits para L bits, então isso é conhecido como modelo de função randômica compartilhada. Como funções randômicas são muito grandes para serem armazenadas, o modelo de função randômica compartilhada não pode ser realizado na prática. Ele é apenas um modelo conceitual. Ele é muito bom para se pensar em criptografia, formular esquemas e analisá-los. A prova de segurança aqui é absoluta, não é necessário fazer qualquer restrição do poder computacional do

adversário. No modo de operação CTR (*Counter Mode*), por exemplo, se for utilizado uma função randômica f, ele funciona como um sistema criptográfico *one-time-pad*. Como será visto na Seção 2.2.2, o esquema é seguro mas não pode ser realizado de forma eficiente. Entretanto, instâncias de PRFs podem ser usadas no lugar de funções randômicas no esquema de chave compartilhada. A vantagem é que uma instância de uma função pseudo-randômica é especificada por uma chave curta K, e as partes necessitam somente guardar esta chave .

Não é possível usar funções pseudo-randômicas em substituição a funções randômicas, não em todos os casos, pois nem sempre elas funcionam bem. Mas, se usadas corretamente, elas podem funcionar bem em um grande número de casos. A idéia é desenvolver esquemas de criptografia, autenticação e outros, usando o modelo de função randômica compartilhada, e então substituir a função randômica por uma função pseudo-randômica, mantendo a segurança [Goldwasser and Bellare 2001].

Modelando Cifradores de Bloco

Uma das principais motivações para estudar PRFs e PRPs é modelar cifradores de bloco e assim poder analisar os protocolos de tais cifradores.

Muito já se falou sobre análise de protocolos quanto à recuperação da chave. Entretanto, um adversário pode conseguir prejudicar as partes envolvidas na transmissão das mensagens sem recuperar a chave, como será visto a seguir. Então, como a dificuldade de recuperação da chave não é suficiente para a segurança, é necessário ter uma propriedade de segurança máxima de um cifrador de bloco. Isto é, o cifrador de bloco deve ser uma PRP segura tanto para CPA quanto para CCA.

Não é possível provar, pode-se apenas fazer conjecturas sobre a função vantagem de vários cifradores de bloco, como por exemplo [Goldwasser and Bellare 2001]:

$$Adv_{DES}^{prp-cpa}(t, q, 64q) = c1.\frac{t/T_{DES}}{2^{55}} + c2.\frac{q}{2^{40}}, \tag{2.4}$$

onde t é a complexidade de tempo do adversário (o número de computações que ele faz), T_{DES} é o tempo de fazer uma computação do DES, 2^{55} é o custo do ataque de força bruta contra o DES, q é o número de consultas que o adversário faz, 2^{40} é o custo do ataque de criptoanálise linear contra o DES e c_1 e c_2 são algumas constantes. O primeiro termo da Equação 2.4 diz que se t é suficiente para 2^{55} computações do DES então este termo é igual a 1 e o adversário terá sucesso com um ataque de força bruta, ou senão, a vantagem desse adversário é menor que 1. O segundo termo diz que se o adversário faz 2^{40} consultas este termo é igual a 1 e ele terá sucesso em um ataque de criptoanálise

linear, ou senão a vantagem é menor que 1.

No que diz respeito ao AES, pode-se conjecturar:

$$Adv_{AES}^{prp-cpa}(t, q, 128q) = c1.\frac{t/T_{AES}}{2^{128}} + c2.\frac{q}{2^{128}}, \tag{2.5}$$

baseando-se na idéia de que os melhores ataques são a força bruta ou criptoanálise linear.

Já para $Adv_{DES}^{prf}(t, q)$, o melhor que podemos fazer é assumir que:

$$Adv_{DES}^{prf}(t, q, 64q) = c1.\frac{t/T_{DES}}{2^{55}} + \frac{q^2}{2^{64}} \tag{2.6}$$

$$Adv_{AES}^{prf}(t, q, 128q) = c1.\frac{t/T_{AES}}{2^{128}} + \frac{q^2}{2^{128}}, \tag{2.7}$$

baseados no Ataque do Aniversário que é discutido na Seção 2.1.9 [Goldwasser and Bellare 2001] [Bellare and Rogaway 2004].

Ao contrário das PRPs, as PRFs são vulneráveis ao ataque do aniversário. Para executar o ataque do aniversário com sucesso são necessárias $\sqrt{2^l}$ consultas, então nas Equações 2.6 e 2.7, se q é igual a $\sqrt{2^l}$ então o segundo termo é igual a 1, ou senão a vantagem é menor que 1.

As próximas seções mostrarão que dizer que um cifrador de blocos é uma PRF segura é mais forte que dizer que ele é seguro quanto à recuperação da chave. Ou seja, se um cifrador de blocos é "quebrado" como uma PRF então ele deve ser considerado inseguro e deve ser substituído.

2.1.7 Exemplos de Ataques

Os exemplos aqui apresentados têm o objetivo de ilustrar, e assim melhorar o entendimento dos modelos apresentados. Diferentes adversários para algumas variações de modelos são apresentados, possibilitando calcular as primeiras funções vantagem e analisar os resultados.

Exemplo 2. 3. *Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$, uma família de funções e a chave K uma matriz de bits de L linhas por l colunas. A primeira coluna é feita dos primeiros L bits de K, os L bits seguintes formam a segunda coluna e assim por diante. Se x é uma* string *de l bits de entrada, então, $F(K, x)$ é o produto das matrizes mapeado conforme ilustrado a seguir*

[Goldwasser and Bellare 2001]:

$$F_K(x) = \begin{bmatrix} K[1,1] & K[1,2] & ... & K[1,l] \\ K[2,1] & K[2,2] & ... & K[2,l] \\ . & & & \\ K[L,1] & K[L,2] & ... & K[L,l] \end{bmatrix} . \begin{bmatrix} x[1] \\ x[2] \\ ... \\ x[l] \end{bmatrix} = \begin{bmatrix} y[1] \\ y[2] \\ ... \\ y[L] \end{bmatrix}$$

Os cálculos de mapeamento são feitos como segue:

$$\begin{aligned} y[1] &= K[1,1].x[1] \oplus K[1,2].x[2] \oplus ... \oplus K[1,l].x[l] \\ y[2] &= K[2,1].x[1] \oplus K[2,2].x[2] \oplus ... \oplus K[2,l].x[l] \\ ... &= ... \\ y[L] &= K[L,1].x[1] \oplus K[L,2].x[2] \oplus ... \oplus K[L,l].x[l] \end{aligned}$$

Deseja-se saber se F é seguro. A resposta é não, porque é possível desenvolver um algoritmo adversário A que consegue uma vantagem grande (próximo de 1) em discernir entre dois mundos. Observe que qualquer chave K com $x = 0^l$ resulta em $y = 0^L$, ou seja, $F_K(0^l) = 0^L$. Isso representa uma fraqueza, já que uma função randômica de l-bits para L-bits dificilmente retornará 0^L para uma entrada 0^l.

O adversário D abaixo acessa um oráculo $g : \{0,1\}^l \rightarrow \{0,1\}^L$ e retorna um bit que, como na Seção 2.1.3, representa a opinião de D sobre em qual mundo ele está.

Adversário D^g
Faça $y \leftarrow g(0^l)$
Se $y = 0^L$ então responda 1 senão responda 0

Esse adversário consulta o oráculo sobre o ponto 0^l e atribui a y. Se $y = 0^L$, ele conclui que g é um instância da familia F, e se $y \neq 0^L$ ele responde que g é uma função randômica. É possível dizer se D faz um bom trabalho em distinguir os dois mundos calculando a vantagem que o adversário D consegue obter nessa tarefa. Primeiro, considere cada experimento em separado, como apresentado na seção 2.1.3.

$$\begin{aligned} P\left[Exp_{F,D}^{prf-1} = 1\right] &= 1 \\ P\left[Exp_{F,D}^{prf-0} = 1\right] &= 2^{-L} \end{aligned}$$

O primeiro experimento considera a probabilidade de estando D no mundo 1, isto é, $g = F_K$, D retornar 1 e acertar. Essa probabilidade é de exatamente 1, já que o resultado de $F_K(0^l)$ é sempre 0^L. No segundo experimento, é considerada a probabilidade de estando D no mundo 0, isto é, g é uma função randômica, D retornar 1 e errar. Essa probabilidade é de 2^{-L}. A vantagem de D é exatamente a diferença entre as duas probabilidades.

$$\begin{aligned} Adv_{F,D}^{prf} &= P\left[Exp_{F,D}^{prf-1} = 1\right] - P\left[Exp_{F,D}^{prf-0} = 1\right] \\ &= 1 - 2^{-L}. \end{aligned}$$

Novamente, a vantagem-prf de F é a vantagem que o melhor adversário, limitado a determinados recursos, consegue obter. Então, sendo t a complexidade de tempo de D. Isto é, $O(l + L)$ mais o tempo de um cálculo de F, então $O(l^2L)$. O número de consultas feitas por D é apenas uma e o tamanho total de todas as consultas é l, então, a vantagem-prf de F é:

$$\begin{aligned} Adv_F^{prf}(t, 1, l) &= \max_A \left\{Adv_{F,A}^{prf}\right\} \\ &\geq Adv_{F,D}^{prf} \\ &\geq 1 - 2^{-L} \end{aligned}$$

A primeira inequação é verdade porque o adversário D é um dos membros de conjunto de adversários A com recursos limitados a $(t, 1, l)$. A conclusão é que a função vantagem de F como uma PRF é muito alta para poucos recursos de computação do adversário. Assim, F é muito insegura como uma PRF.

Exemplo 2. 4. *Este exemplo aborda a seguinte questão: se usarmos uma família de funções, que é sabida segura, para desenvolver uma nova família de funções, esta última também é segura? Em outras palavras, a nova família mantém as característica de segurança da primeira?*

Suponha que $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ é uma PRF segura. Esse exemplo usa F para desenvolver uma PRF $G : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^{2L}$. A entrada de G é de mesmo tamanho que a de F, mas a saída de G é duas vezes o tamanho da saída de F. Considere a seguinte construção [Goldwasser and Bellare 2001]:

$$G_K(x) = F_K(x) \parallel F_K(\bar{x}).$$

Aqui "$\|$" significa a concatenação de strings*, e $\bar{x}$ é o complemento bit a bit da* string x*. A resposta que deve ser dada é se G é uma PRF segura, já que F é segura. A resposta é que G não é uma PRF segura, apesar da qualidade de F, como é demonstrado a seguir.*

Seja um adversário D com acesso a um oráculo para a função g que mapeia l bits para $2L$ bits. No mundo 0, g será escolhida como uma função randômica, enquanto no mundo 1, g será feita G_K, onde K é uma chave randômica. O adversário D deve descobrir em qual mundo ele foi colocado.

Adversário D^g
Faça $y_1 \leftarrow g(1^l)$
Faça $y_2 \leftarrow g(0^l)$
Considere y_1 *como* $y_1 = y_{1,1} \parallel y_{1,2}$ *com* $\mid y_{1,1} \mid = \mid y_{1,2} \mid = L$
Considere y_2 *como* $y_2 = y_{2,1} \parallel y_{2,2}$ *com* $\mid y_{2,1} \mid = \mid y_{2,2} \mid = L$
Se $y_{1,1} = y_{2,2}$ *então responda* 1 *senão responda* 0

O adversário pergunta ao oráculo sobre o ponto 1^l e recebe como resposta y_1, depois ele pergunta sobre o ponto 0^l e recebe de volta y_2. Ele retorna 1 se a primeira metade do y_1 é igual a segunda metade do y_2. Então é possível concluir que:

$$\begin{aligned} P\left[Exp_{G,D}^{prf-1} = 1\right] &= 1 \\ P\left[Exp_{G,D}^{prf-0} = 1\right] &= 2^{-L} \end{aligned}$$

Note que no experimento 1:

$$G_K(1^l) = F_K(1^l) \parallel F_K(0^l)$$
$$G_K(0^l) = F_K(0^l) \parallel F_K(1^l)$$

Então D retorna 1.

Já no experimento 2, $g(1^l)$ e $g(0^l)$ são ambas strings *de $2L$ bit randômicas e indepedentes. Assim a probabilidade de a primeira metade de y_1 ser igual a segunda metade de y_2 é a mesma de, em duas escolhas, as* strings *serem iguais, isto é, 2^{-L}. Então, subtraindo:*

$$\begin{aligned} Adv_{G,D}^{prf} &= P\left[Exp_{G,D}^{prf-1} = 1\right] - P\left[Exp_{G,D}^{prf-0} = 1\right] \\ &= 1 - 2^{-L}. \end{aligned}$$

Considere t a complexidade de D, isto é, $O(l+L)$ mais o tempo de calcular duas vezes G, ou seja, $O(l+L)$ mais o tempo de calcular quatro vezes F. O número de consultas feitas por D é dois e o tamanho total das consultas é $2l$. Então:

$$\begin{aligned} Adv_G^{prf}(t,2,2l) &= \max_A \left\{ Adv_{G,A}^{prf} \right\} \\ &\geq Adv_{G,D}^{prf} \\ &\geq 1 - 2^{-L} \end{aligned}$$

Assim, é possível concluir que G é muito insegura como uma PRF, mesmo sendo F muito segura, pois a função vantagem de G como PRF é muito alta para valores bem pequenos de recursos.

2.1.8 Segurança Quanto à Recuperação da Chave

Apesar de a segurança quanto à recuperação da chave primária não ser suficiente como um conceito de segurança, ela é necessária. Os conceitos de PRF e PRP são mais indicados e a razão disso é que esses conceitos são mais viáveis de se provar. No entanto, para que seja possível usar conceitos de PRF e PRP no lugar de segurança quanto à recuperação da chave, é necessário mostrar que se uma família de funções é insegura quanto à recuperação da chave, ela também é insegura como uma PRF ou uma PRP. A segurança quanto à recuperação da chave considera um adversário que, baseado em exemplos de entradas e de saídas de uma instância F_K da família F, tenta encontrar K. A função vantagem é a probabilidade que ele tem de sucesso. Considere um experimento em que um adversário tenta descobrir K. Ele consulta um oráculo para uma instância g de uma família de funções $F : Keys(F) \times D \rightarrow R$, e retorna uma *string* que considerar ser a chave [Goldwasser and Bellare 2001]:

Experimento Exp $_{F,B}^{kr}$
$K \xleftarrow{R} Keys(F)$
$K' \leftarrow B^{F_K}$
se $K = K'$ então responda 1 senão responda 0

A **vantagem-kr** de B é:

$$Adv_{F,B}^{kr} = P\left[Exp_{F,B}^{kr} = 1\right].$$

Para qualquer t, q, μ a vantagem-kr de F é:

$$Adv_F^{kr}(t, q, \mu) \quad = \quad \max_B \left\{ Adv_{F,B}^{kr} \right\},$$

isto é, a máxima vantagem de qualquer adversário que tenha complexidade de tempo t, faça não mais que q consultas ao oráculo e a soma dos tamanhos dessas consultas seja μ bits.

Essa definição captura todos os tipos de ataque de recuperação de chave, porque todos eles têm o mesmo objetivo de procurar a chave K baseados em alguma quantidade de exemplos de entradas e de saídas de uma instância F_K do cifrador. Para ilustrar, a seguir as implicações dos ataques clássicos de recuperação de chave no DES. Assuma que um ataque de força bruta sempre tem sucesso baseado no teste de dois exemplos, isto é:

$$Adv_{DES}^{kr}(t, 2, 2 \cdot 64) = 1.$$

Considere t aproximadamente 2^{55} vezes o tempo T_{DES} (tempo de uma computação do DES). Por outro lado, usando criptoanálise linear:

$$Adv_{DES}^{kr}(t, 2^{43}, 2^{43} \cdot 64) = 1$$

para t aproximadamente $2^{43} \times T_{DES}$. Segue um exemplo mais concreto, baseado no exemplo 2.3.

Exemplo 2. 5. *Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ a família de funções do exemplo 2.3, no qual a vantagem-prf foi muito alta. O objetivo agora é calcular a vantagem-kr. Isto é, a vantagem de um adversário B que recupera a chave. Considere e_j uma* string *de l-bit em que a posição j é igual a 1 e todas as outras posições são iguais a zero [Goldwasser and Bellare 2001].*

Adversário B^{F_K}
  Faça K' ser uma string *vazia*
  Para $j = 1, ..., l$ faça
    $y_i \leftarrow F_K(e_j)$
    $K' \leftarrow K' \parallel y_i$
  FimPara
  Responda K'

O adversário chama o oráculo para calcular a saída da função para a entrada e_j. O resultado y_i é exatamente a j-ésima coluna da matriz associada a chave K. As entradas da matriz são concatenadas para formar K', que é retornado como a chave. Como o adversário sempre descobre a chave então:

$$Adv_{F,B}^{kr} = 1$$

e como a complexidade de tempo do adversário é $t = O(l^2 L)$, porque ele faz $q = l$ chamadas para o oráculo e cada cálculo de F_K demora $O(lL)$, então:

$$Adv_F^{kr}(t, l, l^2) = 1$$

Como os parâmetros são pequenos (l é 64 ou 128) F é inseguro quanto à recuperação da chave.

Pode-se notar que F é menos seguro quanto à PRF do que quanto à recuperação da chave, porque a vantagem-prf tem um valor próximo de 1 para valores dos parâmetros muito menores que esses acima. A proposição abaixo diz que, para quaisquer valores de parâmetros de recursos, a vantagem-kr de uma família não pode ser significativamente maior que sua vantagem-cpa-prp ou prf. Isso mostra que se um cifrador de blocos é uma PRF ou PRP segura então ele também é seguro contra qualquer ataque de recuperação de chave.

Proposição 2. 6. *Seja $F : \{0,1\}^k \times \{0,1\}^l \to \{0,1\}^L$ uma família de funções. Então para qualquer t, q com $q < 2^l$ tem-se:*

$$Adv_F^{kr}(t, q, ql) \leq Adv_F^{prf}(t', q+1, (q+1)l) + \frac{1}{2^L} \tag{2.8}$$

e conseqüentemente, if $L = l$, então também:

$$Adv_F^{kr}(t, q, ql) \leq Adv_F^{prp-cpa}(t', q+1, (q+1)l) + \frac{1}{2^L - q} \tag{2.9}$$

onde t' é igual a t mais o tempo de calcular um F, já que o lado direito da equação faz uma consulta de tamanho l a mais [Goldwasser and Bellare 2001].

A prova introduz a idéia central de reduções, isto mostra a transformação $B \to A_B$, ou seja,

transformar qualquer adversário-kr B em um adversário-prf A_B tal que:

$$Adv^{kr}_{F,B} \leq Adv^{prf}_{F,A_B} + \frac{1}{2^L}. \tag{2.10}$$

A idéia por trás da redução é que A_B irá executar B como uma sub-rotina e usará as saídas de B para resolver seu próprio problema. O problema de A_B é descobrir se foi colocado no mundo 0 ou no mundo 1. Já B, é um algoritmo que pensa estar em um mundo onde existe um oráculo F_K, e ele tenta descobrir K via consultas ao oráculo. A_B executa B, este último produz algum K'. A_B pode testar se $F(K', x)$ é igual a $g(x)$ para algum valor de x que B não tenha consultado ao oráculo. O problema é que, se A_B usar um valor que B já tenha consultado, $g(x)$ irá retornar o mesmo valor que retornou para B, sendo g randômica ou não, como foi explicado na seção 2.1.2, e, por isso, a proposição afirma que $q < 2^l$, sobrando assim algum valor no domínio $D = \{0,1\}^l$ para a conclusão de A_B. Se $F(K', x)$ é igual a $g(x)$, A_B afirma que g é uma instância de F e, se não, g é randômico. A_B irá executar B e o próprio A_B irá suprir B com respostas às consultas de B, que se comportará como se fossem respostas do oráculo. Se A_B está no mundo 1, $g(x) = F_K(x)$ e então B funcionará como esperado. Se A_B está no mundo 0, g é uma função randômica e B retornará um K' sem significado. No mundo 0, o teste de A_B irá falhar com grande probabilidade [Goldwasser and Bellare 2001].

Prova da primeira equação da proposição 2.6: Dado qualquer adversário B com recursos restritos a t, q, ql é possível construir um adversário A_B, usando recursos $t', q+1, (q+1)l$, tal que

$$Adv^{kr}_{F,B} \leq Adv^{prf}_{F,A_B} + \frac{1}{2^L} \tag{2.11}$$

Então, observe que

$$\begin{aligned} Adv^{kr}_F(t,q,\mu) &= \max_B \left\{ Adv^{kr}_{F,B} \right\} \\ &\leq \max_B \left\{ Adv^{prf}_{F,A_B} + 2^{-L} \right\} \\ &\leq \max_A \left\{ Adv^{prf}_{F,A} + 2^{-L} \right\} \\ &\leq Adv^{prf}_F(t', q+1, (q+1)l) + 2^{-L}. \end{aligned}$$

O máximo de B ($\max_B$) é o mais inteligente de todos os adversários com recursos limitados a t, q e ql. Na segunda linha, foi aplicada a equação 2.11. A terceira linha é verdade porque o

conjunto inclui todos os adversários A_B. Na última linha, a equação 2.8 é encontrada por definição. Ainda é necessário provar a equação 2.11. O adversário A_B proverá um oráculo para a função $g : \{0,1\}^l \rightarrow \{0,1\}^L$, e, então, tentará determinar em qual mundo está, executando B como uma sub-rotina, como segue [Goldwasser and Bellare 2001]:

Adversário A_B^g

Faça $i \leftarrow 0$

Execute o adversário B, respondendo suas consultas ao oráculo como segue

Quando B fizer uma consulta x ao oráculo faça

$i \leftarrow i + 1; x_i \leftarrow x$

$y_i \leftarrow g(x_i)$

Retorne y_i para B como resposta do oráculo de B

Até que B pare e forneça uma chave K' como resultado

Faça z ser uma *string* de l bits que não pertença ao conjunto $\{x_1, ..., x_q\}$

$y \leftarrow g(z)$

Se $F(K', z) = y$ então retorne 1 senão retorne 0

O valor de z deve ser um valor diferente de qualquer um que B consultou e, por isso, $q < 2^l$. Os valores das probabilidades em cada mundo são:

$$P\left[Exp_{F,A_B}^{prf-1} = 1\right] \geq Adv_{F,B}^{kr} \tag{2.12}$$

$$P\left[Exp_{F,A_B}^{prf-0} = 1\right] = 2^{-L}. \tag{2.13}$$

Na equação 2.12 (mundo 1), o oráculo g é F_K, para algum K que B tenta adivinhar. Portanto, é o mesmo experimento Exp $_{F,B}^{kr}$ que foi visto no início desta seção. No entanto, é possível que A_B retorne 1 mesmo que B não tenha sucesso. Isso aconterá se $K' \neq K$ mas $F(K', x) = F(K, x)$, e este é o motivo da inequação e não uma igualdade. Na equação 2.13 (mundo 0), o oráculo g é uma função randômica e como x nunca foi consultado por B, $g(x)$ tem uma possibilidade de 2^{-L} de ser igual a $F(K', x)$ e assim A retornar 1. Com isso, conclui-se que:

$$\begin{aligned} Adv_{F,A_B}^{prf} &= P\left[Exp_{F,A_B}^{prf-1} = 1\right] - P\left[Exp_{F,A_B}^{prf-0} = 1\right] \\ &\geq Adv_{F,B}^{kr} - 2^{-L}. \end{aligned}$$

Reorganizando os termos, tem-se a prova da equação 2.11.

Prova da segunda equação da proposição 2.6: Para provar a segunda equação, busca-se a redução $B \rightarrow A_B$ com a propriedade

$$Adv_{F,B}^{kr} \leq Adv_{F,A_B}^{prp-cpa} + \frac{1}{2^L - q}. \tag{2.14}$$

Então, observe que

$$\begin{aligned} Adv_F^{kr}(t,q,\mu) &= \max_B \left\{ Adv_{F,B}^{kr} \right\} \\ &\leq \max_B \left\{ Adv_{F,A_B}^{prf} + \frac{1}{2^L - q} \right\} \\ &\leq \max_A \left\{ Adv_{F,A}^{prf} + \frac{1}{2^L - q} \right\} \\ &\leq Adv_F^{prf}(t', q+1, (q+1)l) + \frac{1}{2^L - q}. \end{aligned}$$

O máximo de B ($\max_B$) é o mais inteligente de todos os adversários com recursos limitados a t, q e ql. Na segunda linha, foi aplicada a Equação 2.14. A terceira linha é verdade porque o conjunto inclui todos os adversários A_B. Na última linha, a Equação 2.9 é encontrada por definição. Ainda é necessário provar a Equação 2.14. Os valores para as probabilidades são:

$$P\left[Exp_{F,A_B}^{prp-cpa-1} = 1\right] = Adv_{F,B}^{kr} \tag{2.15}$$

$$P\left[Exp_{F,A_B}^{prp-cpa-0} = 1\right] \leq \frac{1}{2^L - q}. \tag{2.16}$$

A Equação 2.16 (mundo 0) se justifica por se tratar de uma permutação e uma consulta é reservada para conclusão de A_B. Assim,

$$\begin{aligned} Adv_{F,A_B}^{prp-cpa} &= P\left[Exp_{F,A_B}^{prp-cpa-1} = 1\right] - P\left[Exp_{F,A_B}^{prp-cpa-0} = 1\right] \\ &\geq Adv_{F,B}^{kr} - \frac{1}{2^L - q}. \end{aligned}$$

Isso prova a Equação 2.14.

O próximo exemplo mostra que a vantagem-kr de uma família pode ser significativamente menor que sua vantagem-cpa-prp, significando que a família pode ser muito segura quanto à recuperação da chave, ainda que insegura como uma prf ou prp e portanto não utilizável no desenvolvimento de protocolos.

Exemplo 2. 7. *Dado o cifrador de blocos* $E : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ *onde* $E_K(x) = x$*, isto é, a cifragem de* x *sempre retorna* x *como texto cifrado. Todas as chaves K são de tamanho k-bits e todas as entradas* x *são de tamanho l-bits. Ele é muito seguro quanto à recuperação da chave mas muito inseguro como uma PRP sob CPA. Para todos os valores de* t *e* q*:*

$$Adv_E^{kr}(t, q, ql) = 2^{-k}.$$

Por outro lado

$$Adv_E^{prp-cpa}(t, 1, l) \geq 1 - 2^{-l}$$

para $t = O(l)$ *[Goldwasser and Bellare 2001].*

Em outras palavras, nesse cifrador de blocos é muito difícil descobrir a chave, mas é muito fácil distingui-lo de uma função randômica. O discernidor pode verificar se $g(0^l) = 0^l$ *e retornar 1 se for verdadeiro, e assim obter uma vantagem de* $1 - 2^{-l}$*.*

2.1.9 Ataque do Aniversário

Suponha que $E : \{0,1\}^K \times \{0,1\}^l \rightarrow \{0,1\}^l$ é uma família de permutações, tal como um cifrador de blocos. Se temos um oráculo $g : \{0,1\}^l \rightarrow \{0,1\}^l$ que é uma instância de E ou uma função randômica, existe um teste simples para determinar qual desses ele é. Consulte o oráculo sobre os pontos distintos $(x_1, x_2, ...x_q)$ e receba as respostas $(y_1, y_2, ..., y_q)$. Se g foi uma permutação, então os valores devem ser distintos. Se g foi uma função randômica, então podem ou não ser distintos. Surpreendentemente, é preciso fazer $q = \sqrt{2^l}$ consultas para conseguir uma significante vantagem. A razão é o paradoxo do aniversário. Isso quer dizer que uma instância de um cifrador de blocos pode ser discernida de uma função randômica baseada em um número aproximado de $2^{l/2}$ exemplos de entradas e saídas.

Proposição 2. 8. *Seja* $E : \{0,1\}^K \times \{0,1\}^l \rightarrow \{0,1\}^l$ *uma família de permutações. Suponha que* q *seja tal que,* $2 \leq q \leq 2^{(l+1)/2}$*. Então:*

$$Adv_E^{prf}(t, q, ql) \geq 0.3.\frac{q(q-1)}{2^l}$$

onde t *é o tempo de* q *cálculos de* E *mais* $O(ql)$*.*

Analisando a Proposição 2.8, para $q = \sqrt{2^l}$ a vantagem é de aproximadamente 0.3. Para $q = 1.5 \times \sqrt{2^l}$ a vantagem é de aproximadamente 0.7 e para $q = 1.8 \times \sqrt{2^l}$, a vantagem é de aproximadamente 0.97. O que é praticável para muitos cifradores de blocos, como por exemplo os que tem $l = 64$, para os quais seriam necessárias aproximadamente 14 bilhões de consultas para conseguir a vantagem de aproximadamente 0.97.

A prova da Proposição 2.8 é dada em [Goldwasser and Bellare 2001].

2.1.10 PRFs *versus* PRPs

Analisar cifradores de bloco é mais simples se forem considerados como PRFs, ainda que a maioria deles seja modelada como PRPs. Então, pode ser interessante relacionar as funções vantagem PRF e PRP. A Proposição 2.9 diz que uma família de permutações E pode ter vantagem-prf maior que vantagem-prp, mas por um valor de no máximo $q(q-1)/2^{l+1}$, que é exatamente o termo da probabilidade do ataque do aniversário. Considerando esse o melhor ataque possível em PRP e que não é possível em PRF.

Proposição 2. 9. *Suponha que* $E : \{0,1\}^K \times \{0,1\}^l \rightarrow \{0,1\}^l$ *é uma família de permutações. Então:*

$$Adv_E^{prf}(t,q,ql) \leq \frac{q(q-1)}{2^{l+1}} + Adv_E^{prp-cpa}(t,q,ql)$$

para quaisquer t *e* q*.*

A prova da Proposição 2.9 é dada em [Goldwasser and Bellare 2001].

2.1.11 Construção de Famílias de PRFs

Como construir famílias de PRFs não faz parte deste estudo, pois o interesse aqui é a análise dos protocolos que usam PRFs e PRPs já existentes, portanto esse tópico não foi apresentado aqui. Algumas sugestões de construção de famílias de PRFs podem ser encontradas em [Goldwasser and Bellare 2001] e [Goldreich et al. 1984].

2.2 A Função Vantagem dos Esquemas de Criptografia de Chave Secreta

O modelo de criptografia de chave secreta, também conhecido como modelo simétrico, considera dois participantes que compartilham uma chave e usam esta chave para conseguir a comunicação de dados com vários atributos de segurança. Os principais objetivos dessa segurança são a privacidade e a autenticidade dos dados transmitidos. Um esquema de criptografia simétrico (também conhecido como esquema de criptografia de chave secreta) permite que os participantes com posse de uma chave secreta alcancem o objetivo de privacidade dos dados.

2.2.1 Esquema de Criptografia Simétrico

Um esquema de criptografia simétrico consiste de três algoritmos: (1) um de cifragem, que diz ao emissor como processar seus dados em função da chave para produzir o objeto para transmitir; (2) um de decifragem que diz ao receptor como recuperar os dados originais a partir da transmissão e, quando possível, também fazer alguma verificação; (3) e um algoritmo de geração de chave, que produz uma chave que os participantes precisam compartilhar. Tal esquema é representado por `SE=(K,E,D)` [Goldwasser and Bellare 2001] onde:

- `K` é o algoritmo de geração da chave, ele é randômico e retorna um *string* K. A operação de retornar uma chave randômica é representada por $K \xleftarrow{R} \mathtt{K}$.
- `E` é o algoritmo de cifragem, este toma a chave $K \in Keys(\mathtt{SE})$ e um *plaintext* $M \in \{0,1\}^*$ para retornar um *ciphertext* $C \in \{0,1\}^* \cup \{\perp\}$. Essa operação é representada por: $C \leftarrow E_K(M)$.
- `D` é o algoritmo de decifragem determinístico que recebe a chave $K \in Keys(\mathtt{SE})$ e um *ciphertext* $C \in \{0,1\}^*$ para retornar $M \in \{0,1\}^* \cup \{\perp\}$, operação representada por: $M \leftarrow D_K(C)$.

É necessário que se $C = E_K(M)$, então $M = D_K(C)$. $\{0,1\}^*$ especifica um conjunto de *strings* binárias de tamanho não determinado aqui. Restrições capturadas retornam o símbolo especial $\perp$.

O tópico sobre como os participantes obtêm a chave sem que o adversário tenha acesso a ela não será abordado aqui, só é assumido que isto acontece de alguma forma segura.

Tipos de Esquemas de Criptografia Simétrico

O algoritmo de criptografia pode ser randomizado (*randomized*) ou dependente de um estado interno (*stateful*) [Goldwasser and Bellare 2001].

- Se é *randomized*, o algoritmo joga moedas e usa o resultado para calcular suas saídas para uma dada entrada de K e M. Cada vez que o algoritmo é invocado, ele joga moedas novamente. Para duas entradas com os mesmos valores de K e M, as saídas deverão ser diferentes.

- Se é *stateful*, seu funcionamento depende de uma variável global tal como um contador, o qual é iniciado de alguma forma pré-especificada. Quando o algoritmo é invocado, ele calcula um *ciphertext* baseado em K, M e o valor atual do contador. Ele então atualiza o contador e armazena-o.

No esquema *stateful*, $E_K(M)$ pode ou não retornar $\perp$ dependendo não somente de M mas também do valor da variável de estado, como por exemplo se o contador chegar ao seu valor máximo.

Quando não existe o contador ou variáveis globais, o esquema é independente de estado (*stateless*). No esquema *stateless*, existe um conjunto de *strings*, chamadas de *plaintext space*, tal que $E_K(M) \neq \perp$ para todos os K e todos os M no espaço do *plaintext*. Nas próximas seções, será mostrado que no esquema *stateless* a randomização é essencial para a segurança.

2.2.2 Alguns Esquemas de Criptografia

Esquema *One-time-pad*

Esquema 2. 10. *O* one-time-pad *encryption scheme (também chamado de Vernam cipher), da forma* `SE=(K,E,D)`*, é stateful e determinístico. O algoritmo* `K`*, gerador da chave, retorna uma* string *randômica K de tamanho k-bit. O tamanho da chave é um parâmetro do esquema, tal que o espaço da chave é $Keys(\texttt{SE}) = \{0,1\}^k$. O cifrador mantém um contador ctr, o qual é iniciado com zero [Goldwasser and Bellare 2001]).*

Seguem os algoritmos de cifragem $\mathtt{E}$ e decifragem $\mathtt{D}$:

Algoritmo $\mathtt{E}_K(M)$
- Faça $n = \mid M \mid$
- Se $ctr + n > k$ então responda $\perp$
- Para i de 1 até n faça
  - $C[i] \leftarrow K[ctr + i] \oplus M[i]$
- FimPara
- $ctr \leftarrow ctr + n$
- $C \leftarrow C[1]...C[n]$
- Retorne (ctr, C)

Algoritmo $\mathtt{D}_K(C, ctr)$
- Faça $n = \mid M \mid$
- Para i de 1 até n faça
  - $M[i] \leftarrow K[ctr + i] \oplus C[i]$
- FimPara
- $M \leftarrow M[1]...M[n]$
- Retorne M

Nos algoritmos acima, $C[i], M[i] e K[i]$ representam o i-ésimo bit da respectiva *string* de bits. Os bits da chave nunca são reutilizados, assim, se não há bits na chave suficientes para cifrar a mensagem, o algoritmo retorna $\perp$. Junto com o *ciphertext* é retornado o contador, para tornar possível a decifragem.

Os esquemas seguintes são na realidade uma família de permutações ou uma família de funções. É necessária uma função *padding* para ajustar o tamanho da mensagem de maneira apropriada para múltiplos do tamanho do bloco associado para a família [Goldwasser and Bellare 2001].

Modo ECB

Esquema 2. 11. *Seja* $E : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ *um cifrador de blocos operando no modo* Electronic Code Book *(ECB) como um esquema de criptografia simétrica stateless,* `SE = (K, E, D)`. *O algoritmo* `K` *de geração de chave simplesmente retorna uma chave randômica para o cifrador de bloco, tal que o espaço da chave seja* $\{0,1\}^k$ *[Goldwasser and Bellare 2001].*

Os algoritmos de cifragem $\mathtt{E}$ e decifragem $\mathtt{D}$ funcionam como segue:

Algoritmo $\mathtt{E}_K(M)$

Se $\mid M \mid < l$ então retorne $\perp$

Se $\mid M \mid \bmod l \neq 0$ então retorne $\perp$

Divida M em $M[1]...M[n]$

Para i de 1 até n faça

$C[i] \leftarrow E_K(M[i])$

FimPara

$C \leftarrow C[1]...C[n]$

Retorne C

Algoritmo $\mathtt{D}_K(C)$

Se $\mid C \mid < l$ então retorne $\perp$

Se $\mid C \mid \bmod l \neq 0$ então retorne $\perp$

Divida C em $C = C[1]...C[n]$

Para i de 1 até n faça

$M[i] \leftarrow E_K^{-1}(C[i])$

FimPara

$M \leftarrow M[1]...M[n]$

Retorne M

Divida M significa que M será dividido em blocos de tamanho l-bits e então $M[i]$ é o i-ésimo bloco.

A Figura 2.1 e a Figura 2.2 ilustram bem a cifragem e decifragem, respectivamente, no Esquema 2.11 [Stallings 1998].

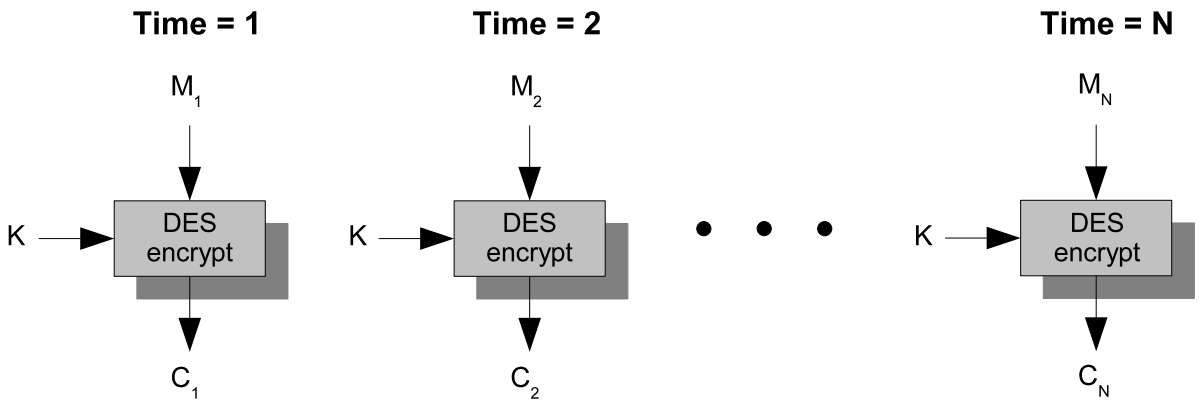

Figura 2.1: Cifragem no modo ECB (*Electronic Codebook*)

A característica mais importante do ECB é que, se o mesmo bloco aparecer mais de uma vez na mensagem, ele sempre produzirá o mesmo texto cifrado [Stallings 1998].

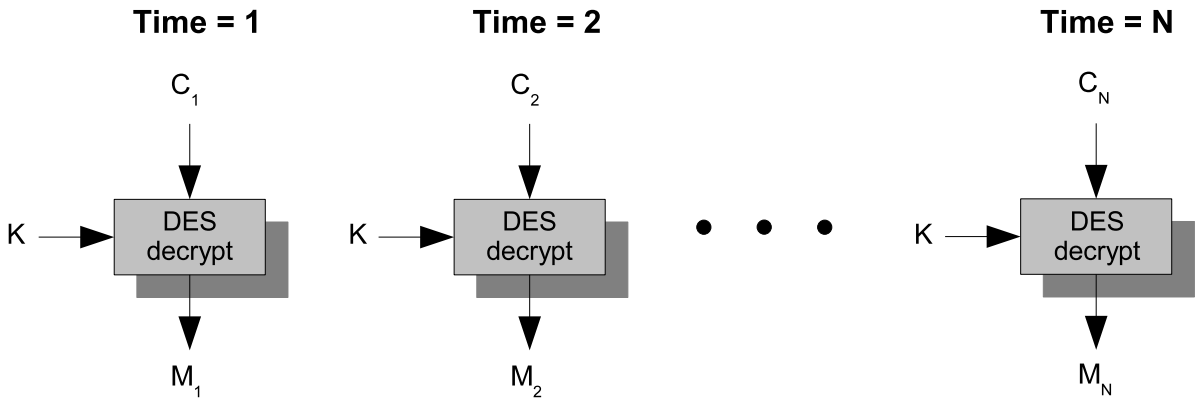

Figura 2.2: Decifragem no modo ECB (*Electronic Codebook*)

Modo CBC

Esquema 2. 12. *Seja* $E : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ *um cifrador de blocos operando no modo* Cipher-Block Chaining *(CBC) com um Vetor de Inicialização (IV) randômico. O esquema de criptografia é simétrico e stateless na forma* `SE=(K,E,D)`*. Como no anterior, o algoritmo* `K` *de geração de chave retorna uma chave randômica para o cifrador de bloco, tal que o espaço da chave é* $\{0,1\}^k$ *[Goldwasser and Bellare 2001].*

Veja os algoritmos de cifragem e decifragem a seguir:

Algoritmo $\mathtt{E}_K(M)$
- Se $\mid M \mid < l$ então retorne $\perp$
- Se $\mid M \mid \bmod l \neq 0$ então retorne $\perp$
- Divida M em $M[1]...M[n]$
- $C[0] \xleftarrow{R} \{0,1\}^i$
- Para i de 1 até n faça
  - $C[i] \leftarrow E_K(C[i-1] \oplus M[i])$
- FimPara
- $C \leftarrow C[0]C[1]...C[n]$
- Retorne C

Algoritmo $\mathtt{D}_K(C)$
- Se $\mid C \mid < 2l$ então retorne $\perp$
- Se $\mid C \mid \bmod l \neq 0$ então retorne $\perp$
- Divida C em $C = C[0]C[1]...C[n]$
- Para i de 1 até n faça
  - $M[i] \leftarrow E_K^{-1}(C[i]) \oplus C[i-1]$
- FimPara
- $M \leftarrow M[1]...M[n]$
- Retorne M

Divida M é como já foi explicado no esquema anterior. O IV ($C[0]$) é escolhido randomicamente toda vez que o algoritmo é executado. Isso torna muito difícil a ocorrência do mesmo texto cifrado para mensagens idênticas.

A Figura 2.3 e a Figura 2.4 ilustram bem a cifragem e decifragem, respectivamente, no Esquema 2.12 [Stallings 1998].

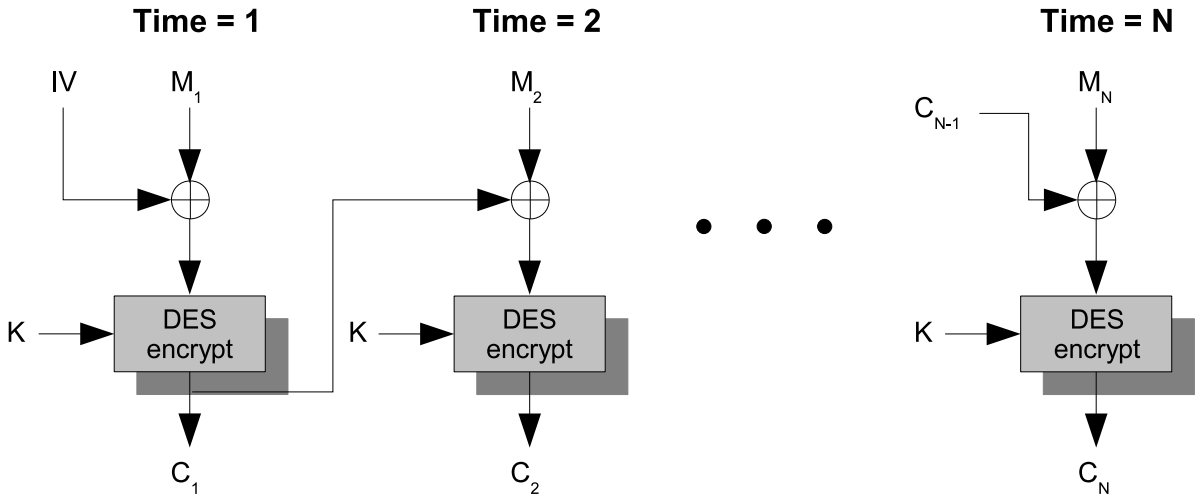

Figura 2.3: Cifragem no modo CBC (*Chipher Block Chaining*)

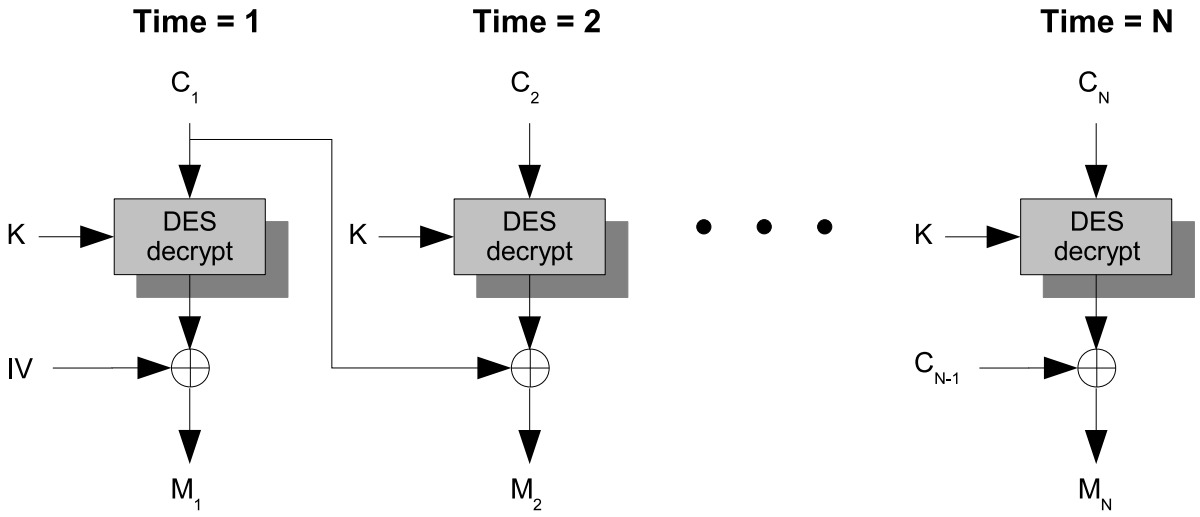

Figura 2.4: Decifragem no modo CBC (*Chipher Block Chaining*)

O inconveniente do modo CBC é que tanto a chave quanto o IV devem ser protegidos, para evitar que o texto a ser recebido possa ser alterado por um atacante [Stallings 1998]. Se alguém (um atacante) quiser alterar apenas um determinado bit da mensagem original, mudando por exemplo

uma ordem de crédito para débito, de posse do IV ele pode conseguir isso. Considere o seguinte:

$$C_1 = E_K(IV \oplus M_1)$$

$$M_1 = IV \oplus D_K(C_1)$$

O bit i da mensagem original é conseguido pelo destinatário da seguinte forma:

$$M_1[i] = IV[i] \oplus D_K(C_1)[i],$$

mudando um bit no IV o atacante muda apenas um bit da mensagem original, veja

$$M_1[i]' = IV[i]' \oplus D_K(C_1)[i].$$

Modos CTRs

Nos esquemas a seguir $NtS_l(i)$ (leia "*number to string*") denota a *string* de tamanho l-bits que é a representação binária do inteiro i. E $StN(s)$ (leia "*string to number*") denota o inteiro não negativo cuja representação é a *string* de bits s.

Apesar de os modos CTR (*counter*) não terem a mesma força do CBC, nós veremos que eles têm boas propriedades de segurança e, ao contrário do CBC, pode ser paralelizado. Existem duas variantes do modo, uma randômica e outra *stateful*, com propriedades de segurança diferentes.

Esquema 2. 13. *Seja* $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ *uma família de funções (não necessariamente uma família de permutações), operando no modo CTR. O ponto de início é escolhido como randômico, e isto é feito novamente a cada mensagem. Esse esquema é simétrico, stateless e é*

representado por `SE=(K,E,D)`*. Ele é também conhecido por modo R-CTR (R de randômico). O algoritmo* `K`*, de geração de chave, retorna uma chave randômica e o espaço da chave é* $\{0,1\}^k$ *[Goldwasser and Bellare 2001].*

Os algoritmos de cifragem `E` e de decifragem `D` funcionam como segue:

Algoritmo $\mathtt{E}_K(M)$	Algoritmo $\mathtt{D}_K(C)$
Se $\mid M \mid < L$ então retorne $\perp$	Se $\mid C \mid < l + L$ então retorne $\perp$
Se $\mid M \mid \bmod L \neq 0$ então retorne $\perp$	Se $(\mid C \mid - l) \bmod L \neq 0$ então retorne $\perp$
Divida M em $M[1]...M[n]$	Faça $C[0]$ ser os primeiros l bits de C
$R \xleftarrow{R} \{0,1,...,2^l - 1\}$	Divida o resto de C em $C[1]...C[n]$
Para i de 1 até n faça	$R \leftarrow StN(C[0])$
$C[i] \leftarrow F_K(NtS_l(R+i)) \oplus M[i]$	Para i de 1 até n faça
FimPara	$M[i] \leftarrow F_K(NtS_l(R+i)) \oplus C[i])$
$C[0] \leftarrow NtS_l(R)$	FimPara
$C \leftarrow C[0]C[1]...C[n]$	$M \leftarrow M[1]...M[n]$
Retorne C	Retorne M

Aqui, *Divida* M significa que M será dividido em blocos de tamanho L-bits (e não l-bits como feito antes), sendo $M[i]$ o i-ésimo bloco. O valor randômico R é incluído no texto cifrado para permitir a decifragem. O algoritmo de decifragem então, retira os primeiros l-bits do C e divide o restante em blocos de L-bits.

Esquema 2. 14. *Seja* $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ *uma família de funções (não necessariamente uma família de permutações), operando no modo CTR com um contador. Esse esquema é simétrico e stateful e é representado por* `SE=(K,E,D)`*. Ele é conhecido também como modo C-CTR (C de contador). O algoritmo* `K`*, de geração de chave, retorna uma chave randômica e o espaço da chave é* $\{0,1\}^k$ *[Goldwasser and Bellare 2001].*

Os algoritmos de cifragem `E` e de decifragem `D` funcionam como segue:

Algoritmo $\mathtt{E}_K(M)$

Se $\mid M \mid < L$ então retorne $\perp$

Se $\mid M \mid \bmod L \neq 0$ então retorne $\perp$

Divida M em $M[1]...M[n]$

Se $ctr + n \geq 2^l$ então retorne $\perp$

Para i de 1 até n faça

$C[i] \leftarrow F_K(NtS_l(ctr + i)) \oplus M[i]$

FimPara

$C[0] \leftarrow NtS_l(ctr)$

$C \leftarrow C[0]C[1]...C[n]$

$ctr \leftarrow ctr + n$

Retorne C

Algoritmo $\mathtt{D}_K(C)$

Se $\mid C \mid < l + L$ então retorne $\perp$

Se $(\mid C \mid - l) \bmod L \neq 0$ então retorne $\perp$

Faça $C[0]$ ser os primeiros l bits de C

Divida o resto de C em $C[1]...C[n]$

$ctr \leftarrow StN(C[0])$

Para i de 1 até n faça

$M[i] \leftarrow F_K(NtS_l(ctr + i)) \oplus C[i])$

FimPara

$M \leftarrow M[1]...M[n]$

Retorne M

Divida M é como explicado no esquema anterior. Ao contador não é permitido zerar ou reusar. O contador é incluído no *ciphertext* para possibilitar a decriptação. O algoritmo de cifragem atualiza o contador a cada invocação, e inicia com o valor atualizado a cada nova invocação. Para reiniciar o contador é necessário gerar outra chave.

A Figura 2.5 e a Figura 2.6 ilustram bem a cifragem e decifragem, respectivamente, nos esquemas 2.13 e 2.14.

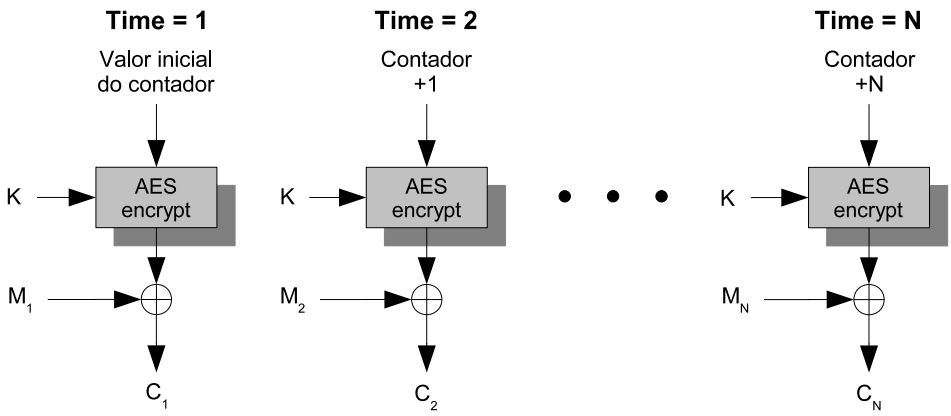

Figura 2.5: Cifragem no modo CTR (*Counter*)

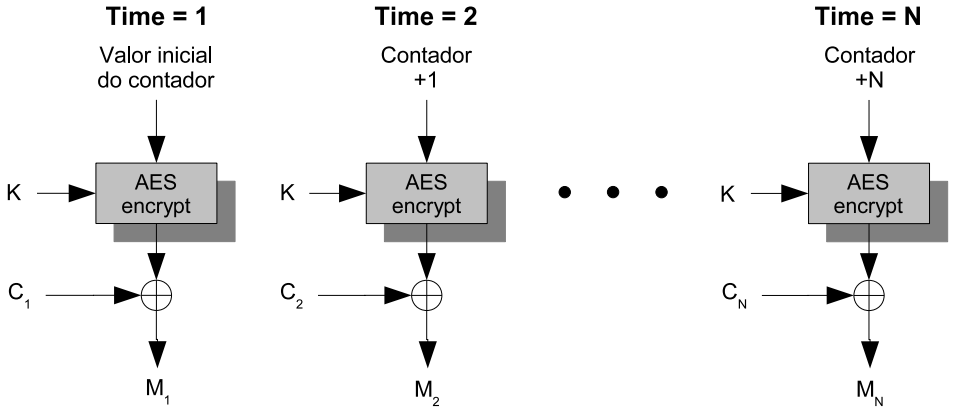

Figura 2.6: Decriptação no modo CTR (*Counter*)

2.2.3 Pensando em Segurança

É muito mais fácil pensar em insegurança do que em segurança, porque é possível identificar ações que, sem dúvidas, implicam que o esquema é inseguro [Goldwasser and Bellare 2001].

Por exemplo, se o adversário pode, a partir de poucos textos cifrados, derivar a chave K, ele pode depois decifrar qualquer coisa que vê, então se o esquema permite uma fácil recuperação da chave a partir de poucos textos cifrados ele é considerado inseguro. Por outro lado, uma ausência de facilidade de recuperar a chave certamente não é certeza de que o esquema é seguro [Goldwasser and Bellare 2001].

No esquema `SE=(K,E,D)` que é assumido aqui, dois participantes compartilham uma chave $K \xleftarrow{R} K$. O adversário não conhece K. Será permitido que o adversário capture qualquer texto cifrado que fluir pelo canal entre os dois participantes. Ele pode colecionar os textos cifrados e tentar obter alguma coisa deles. Alguém pode dizer que, dado um texto cifrado C, o adversário não tem qualquer idéia do que é a mensagem original M. Isso entretanto não é verdade, em decorrência daquilo que é chamado "informação a priori". Freqüentemente, alguma coisa sobre a mensagem é sabida. Por exemplo, ela pode ser um pacote com cabeçalho conhecido, ou ela pode ser um texto em inglês. Alguém também pode tentar dizer que segurança é: dado um texto cifrado C, o adversário não consegue recuperar a mensagem original M facilmente. Mas atualmente, isso não é suficiente. A razão é que o adversário pode ter a habilidade de obter uma informação parcial de M. Por exemplo, se o adversário puder recuperar o primeiro bit de M ou a soma de todos os bits de M, isso pode ser uma informação valiosa [Goldwasser and Bellare 2001].

Para um exemplo concreto, digamos que eu estou transmitindo para o meu corretor uma men-

sagem com a seqüência que é uma decisão de "Compre" ou "Venda" para uma pré-especificada sequência de "ações". Nós combinamos que um bit 1 significa "venda" e um bit 0 significa "compre". A mensagem é enviada cifrada. Mas, se o primeiro bit escapa, o adversário sabe se eu quero vender ou comprar ações. É verdade que, se os dados estão em diferentes formatos isso não acontece. Entretanto, assumir requisitos a respeito do formato dos dados dos usuários, ou como eles usaram os esquemas é perigoso para desenvolvimento de protocolos de segurança. Em outras palavras os desenvolvedores de protocolos de segurança não podem fazer presunções sobre o conteúdo ou formato dos dados [Goldwasser and Bellare 2001].

No Esquema 2.11 modo ECB, por exemplo, dado um texto cifrado, um adversário pode recuperar a mensagem? Se F é um bom cifrador de blocos e o adversário não sabe a chave, então ele levará um longo tempo invertendo F_k. Não obstante esse não é um bom esquema. Considere só o caso $n = 1$ de um simples bloco de mensagem, suponha que eu tenha só duas mensagens, 0^l para comprar e 1^l para vender, enviando sempre esses dois dados, mas sempre um desses dois. O adversário pode ver que os dois primeiros blocos são iguais e o terceiro também [Goldwasser and Bellare 2001].

O objetivo da segurança deve ser aproximar-se o máximo possível do perfeito. Nenhuma informação parcial deve escapar. Em um esquema seguro, não deve ser possível co-relacionar textos cifrados de mensagens diferentes, isso pode significar que informações estão escapando. Isso significa que a criptografia deve ser probabilística ou dependente da informação de estado. Cada cifragem deve usar moedas novas, ou um contador, para que uma mesma mensagem seja diferente a cada vez. Isso significa que `E` deve ser um algoritmo probabilístico ou *stateful*. Por isso os esquemas de criptografia simétrica são definidos a partir de tais algoritmos [Goldwasser and Bellare 2001].

2.2.4 Segurança na Teoria da Informação

Como foi dito na seção 2.2.3, o objetivo da segurança deve ser aproximar-se o máximo possível do perfeito. Nenhuma informação parcial deve escapar. Alguém no entanto, poderia pensar em segurança perfeita como: o adversário não ter qualquer idéia do que a mensagem é. Para abordar a noção de segurança da Teoria da Informação, chamada de segurança perfeita, veja o esquema *one-time-pad*, ele possui tal propriedade [Goldwasser and Bellare 2001].

Seja um esquema de criptografia simétrico `SE=(K,E,D)`, onde dois participantes compartilham uma chave K, sendo que o adversário não conhece K. Ao adversário é permitido capturar qualquer texto cifrado que flui no canal entre as partes. Tendo capturado um texto cifrado, ele tenta obter

alguma informação sobre a mensagem original correspondente [Goldwasser and Bellare 2001].

Considerando agora o esquema *one-time-pad* que se encaixa na descrição acima e assumindo que uma mensagem de tamanho k-bits é cifrada e transmitida, onde k é o tamanho da chave, devido a escolha aleatória da chave, isso é visto como muito seguro. Mas não é possível realmente dizer que o adversário não tem qualquer idéia do que a mensagem é, dado que ele tem o texto cifrado. O adversário sempre pode tentar adivinhar a mensagem, ou ter uma idéia do que é a mensagem, como por exemplo, ele pode saber que alguns bytes do cabeçalho indicam o "IP" do transmissor [Goldwasser and Bellare 2001].

Assim, adotaremos uma medida de segurança comparativa. O que mais o adversário sabe sobre a mensagem, dado o texto cifrado, que ele não sabia antes de ver o texto cifrado. A segurança perfeita espera que o texto cifrado não adicione qualquer novidade sobre a mensagem que já não era sabida antes [Goldwasser and Bellare 2001].

Exemplo 2. 15. *Assuma que uma simples mensagem será cifrada e que estamos interessados na segurança desta cifragem. Existe um espaço de mensagens que o transmissor irá cifrar, chamado plaintext. Por exemplo, com o esquema* one-time-pad*, se o tamanho da chave é k bits então plaintext $= \{0,1\}^k$. Nós modelamos a informação a priori (a informação que o adversário possui sobre a mensagem) como a distribuição da probabilidade no conjunto de possíveis mensagens. Formalmente: $D : Plaintext \rightarrow [0,1]$ tal que*

$$\sum_{M \in Plaintexts} D(M) = 1,$$

e também $D(M) > 0$ para todos $M \in plaintexts$, por exemplo, podem ser quatro mensagens, $00, 01, 10, 11$, com $D(00) = 1/6, D(01) = 1/3, D(10) = 1/4, eD(11) = 1/4$. O transmissor irá escolher 00 com uma probabilidade de 1/6, e assim por diante. E essas probabilidades são de conhecimento do adversário [Goldwasser and Bellare 2001].

Como foi dito, o esquema de criptografia é perfeitamente seguro se a posse do texto cifrado não acrescenta qualquer informação adicional sobre a mensagem que já era conhecida a priori.

Após o transmissor ter escolhido a mensagem de acordo com D, a chave K é também escolhida e a mensagem é cifrada para gerar um texto cifrado, via $C \leftarrow \mathtt{E}_K(M)$. O adversário recebe C. Se o adversário, que agora conhece C, não consegue nada além de dizer a probabilidade que M tem de ser escolhida como $D(M)$, isto significa que a posse do texto cifrado não adiciona qualquer nova

informação ao que já era sabido, assim, a segurança é perfeita. De modo mais formal:

$$S = Keys(\mathtt{SE}) \times Plaintexts \times \{0,1\}^r,$$

onde S denota o espaço do nosso experimento, r é o número de moedas que o algoritmo de cifragem joga. (Se zero, o algoritmo de cifragem é determinístico, como é o caso do *one-time-pad*). Podemos introduzir as seguintes variáveis randômicas [Goldwasser and Bellare 2001]:

$$\begin{array}{ll} \mathbf{K} : S \to Keys(\mathtt{SE}) & \text{defined by } (K, M, R) \mapsto K \\ \mathbf{M} : S \to Plaintexts & \text{defined by } (K, M, R) \mapsto M \\ \mathbf{C} : S \to \{0,1\}^* & \text{defined by } (K, M, R) \mapsto \mathtt{E}_K(M; R), \end{array}$$

assim **K** simplesmente retorna o valor da chave escolhida, enquanto **M** retorna o valor da mensagem escolhida. A última variável randômica retorna a cifragem da mensagem usando a chave K e as moedas R. A distribuição probabilista nesse espaço de exemplo é denotado $\mathbf{P}_{D,\mathtt{SE}}[.]$ e é dado pela escolha de K por $\mathtt{K}$, a escolha de M como por D e uma escolha randômica de R, todas sendo feitas de forma independente.

Definição 2. 16. *Seja* `SE = (K,E,D)` *um esquema de criptografia simétrica com um espaço de mensagens associado Plaintexts. Seja* $D : Plaintexts \to [0,1]$ *uma distribuição de mensagens no Plaintexts. Nós dizemos que* `SE` *é perfeitamente seguro com referência a distribuição* D*, se para cada* $M \in Plaintexts$ *e cada texto cifrado* C *possível, é o caso em que [Goldwasser and Bellare 2001]:*

$$\boldsymbol{P}_{D,\mathtt{SE}}[\boldsymbol{M} = M \mid \boldsymbol{C} = C] = D(M). \tag{2.17}$$

Aqui "$\mathbf{M} = M$" é o caso da mensagem escolhida pelo emissor ter sido M, e "$\mathbf{C} = C$" é o caso onde o texto cifrado calculado pelo emissor e recebido pelo adversário ter sido C. A definição considera a probabilidade condicional de que a mensagem foi M dado que o texto cifrado foi C. Ela diz que essa probabilidade é exatamente a probabilidade anterior da mensagem M, chamada de $D(M)$.

Exemplo 2. 17. *Seja* `SE=(K, E, D)` *o esquema criptográfico* one-time-pad *com o tamanho da chave (e também tamanho da mensagem e tamanho do ciphertext) fixado em* $k = 2$ *bits e o espaço da mensagem fixado em* $Plaintexts = \{0,1\}^k$*. Seja* D *a distribuição das mensagens no Plaintexts definida por* $D(00) = 1/6$*,* $D(01) = 1/3$*,* $D(10) = 1/4$ *e* $D(11) = 1/4$*. Para cada*

texto cifrado $C \in \{0,1\}^k$ possível, a Tabela 2.1 mostra o valor da $\boldsymbol{P}_{D,SE}[\boldsymbol{C} = C \mid \boldsymbol{M} = M]$, isto é, a probabilidade de obter um texto cifrado particular se você cifrar M com o esquema one-time-pad *[Goldwasser and Bellare 2001].*

A Tabela 2.1 indica essa probabilidade é sempre 0.25, porque tendo fixado M, os textos cifrados possíveis são $M \oplus K$ com limites de K sobre $\{0,1\}^k$. Assim todas as *strings* C de k bits tem igual possibilidade de ser o texto cifrado.

$D(M)$	M \ C	00	01	10	11
1/6	00	0.25	0.25	0.25	0.25
1/3	01	0.25	0.25	0.25	0.25
1/4	10	0.25	0.25	0.25	0.25
1/4	11	0.25	0.25	0.25	0.25

Tabela 2.1: Probabilidades de se obter C se a mensagem cifrada for M.

Os valores da Tabela 2.1 podem ser provados pelo Lema 2.18.

Lema 2. 18. *Seja $k \geq 1$ um inteiro e `SE=(K,E,D)` o esquema one-time-pad com o tamanho da chave k bits e o espaço da mensagem fixado em $Plaintexts = \{0,1\}^k$. Seja D uma distribuição das mensagens no* Plaintexts*, então*

$$\boldsymbol{P}_{D,SE}[\boldsymbol{C} = Y \mid \boldsymbol{M} = X] = 2^{-k}$$

para qualquer $X \in Plaintexts$ e qualquer $Y \in \{0,1\}^k$ [Goldwasser and Bellare 2001].

Prova do Lema 2.18: No esquema *one-time-pad* $Y = K \oplus X$ se somente se $K = Y \oplus X$ e a probabilidade de K ser uma *string* em particular é 2^{-k}, porque K é escolhido de forma randômica.

A Tabela 2.2 mostra o valor de $\mathbf{P}_{D,\mathrm{SE}}[\mathbf{M} = M \mid \mathbf{C} = C]$, isso é, a probabilidade de a mensagem escolhida ter sido M dado que o adversário viu o ciphertext C. Note que é sempre igual a probabilidade anterior $D(M)$, o que mostra que o esquema *one-time-pad* alcança a noção de segurança perfeita.

Os valores da Tabela 2.2 podem ser provados pelo Teorema 2.19.

Teorema 2. 19. *Seja $k \geq 1$ um inteiro, `SE=(K,E,D)` o esquema* one-time-pad *com uma chave de k bits e o espaço da mensagem fixado em $Plaintexts = \{0,1\}^k$. Para uma distribuição das mensagens D no Plaintexts, `SE` é perfeitamente seguro com relação a D [Goldwasser and Bellare 2001].*

$D(M)$	M \ C	00	01	10	11
1/6	00	1/6	1/6	1/6	1/6
1/3	01	1/3	1/3	1/3	1/3
1/4	10	1/4	1/4	1/4	1/4
1/4	11	1/4	1/4	1/4	1/4

Tabela 2.2: Probabilidades de a mensagem ter sido M dado que o adversário viu C.

Prova do Teorema 2.19: Para uma mensagem $M \in Plaintexts$ e C um possível texto cifrado, $C \in \{0,1\}^k$. A prova está em mostrar que a Equação 2.17 é verdadeira. Então

$$\begin{aligned}\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = M \mid \mathbf{C} = C\right] &= \mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{C} = C \mid \mathbf{M} = M\right].\frac{\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = M\right]}{\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{C} = C\right]} \\ &= 2^{-k}.\frac{\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = M\right]}{\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{C} = C\right]}\end{aligned}$$

A primeira equação foi obtida pela regra de Bayes. A segunda foi obtida aplicando o Lema 2.18 com $X = M$ e $Y = C$. Já por definição

$$\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = M\right] = D(M)$$

isso é, a probabilidade *a priori* de M. Para o último termo

$$\begin{aligned}\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{C} = C\right] &= \sum_x \mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = X\right].\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{C} = C \mid \mathbf{M} = X\right] \\ &= \sum_x D(X).2^{-k} \\ &= 2^{-k}.\sum_x D(X) \\ &= 2^{-k}.1\end{aligned}$$

A soma aqui foi sobre todas as possíveis mensagens $X \in Plaintexts$, e nós usamos o Lema 2.18. Assim

$$\mathbf{P}_{D,\mathtt{SE}}\left[\mathbf{M} = M \mid \mathbf{C} = C\right] = 2^{-k}.\frac{D(M)}{2^{-k}} = D(M)$$

como queríamos demonstrar.

O esquema *one-time-pad* não é somente dotado de segurança perfeita, mas é também considerado o mais simples e natural de todos [Goldwasser and Bellare 2001].

2.2.5 Indiscernibilidade Sob *Chosen-Plaintext Attack* (IND-CPA)

Somente esquemas com chaves tão longas quanto a mensagem podem prover segurança perfeita, apesar de isso não ter sido provado ainda. Como tais esquemas são impraticáveis, o objetivo é encontrar uma noção de segurança que, apesar de não ser perfeita, seja efetivamente boa. Na prática, podemos levar em conta o fato de que o adversário é computacionalmente restrito. Pode existir informação "utilizável" no texto cifrado, mas se você não pode computá-la, o texto cifrado na verdade não lhe dá qualquer informação sobre a mensagem [Goldwasser and Bellare 2001].

Podemos pensar que os esquemas criptográficos modernos são quebráveis matematicamente, a princípio. Mas computacionalmente são resistentes, já que não existe poder de processamento suficiente para quebrá-los. A definição de IND-CPA formaliza essas idéias.

Definição

O mais forte argumento de segurança pode ser ilustrado por um cenário onde um transmissor cifra uma de duas mensagens conhecidas, de mesmo tamanho, e em seguida envia-a (as duas mensagens são de conhecimento do adversário). Nesse cenário, o adversário deve, de posse do texto cifrado, dizer qual das duas mensagens foi cifrada e corresponde a tal texto cifrado. É possível mostrar que, se o adversário leva um tempo muito longo para dizer qual foi a mensagem, então, o esquema é seguro [Goldwasser and Bellare 2001].

Aqui, a idéia acima será extendida para considerar a cifragem não de uma mensagem, mas de uma seqüência delas. Então, existe uma seqüência de pares de mensagens $(M_{1,0}, M_{1,1})$, ..., $(M_{q,0}, M_{q,1})$, em que, cada par de duas mensagens tem o mesmo tamanho. A seqüência é conhecida do adversário. Um bit b é, então, escolhido como randômico e uma seqüência de textos cifrados $C_1, ..., C_q$ é produzida, em que $C_i \leftarrow \mathrm{E}_K(M_{i,b})$. O esquema é randomizado ou *stateful*. O adversário recebe uma seqüência de textos cifrados e deve adivinhar o bit b para ganhar. Em outras palavras, o adversário está tentando determinar se o transmissor enviou $M_{1,0}, ..., M_{q,0}$ ou $M_{1,1}, ..., M_{q,1}$. Para dar mais força ao adversário, ele pode escolher a seqüência de pares de mensagens via um *Chosen-Ciphertext Attack* (CPA). Isso é, ele escolhe o primeiro par e recebe C_1, escolhe o segundo, e assim

por diante [Goldwasser and Bellare 2001].

Mais formalmente, é considerado um esquema de criptografia especifico `SE=(K,E,D)` (ele pode ser *stateless* ou *stateful*). Nós consideramos o adversário A um programa que tem acesso a um oráculo para o qual ele fornece um par (M_0, M_1) como entrada. O oráculo irá retornar um texto cifrado. Serão consideradas duas formas possíveis para computar o texto cifrado, e essas duas formas correspondem a dois mundos possíveis [Goldwasser and Bellare 2001]:

Mundo 0: O oráculo entregue ao adversário é $\mathtt{E}_K(LR(.,.,0))$. Então sempre que o adversário faz uma consulta (M_0, M_1) para o oráculo, ele computa $C \xleftarrow{R} \mathtt{E}_K(M_0)$, e retorna C como resposta.

Mundo 1: O oráculo entregue ao adversário é $\mathtt{E}_K(LR(.,.,1))$. Então sempre que o adversário faz uma consulta (M_0, M_1) para o oráculo, ele computa $C \xleftarrow{R} \mathtt{E}_K(M_1)$, e retorna C como resposta.

O oráculo cifrador *left*-ou-*right* $\mathtt{E}_K(LR(.,.,b))$ funciona como indicado:

Oráculo $\mathtt{E}_K(LR(M_0, M_1, b))$ // $b \in \{0,1\}$ e $M_0, M_1 \in \{0,1\}^*$
$C \leftarrow \mathtt{E}_K(M_b)$
Responda C

O oráculo cifra uma das mensagens, a escolha é feita de acordo com o bit b. O primeiro mundo pode ser chamado também de esquerdo (*left*) e o segundo de direito (*right*). O problema do adversário é, depois de conversar com o oráculo por algum tempo, dizer qual dos dois oráculos lhe foi dado. Ao adversário somente é permitido fazer consultas ao oráculo com mensagens M_0 e M_1 de mesmo tamanho. Tais consultas são consideradas legítimas. E o adversário será legítimo se fizer somente consultas legítimas. Um esquema de criptografia é considerado "seguro quanto a um *chosen-plaintext attack*" se um adversário não consegue obter uma vantagem significativa em discernir os casos $b = 0$ e $b = 1$. E essa noção é conhecida como Indiscernibilidade Sob *Chosen-Plaintext Attack* (IND-CPA) [Goldwasser and Bellare 2001].

Definição 2. 20. *Seja* `SE=(K,E,D)` *um esquema de criptografia simétrico, seja* $b \in \{0,1\}$*, e seja* A *um algoritmo que tem acesso a um oráculo que, dado um par de* strings *como entrada, retorna uma* string. *Considere o seguinte experimento [Goldwasser and Bellare 2001]:*

Experimento $\boldsymbol{Exp}^{ind-cpa-b}_{SE,A}$

$K \xleftarrow{R} K$

$d \leftarrow A^{\mathtt{E}_K(LR(.,.,b))}$

Responda d

A vantagem-ind-cpa de A é definida como

$$\boldsymbol{Adv}_{SE,A}^{ind-cpa} = \boldsymbol{P}\left[\boldsymbol{Exp}_{SE,A}^{ind-cpa-1} = 1\right] - \boldsymbol{P}\left[\boldsymbol{Exp}_{SE,A}^{ind-cpa-0} = 1\right]$$

Para qualquer t, q, μ nós definimos a vantagem-ind-cpa de SE via

$$\boldsymbol{Adv}_{SE}^{ind-cpa}(t, q, \mu) =_{A}^{max} \left\{\boldsymbol{Adv}_{\boldsymbol{SE},A}^{ind-cpa}\right\}$$

onde o máximo é sobre todos os A legítimos com complexidade de tempo de t, fazendo para o oráculo no máximo q consultas e com a soma dos tamanhos das consultas no máximo de μ bits.

Como já foi discutido, um adversário é legítimo se todas as consultas que ele faz ao oráculo têm o mesmo tamanho de $\mid M_0 \mid$. A medida da complexidade de tempo é a mesma usada nas seções anteriores. Se $\mathbf{Adv}_{\mathrm{SE},A}^{ind-cpa}$ é pequeno, isso significa que A está fazendo um bom trabalho, ele está respondendo 1 com mais freqüência no primeiro experimento do que no segundo. A função vantagem-ind-cpa do esquema mede a probabilidade máxima que a segurança do esquema SE tem de ser comprometida por um adversário usando os recursos indicados [Goldwasser and Bellare 2001].

Interpretação Alternativa da Vantagem

Uma resposta possível para o problema da seção anterior é o adversário retornar um bit randômico. Nesse caso, ele tem uma probabilidade de 1/2 de acertar. Uma visão alternativa, então, é medir o excesso sobre 1/2. Considere o experimento [Goldwasser and Bellare 2001]:

Experimento $\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-cg}$

Pegue um bit b de forma randômica

Faça $K \xleftarrow{R} \mathrm{K}$

$g \leftarrow A^{\mathrm{E}_K(LR(.,.,b))}$

Se $b = g$ responda 1 senão responda 0.

Aqui A executa com um oráculo para o mundo b. O experimento retorna 1 se A acertou o mundo (ind-cpa-cg - *correctly guess*). Assim

$$\mathbf{P}\left[\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-cg} = 1\right]$$

é a probabilidade que tem A de acertar em qual mundo está. A proposição a seguir diz que metade da vantagem é exatamente o excesso sobre a metade da chance que A tem de adivinhar em qual mundo está.

Proposição 2. 21. *Para um esquema de criptografia simétrico* $\mathtt{SE}$

$$\boldsymbol{P}\left[\boldsymbol{Exp}_{\mathtt{SE},A}^{ind-cpa-cg}=1\right]=\frac{1}{2}+\frac{\boldsymbol{Adv}_{\mathtt{SE},A}^{ind-cpa}}{2}$$

para qualquer adversário-ind-cpa A [Goldwasser and Bellare 2001].

Prova da Proposição 2.21: Seja $\mathbf{P}[\cdot]$ a probabilidade do evento "$\cdot$" no experimento $\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-cg}$, a prova é como abaixo.

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-cg}=1\right] &= \mathbf{P}[b=g] \\
&= \mathbf{P}[b=g \mid b=1]\cdot\mathbf{P}[b=1]+\mathbf{P}[b=g \mid b=0]\cdot\mathbf{P}[b=0] \\
&= \mathbf{P}[b=g \mid b=1]\cdot\frac{1}{2}+\mathbf{P}[b=g \mid b=0]\cdot\frac{1}{2} \\
&= \mathbf{P}[g=1 \mid b=1]\cdot\frac{1}{2}+\mathbf{P}[g=0 \mid b=0]\cdot\frac{1}{2} \\
&= \mathbf{P}[g=1 \mid b=1]\cdot\frac{1}{2}+(1-\mathbf{P}[g=1 \mid b=0])\cdot\frac{1}{2} \\
&= \frac{1}{2}+\frac{1}{2}\cdot(\mathbf{P}[g=1 \mid b=1]-\mathbf{P}[g=1 \mid b=0]) \\
&= \frac{1}{2}+\frac{1}{2}\cdot\left(\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-1}=1\right]-\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-0}=1\right]\right) \\
&= \frac{1}{2}+\frac{1}{2}\cdot\mathbf{Adv}_{\mathtt{SE},A}^{ind-cpa}
\end{aligned}$$

Na terceira linha aparece o 1/2 porque a escolha de b é randômica. O restante das linhas são manipulações de probabilidades e simplificações. Na sétima linha, a condição da probabilidade é substituída pela probabilidade do experimento relacionado. E na última linha obtêm-se o que se queria demonstrar.

2.2.6 Exemplos de Ataques CPA (*Chosen-Plaintext Attack*)

Aqui são apresentadas aplicações do conceito de IND-CPA. Os exemplos propõem ataques, primeiro ao modo ECB e depois a um esquema determinístico e *stateless*, generalizando assim os resultados.

Ataque CPA no modo ECB

Considere um cifrador de blocos $E : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ e o Esquema 2.11 de criptografia simétrica ECB na forma $\mathtt{SE} = (\mathtt{K}, \mathtt{E}, \mathtt{D})$. Suponha que um adversário vê o texto cifrado $C = \mathtt{E}_K(M)$ correspondente a alguma desconhecida mensagem M, cifrada com a chave K também desconhecida do adversário. Se E é o AES, por exemplo, é difícil para o adversário recuperar M. Mas ECB tem outras fraquezas. Se duas mensagens M e M' tem o mesmo primeiro bloco, o adversário saberá disso, sendo um vazamento de parte da informação da mensagem, o que não é permitido num esquema criptográfico seguro [Goldwasser and Bellare 2001].

Proposição 2. 22. *Dado o esquema acima, vamos mostrar que existe um adversário que tem uma vantagem-ind-cpa alta usando poucos recursos. Isso é,*

$$\boldsymbol{Adv}_{SE}^{ind-cpa}(t, 1, 2l) = 1$$

para $t = O(l)$ mais o tempo de duas execuções do cifrador de blocos. A vantagem do adversário é 1 com apenas uma consulta [Goldwasser and Bellare 2001].

Prova da Proposição 2.22: O adversário A recebe um oráculo $\mathtt{E}_K(LR(.,.,b))$ que aceita um par de mensagens como entrada e retorna a cifragem de uma das duas entradas, a esquerda ou a direita, dependendo do valor de b. O objetivo de A é determinar o valor de b. O adversário trabalha como segue:

Adversário $A^{\mathtt{E}_K(LR(.,.,b))}$
$M_1 \leftarrow 0^{2l}; M_0 \leftarrow 0^l \parallel 1^l$
$C[1]C[2] \leftarrow \mathtt{E}_K(LR(M_0, M_1, b))$
Se $C[1] = C[2]$ então responda 1 senão responda 0

As mensagens M_0 e M_1 são do tamanho de dois blocos cada e são computadas de acordo com o

modo ECB. Assim,

$$\begin{aligned}\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-1}=1\right] &= 1\\ \mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-0}=1\right] &= 0.\end{aligned}$$

As duas equações são verdadeiras porque no mundo 1, o oráculo retorna $C[1]C[2] = E_K(0^l) \parallel E_K(0^l)$, então $C[1] = C[2]$ e A retorna 1. No mundo 0, o oráculo retorna $C[1]C[2] = E_K(0^l) \parallel E_K(1^l)$. Como E_K é uma permutação, então sempre $C[1] \neq C[2]$ e A retorna 0.

A vantagem é calculada como segue

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cpa} = 1 - 0 = 1.$$

Como pode ser observado, A consegue uma vantagem alta fazendo apenas uma consulta de tamanho $2l$ bits. Então

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t, 1, 2l) = 1.$$

Esquemas Determinísticos e *Stateless* São Inseguros

O modo ECB é determinístico e *stateless*, isto quer dizer que se uma mesma mensagem é cifrada duas vezes, o mesmo texto cifrado será gerado. Por isso é preciso analisar essas propriedades de uma maneira mais generalista.

Proposição 2. 23. *Se um esquema determinístico e* stateless *na forma* `SE = (K,E,D)`*, então:*

$$\boldsymbol{Adv}_{SE,A}^{ind-cpa}(t, 2, 2m) = 1,$$

para $t = O(l)$ *mais o tempo de duas cifragens [Goldwasser and Bellare 2001].*

Essa proposição pode ser aplicada em qualquer esquema determinístico e *stateless* ao contrário da proposição 2.22 que é aplicável especificamente ao modo ECB.

Prova da Proposição 2.23: A seguir é apresentado um adversário A que tem vantagem 1, isto é:

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cpa} = 1.$$

Adversário $A^{\mathtt{E}_K(LR(.,.,b))}$

Faça X e Y *strings* de m bits distintas no espaço do texto plano

$C[1] \leftarrow \mathtt{E}_K(X, Y, b))$

$C[2] \leftarrow \mathtt{E}_K(Y, Y, b))$

Se $C[1] = C[2]$ então responda 1 senão responda 0

Para esse adversário

$$\begin{aligned} \mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-1} = 1\right] &= 1 \\ \mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-0} = 1\right] &= 0, \end{aligned}$$

assim

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cpa} = 1 - 0 = 1$$

e

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t, 2, 2m) = 1.$$

O adversário A consegue essa vantagem fazendo apenas duas consultas ao oráculo, cada uma de tamanho m. Isso mostra que esquemas criptográficos não podem ser determinísticos e *stateless*.

Para demonstrar a validade dos experimentos, vamos usar o adversário anterior contra os modos CBC e CTR.

Ataque CPA no modo CBC

Já para o Esquema 2.12 (CBC) a criptografia é $C[i] \leftarrow F_K(C[i-1] \oplus M[i])$, onde $C[0]$ é randômico, então:

No **mundo 1** $M_{1,0} = 0^l$ é cifrado para $C[1] = F_K(C[0] \oplus 0^l)$

e $M_{1,1} = 0^l$ é cifrado para $C[2] = F_K(C[1] \oplus 0^l)$, onde

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-1} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[F_K(C[0] \oplus 0^l) = F_K(C[1] \oplus 0^l)\right] \\
&= \mathbf{P}\left[C[0] \oplus 0^l = C[1] \oplus 0^l\right] \\
&= \mathbf{P}\left[C[0] = C[1]\right] \\
&= 2^{-k}.
\end{aligned}$$

No **mundo 0** $M_{0,0} = 0^l$ é cifrado para $C[1] = F_K(C[0] \oplus 0^l)$
e $M_{0,1} = 1^l$ é cifrado para $C[2] = F_K(C[1] \oplus 1^l)$, onde

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-0} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[F_K(C[0] \oplus 0^l) = F_K(C[1] \oplus 1^l)\right] \\
&= \mathbf{P}\left[C[0] \oplus 0^l = C[1] \oplus 1^l\right] \\
&= \mathbf{P}\left[C[0] = \overline{C[1]}\right] \\
&= 2^{-k}.
\end{aligned}$$

Assim

$$\mathbf{Adv}_{\mathrm{SE},A}^{ind-cpa} = 2^{-k} - 2^{-k} = 0.$$

Esse adversário não leva vantagem alguma no esquema CBC.

Ataque CPA no modo C-CTR

No Esquema 2.14 (C-CTR), a criptografia é $C[i] \leftarrow F_K(NtS_l(ctr + i)) \oplus M[i]$, onde ctr é inicialmente zero.

No **mundo 1** $M_{1,0} = 0^l$ é cifrado para $C[1] = F_K(NtS_l(1)) \oplus 0^l$

e $M_{1,1} = 0^l$ é cifrado para $C[2] = F_K(NtS_l(2)) \oplus 0^l$, onde

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-1} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[F_K(NtS_l(1)) = F_K(NtS_l(2))\right] \\
&= \mathbf{P}\left[NtS_l(1) = NtS_l(2)\right] \\
&= 0.
\end{aligned}$$

No **mundo 0**, $M_{1,0} = 0^l$ é cifrado para $C[1] = F_K(NtS_l(1)) \oplus 0^l$

e $M_{1,1} = 1^l$ é cifrado para $C[2] = F_K(NtS_l(2)) \oplus 1^l$, onde

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathrm{SE},A}^{ind-cpa-0} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[F_K(NtS_l(1)) = \overline{F_K(NtS_l(2))}\right] \\
&= 2^{-k}.
\end{aligned}$$

Assim

$$\mathbf{Adv}_{\mathrm{SE},A}^{ind-cpa} = 0 - 2^{-k} = 2^{-k}.$$

Esse adversário leva vantagem muito pequena no esquema C-CTR.

Uma característica importante que esse ataque deve enfatizar, é ECB ser um esquema de criptografia inseguro, mesmo que o cifrador de blocos por trás dele seja muito seguro. A fraqueza não está na ferramenta, mas na maneira de utilizá-la.

2.2.7 Segurança Quanto à Recuperação da Mensagem

Já foi discutido nas seções anteriores que a dificuldade quanto a recuperar a chave ou a mensagem original são propriedades necessárias, mas não suficientes para segurança. É possível provar que a definição de IND-CPA implica nessas propriedades. Isto é, se um esquema é seguro quanto à definição de IND-CPA, então, ele também é seguro quanto à recuperação da chave ou recuperação da mensagem [Goldwasser and Bellare 2001].

Para formalizar isso, será mostrado que se existe um adversário B capaz de recuperar a men-

sagem de um dado texto cifrado, então, isso nos permite construir um adversário A que quebra o esquema no sentido IND-CPA, dizendo em qual dos dois mundos ele está. Mas se o esquema é seguro no IND-CPA, o adversário B não pode existir [Goldwasser and Bellare 2001].

Nesse exemplo, a tarefa do adversário B será decifrar um texto cifrado que é formado pela cifragem de uma mensagem de tamanho m, escolhida randomicamente. Ao adversário é dada a habilidade de ver pares de "mensagem" *versus* "texto cifrado", que ele capturará através de acessos a um oráculo de cifragem. O oráculo recebe uma mensagem M, como entrada, e retorna um texto cifrado $C \xleftarrow{R} \mathtt{E}_K(M)$. Para prover, ao adversário, um texto cifrado como desafio, será introduzido um outro oráculo que não recebe entrada alguma, apenas gera uma *string* m-bits randômica como $M \in plaintexts$, calcula $C \xleftarrow{R} \mathtt{E}_K(M)$ e retorna C. O adversário B faz apenas uma consulta a este último oráculo. Ele ganha se puder retornar a mensagem M correspondente ao texto cifrado C retornado pelo oráculo do desafio. O oráculo do desafio é chamado de $\mathtt{E}_K(\$^m)$, onde $\$^m$ indica que a *string* m-bits é escolhida de forma randômica para ser cifrada.

Experimento $\mathbf{Exp}^{pr}_{\mathtt{SE},B}$

$K \xleftarrow{R} \mathtt{K}$

$M \leftarrow B^{\mathtt{E}_K(\cdot),\mathtt{E}_K(\$^m)}$

Se $D_K(C) = M$, onde C é a resposta para a consulta de B para $\mathtt{E}_K(\$^m)$

então responda 1

senão responda 0

A vantagem-pr (*plaintext recovery*) de B é definida como

$$\mathbf{Adv}^{pr}_{\mathtt{SE},B} = \mathbf{P}\left[\mathbf{Exp}^{pr}_{\mathtt{SE},B} = 1\right]$$

Para qualquer t, q, μ nós definimos a vantagem-pr de $\mathtt{SE}$ via

$$\mathbf{Adv}^{pr}_{\mathtt{SE}}(t, q, \mu) =^{max}_{B} \left\{\mathbf{Adv}^{pr}_{\mathbf{SE},B}\right\}$$

onde o máximo de todos os Bs que tem complexidade de tempo t, fazendo até q consultas ao oráculo e que a soma das consultas sejam no máximo μ bits [Goldwasser and Bellare 2001].

A proposição abaixo diz que a probabilidade de um adversário ter sucesso em recuperar a mensagem de um texto cifrado não pode exceder o vantagem-ind-cpa do esquema, usando os mesmos recursos, mais a chance de simplesmente adivinhar a mensagem. Em outras palavras, segurança no IND-CPA implica em segurança quanto à recuperação da mensagem.

Proposição 2. 24. *Seja* `SE=(K,E,D)` *um esquema criptográfico simétrico e stateless cujo espaço da mensagem inclui* $\{0,1\}^m$*, então*

$$\boldsymbol{Adv}_{SE}^{pr}(t,q,\mu) \leq \boldsymbol{Adv}_{SE}^{ind-cpa}(t,q+1,\mu+m) + \frac{1}{2^m}$$

para qualquer t, q, μ *[Goldwasser and Bellare 2001].*

Na prova, é usada mais uma vez a redução. O adversário A será construído de forma a utilizar o adversário B como sub-rotina, mostrando que, se existe tal adversário B, é muito fácil construir o adversário A.

Prova da Proposição 2.24: Dado qualquer adversário B com recursos restritos a t, q, μ, é possível construir um adversário A_B, com recursos $t, q+1, \mu+m$, tal que

$$\mathbf{Adv}_{\mathtt{SE},\mathtt{B}}^{pr} \leq \mathbf{Adv}_{\mathtt{SE},A_B}^{ind-cpa} + \frac{1}{2^m} \tag{2.18}$$

O adversário A_B recebe um oráculo cifragem-lr e tenta determinar em qual mundo está. Para isso ele executa o adversário B como uma sub-rotina.

Adversário $A_B^{\mathtt{E}_K(LR(.,.,b))}$

Execute o adversário B e responda as suas consultas ao oráculo como segue

Quando B fizer uma consulta X ao oráculo de cifragem, faça

$Y \leftarrow \mathtt{E}_K(LR(X,X,b))$

Retorne Y para B como resposta

Quando B fizer uma consulta ao oráculo do desafio, faça

$M_0 \xleftarrow{R} \{0,1\}^m; M_1 \xleftarrow{R} \{0,1\}^m$

$C \leftarrow \mathtt{E}_K(LR(M_0,M_1,b))$

Retorne C para B como resposta

Até B parar e devolver uma mensagem decriptada M

Se M = M_1 então responda 1 senão responda 0.

O adversário A_B executa B e responde as consultas de B com respostas do seu próprio oráculo. Quando B faz uma consulta X ao oráculo, A_B retorna $\mathtt{E}_K(LR(X,X,b))$. Quando B faz uma consulta ao oráculo do desafio, A_B retorna $\mathtt{E}_K(LR(M_0,M_1,b))$, onde M_0 e M_1 são escolhidos randomicamente e independente um do outro, e isso é feito apenas uma vez. No final, o adversário A_B testa se $M = M_1$ e se verdadeiro retorna 1.

Assim,

$$\mathbf{P}\left[\mathbf{Exp}^{ind-cpa-1}_{\mathtt{SE},A_B} = 1\right] \geq \mathbf{Adv}^{pr}_{\mathtt{SE},B} \tag{2.19}$$

$$\mathbf{P}\left[\mathbf{Exp}^{ind-cpa-0}_{\mathtt{SE},A_B} = 1\right] \leq 2^{-m} \tag{2.20}$$

e então

$$\begin{aligned}\mathbf{Adv}^{ind-cpa}_{\mathtt{SE},A_B} &= \mathbf{P}\left[\mathbf{Exp}^{ind-cpa-1}_{\mathtt{SE},A_B} = 1\right] - \mathbf{P}\left[\mathbf{Exp}^{ind-cpa-0}_{\mathtt{SE},A_B} = 1\right] \\ &\geq \mathbf{Adv}^{pr}_{\mathtt{SE},B} - 2^{-m}\end{aligned}$$

Rearranjando os termos o resultado é a equação 2.18. As equações 2.19 e 2.20 devem ainda ser justificadas. O adversário B irá retornar o $M = D_K(C)$ com uma probabilidade de pelo menos $\mathbf{Adv}^{pr}_{\mathtt{SE},B}$. No **mundo 1**, texto cifrado C é uma cifragem de M_1, assim isto significa que $M = M_1$ com uma probabilidade de pelo menos $\mathbf{Adv}^{pr}_{\mathtt{SE},B}$. No **mundo 0**, A_B irá retornar 1 somente se B retornar $M = M_1$. Mas B recebeu M_0 cifrado e $M_0 = M_1$ com uma probabilidade de no máximo 2^{-m}, devido as escolhas randômicas de M_0 e M_1.

2.2.8 Segurança do CTR sob CPA (*Chosen-Plaintext Attack*)

O esquema de criptografia simétrico CTR tem duas variantes: a randomizada (*stateless*) e a baseada em um contador (*stateful*). Ambos são seguros com relação a CPA, mas a versão do contador é mais segura. Isso é formalizado nos teoremas abaixo.

Teorema 2. 25. *Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e* `SE=(K,E,D)` *o esquema de criptografia simétrico C-CTR. Então para qualquer t, q, μ com $\mu < L2^l$ nós temos*

$$\boldsymbol{Adv}^{ind-cpa}_{SE}(t,q,\mu) \leq 2.\boldsymbol{Adv}^{prf}_{F}(t,q',lq'),$$

onde $q' = \mu/L$ [Goldwasser and Bellare 2001].

Teorema 2. 26. *Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e* `SE=(K,E,D)` *o esquema de criptografia simétrico R-CTR. Então para qualquer t, q, μ com $\mu < L2^l$ nós temos*

$$\boldsymbol{Adv}^{ind-cpa}_{SE}(t,q,\mu) \leq 2.\boldsymbol{Adv}^{prf}_{F}(t,q',lq') + \frac{\mu(q-1)}{L2^l},$$

onde $q' = \mu/L$ [Goldwasser and Bellare 2001].

Não há falhas aparentes no CTR. Mas como convencer alguém de que elas não existem? Não é possível esgotar todos os possíveis ataques. Os teoremas acima dizem que o CRT não tem falhas de projeto, dizem, que se você usar um bom cifrador de blocos, pode estar seguro de que ninguém irá quebra seu esquema criptográfico. É possível ter a convicção de que todos os ataques falham, mesmo que não se saiba exatamente como os ataques funcionam. Isso mostra o poder da função vantagem [Goldwasser and Bellare 2001].

Exemplo 2. 27. *Vamos supor que F é AES. Então o tamanho da chave é $k = 128$ e o tamanho do bloco é $l = L = 128$. Suponha que eu queira cifrar $q = 2^{40}$ mensagens, cada uma com $128 * 2^3$ bits de comprimento, então vou cifrar um total de $\mu = 2^{50}$ bits de dados. É seguro usar o modo contador do CTR? Quais as chances que um adversário tem de descobrir algo sobre os dados? Bem, se o adversário tem $t = 2^{60}$ ciclos de computador, então as chances não são mais que $\mathbf{Adv}_{SE}^{inc-cpa}(t, q, \mu)$ [Goldwasser and Bellare 2001].*

Segundo o Teorema 2.25, a chance é de no máximo $2.\mathbf{Adv}_F^{prf}(t, q', 128q')$, onde q' é como dado no teorema: $q' = \mu/L = 2^{50}/128 = 2^{43}$. Então a questão é qual é o valor de $\mathbf{Adv}_F^{prf}(t, q', 128q')$ com esses valores de t e q'? O teorema reduz a questão de estimar a probabilidade de perda de privacidade do esquema de criptografia para a questão de estimar o comportamento randômico do AES. Como visto na seção 2.1.6, alguém poderia conjecturar que

$$\mathbf{Adv}_{AES}^{prf}(t, q', 128q') = c_1.\frac{t/T_{AES}}{2^{128}} + \frac{(q')^2}{2^{128}},$$

onde T_{AES} é o tempo de fazer uma computação AES. Agora coloque em $t = 2^{60}$ e $q' = 2^{43}$ e se tem

$$\begin{aligned}
\mathbf{Adv}_{\mathrm{SE}}^{ind-cpa}(t, q, \mu) &\leq 2.\mathbf{Adv}_{AES}^{prf}(t, q', 128q') \\
&\leq 2c_1.\frac{t/T_{AES}}{2^{128}} + \frac{2(q')^2}{2^{128}} \\
&\leq \frac{2^{61}}{2^{128}}.\frac{c_1}{T_{AES}} + \frac{2^{43*2+1}}{2^{128}} \\
&\leq \frac{1}{2^{67}}.\frac{c_1}{T_{AES}} + \frac{1}{2^{41}} \\
&\leq \frac{1}{2^{41}}
\end{aligned}$$

No último passo (muito razoavelmente), é assumido que c_1/T_{AES} é no máximo 2^{26}. Assim

a chance de o adversário conseguir alguma informação sobre os dados cifrados aproximadamente 2^{-41}, mesmo que seja permitido para o adversário computar um tempo de 2^{60} e cifrar 2^{50} bits de dados. Essa chance é muito pequena e, certamente, possível conviver com ela. Nesse sentido que é possível dizer que o esquema é seguro [Goldwasser and Bellare 2001].

Observe que a cifragem e decifragem no Esquema 2.13 (C-CTR) não necessita de um acesso direto para a chave K, mas somente o acesso a uma sub-rotina, ou oráculo, que implementa a função F_K. É possível então substituir a função F_K pela família de funções G e estudar os casos onde $G = F$ e onde $G = Rand^{l \to L}$. Substituindo F por $Rand^{l \to L}$, é criado uma versão idealizada. O Lema 2.28 afirma que a chance de um adversário A quebrar o esquema idealizado é zero.

Lema 2. 28. *Seja A qualquer adversário-ind-cpa atacando o esquema $SE\left[Rand^{l \to L}\right]$ idealizado descrito acima. Então,*

$$\boldsymbol{Adv}^{ind-cpa}_{SE[Rand^{l \to L}],A} = 0$$

desde que o tamanho total das consultas de A ao oráculo seja no máximo $L2^l$. Nenhuma restrição é feita a respeito da complexidade de tempo de A [Goldwasser and Bellare 2001].

Prova do Lema 2.28: É muito fácil notar que, se g é uma função verdadeiramente randômica baseada em um contador (que nunca é reusado) e os dados sofrem uma operação XOR com o resultado randômico dela, então, o esquema funciona como um esquema *one-time-pad* e o adversário não obtém qualquer informação sobre os dados cifrados.

Considere um adversário A que tem acesso a um oráculo de cifragem-lr (como o da seção 2.2.5) e faz q consultas ao oráculo. Seja $(M_{i,0}, M_{i,1})$ a i-ésima consulta e n_i o número de blocos de $M_{i,0}$. Então, $M_{i,c}[j]$ é o valor do i-ésimo bloco de L bits de $M_{i,c}$ para $c \in \{0, 1\}$. C_i é a resposta do oráculo para a consulta $(M_{i,0}, M_{i,1})$. Então,

$$C_i[j] = g(NtS_l(n_1 + \ldots + n_{n-1} + j)) \oplus \begin{cases} M_{i,1}[j], & \text{se estiver no mundo } 1; \\ M_{i,0}[j], & \text{se estiver no mundo } 0. \end{cases}$$

Os valores aplicados a g são todos distintos e, assim, as saídas de g são totalmente randômicas e independentes. As probabilidades de o adversário retornar 1, em qualquer um dos mundos, são as mesmas, isto é,

$$\mathbf{P}\left[\mathbf{Exp}^{ind-cpa-1}_{\mathtt{SE}[Rand^{l\to L}],A} = 1\right] \quad = \mathbf{P}\left[\mathbf{Exp}^{ind-cpa-0}_{\mathtt{SE}[Rand^{l\to L}],A} = 1\right],$$

e então, a vantagem-ind-cpa de A é zero.

O Lema 2.28 trata de um esquema C-CTR idealizado, mas é necessário lembrar que outros esquemas, como o ECB, são inseguros mesmo trocando o cifrador por uma função verdadeiramente randômica. O Lema diz que o esquema é muito seguro quando g é uma função randômica. Mas para o caso em que g é uma família de funções F, é preciso provar o Teorema 2.25.

Prova do Teorema 2.25: Seja A um adversário-ind-cpa que ataca o esquema C-CTR (`SE = K, E, D`). O adversário A faz q consultas ao oráculo totalizando μ bits e tem uma complexidade de tempo t. Será apresentado um discernidor D_A tal que

$$\mathbf{Adv}^{ind-cpa}_{\mathtt{SE},A} \leq 2.\mathbf{Adv}^{prf}_{F,D_A}.$$

O discernidor usará A como uma sub-rotina e simulará um oráculo de cifragem-lr para A. Isso é, D_A irá responder as consultas que A fizer.

Discernidor D^g_A
- $b \xleftarrow{R} \{0,1\}$
- Execute o adversário A, respondendo suas consultas ao oráculo como segue
- Quando A fizer uma consulta ao oráculo (M_0, M_1) faça
  - $C \xleftarrow{R} \mathtt{E}_g(M_b)$
  - Responda C para A
- Até A parar e reportar um bit d como saída
- Se $d = b$ então retorne 1 senão retorne 0

Seguindo o modelo ind-cpa-cg (*correct guess*) apresentado na seção 2.2.5, o discernidor D_A retorna 1 quando $b = d$, o que significa que A identificou corretamente o mundo b no qual foi colocado. Já que a chance de D_A vencer é a mesma de A discernir entre $\mathtt{SE}[F]$ e $\mathtt{SE}[Rand^{l\to L}]$, então

$$\mathbf{P}\left[\mathbf{Exp}^{prf-1}_{F,D_A} = 1\right] = \mathbf{P}\left[\mathbf{Exp}^{ind-cpa-cg}_{\mathtt{SE}[F],A} = 1\right]$$

$$\mathbf{P}\left[\mathbf{Exp}^{prf-0}_{F,D_A} = 1\right] = \mathbf{P}\left[\mathbf{Exp}^{ind-cpa-cg}_{\mathtt{SE}[Rand^{l\to L}],A} = 1\right]$$

Aplicando a Proposição 2.21 é possível obter as Equações 2.21 e 2.22.

$$\mathbf{P}\left[\mathbf{Exp}^{prf-1}_{F,D_A} = 1\right] = \frac{1}{2} + \frac{1}{2} \cdot \mathbf{Adv}^{ind-cpa}_{\mathtt{SE}[F],A} \tag{2.21}$$

$$\mathbf{P}\left[\mathbf{Exp}^{prf-0}_{F,D_A} = 1\right] = \frac{1}{2} + \frac{1}{2} \cdot \mathbf{Adv}^{ind-cpa}_{\mathtt{SE}[Rand^{l\to L}],A} \tag{2.22}$$

Já por definição:

$$\begin{aligned}
\mathbf{Adv}^{prf}_{F,D_A} &= \mathbf{P}\left[\mathbf{Exp}^{prf-1}_{F,D_A} = 1\right] - \mathbf{P}\left[\mathbf{Exp}^{prf-0}_{F,D_A} = 1\right] \\
&= \frac{1}{2} \cdot \mathbf{Adv}^{ind-cpa}_{\mathtt{SE}[F],A} - \frac{1}{2} \cdot \mathbf{Adv}^{ind-cpa}_{\mathtt{SE}[Rand^{l\to L}],A} \\
&= \frac{1}{2} \cdot \mathbf{Adv}^{ind-cpa}_{\mathtt{SE}[F],A}
\end{aligned}$$

A última linha é obtida pela aplicação do Lema 2.28. Dessa equação é fácil obter:

$$\mathbf{Adv}^{ind-cpa}_{\mathtt{SE},A} \leq 2.\mathbf{Adv}^{prf}_{F,D_A}.$$

E isso é suficiente para provar o Teorema 2.25.

A prova do Teorema 2.26 não será apresentada aqui porque o esquema R-CTR não é utilizado em redes locais sem fios, ela pode ser encontrada em [Goldwasser and Bellare 2001].

2.2.9 Segurança do CBC sob CPA (*Chosen-Plaintext Attack*)

O CBC é o modo de cifragem mais popular, sendo mais complexo de analisar que o CTR. A prova pode ser encontrada em [Bellare et al. 1997].

Teorema 2. 29. *Suponha que* $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ *é uma família de funções e* `SE=(K,E,D)` *é o esquema CBC. Então para qualquer* t, q, μ *nós temos*

$$\boldsymbol{Adv}_{SE}^{ind-cpa}(t,q,\mu) \leq 2.\boldsymbol{Adv}_{F}^{prf}(t,q',lq') + \frac{2\mu^2}{l^2 2^l},$$

onde $q' = \mu/L$ *[Goldwasser and Bellare 2001].*

Note que se todas as mensagens são de n blocos então $\mu = nql$ assim o termo aditivo acima é $O(n^2q^2/2^l)$ [Goldwasser and Bellare 2001].

2.2.10 Indiscernibilidade sob CCA (*Chosen-Ciphertext Attack*)

O CCA é mais poderoso que CPA. O CCA permite ao adversário acessar um oráculo de decriptação. Ele pode alimentar esse oráculo com um texto cifrado e pegar de volta o *plaintext* correspondente. Imagine a situação em que um adversário em algum lugar consiga acesso temporário para o equipamento que executa a decriptação. Ele não pode, entretanto, extrair diretamente a chave do equipamento. Nós podemos pensar, de início, que o adversário que tem acesso ao oráculo de decriptação pode decriptar qualquer coisa. Para aproximar da realidade, vamos impor uma restrição ao acesso a tal oráculo. O adversário não pode enviar ao oráculo um texto cifrado que ele recebeu, anteriormente. Vamos considerar dois mundos [Goldwasser and Bellare 2001]:

Mundo 0: O adversário recebe o oráculo $\mathtt{E}_K(LR(.,.,0))$ e o oráculo $\mathtt{D}_K(.)$.

Mundo 1: O adversário recebe o oráculo $\mathtt{E}_K(LR(.,.,1))$ e o oráculo $\mathtt{D}_K(.)$.

O objetivo do adversário é o mesmo, identificar em qual dos dois mundos ele foi colocado. Existe um modo fácil para isso. Consultar o oráculo de cifragem-lr com duas mensagens distintas de igual tamanho M_0, M_1, para pegar de volta o texto cifrado C, e agora chamar o oráculo de decriptação com C. O resultado comparado com as mensagens M_0, M_1 enviadas anteriormente, nós diz em qual mundo está. Mas a restrição impõe que esta chamada para o oráculo de decriptação não é permitida. Se C foi retornado do oráculo de cifragem-lr, uma consulta com C ao oráculo de decriptação é ilegítima, e o experimento retorna $\perp$. Apesar dessa restrição o adversário ainda tem muito poder, podendo pegar o texto cifrado C retornado pelo oráculo de cifragem-lr, modificá-lo, criando assim o texto cifrado C', e consultar o oráculo de decriptação com C' [Goldwasser and Bellare 2001].

O nosso modelo pode parecer um pouco artificial. Afinal, como podemos impedir que o adversário que tem acesso ao oráculo de decriptação, o utilize de forma ilegítima? A restrição pode ser

realizada se o adversário tem acesso ao equipamento de decriptação por um período de tempo limitado. Após esse período, ele vê alguns textos cifrados e tenta quebrar a segurança baseando-se nos acessos anteriores. Quando um esquema de criptografia é usado em protocolos de troca autenticada de chaves, o adversário efetivamente tem a habilidade de montar *chosen-ciphertext attacks* como discutimos [Goldwasser and Bellare 2001].

Definição 2. 30. *: Seja* $\mathtt{SE}=(\mathtt{K},\mathtt{E},\mathtt{D})$ *um esquema simétrico de criptografia,* $b \in \{0,1\}$ *e* A *um algoritmo que tem acesso a dois oráculos e retorna um bit. Nós consideramos a seguinte experimento:*

Experimento $\boldsymbol{Exp}^{ind-cca-b}_{SE,A}$

$K \xleftarrow{R} \mathtt{K}$

$d \leftarrow A^{\mathtt{E}_K(LR(.,.,b)),\mathtt{D}_K(.)}$

Se A *consulta* $\mathtt{D}_K(.)$ *com um texto cifrado anteriormente retornado por* $\mathtt{E}_K(LR(.,.,b))$

Então responda $\perp$

Senão responda d.

A vantagem-ind-cpa de A é definida como

$$\mathbf{Adv}^{ind-cca}_{\mathtt{SE},A} = \mathbf{P}\left[\mathbf{Exp}^{ind-cca-1}_{\mathtt{SE},A} = 1\right] - \mathbf{P}\left[\mathbf{Exp}^{ind-cca-0}_{\mathtt{SE},A} = 1\right].$$

Para qualquer $t, q_e, \mu_e, q_d, \mu_d$ nós definimos a vantagem-pr de $\mathtt{SE}$ via

$$\mathbf{Adv}^{ind-cca}_{\mathtt{SE}}(t, q_e, \mu_e, q_d, \mu_d) =^{max}_{A} \left\{\mathbf{Adv}^{ind-cca}_{\mathbf{SE},A}\right\}$$

onde o máximo de todos os As que têm complexidade de tempo t, fazendo até q_e consultas ao oráculo cifragem-lr e a soma dessas consultas sejam no máximo μ_e bits, e fazendo até q_d consultas ao oráculo de decriptação e a soma dessas consultas sejam no máximo μ_d bits. Nós consideramos um esquema como "seguro sobre *chosen-ciphertext attack*" se um adversário "razoável" não pode obter uma vantagem "significante" em distinguir os casos $b = 0$ e $b = 1$ dado o acesso aos oráculos. Essa noção técnica é chamada de indiscernibilidade sob *chosen-ciphertext attack*, e é denotado por IND-CCA [Goldwasser and Bellare 2001].

2.2.11 Exemplos de CCA (*Chosen-Ciphertext Attack*)

Os *chosen-ciphertext attacks* são poderosos o suficiente para quebrar todos os modos padrões de operação, mesmo o CTR e o CBC. O esquema *one-time-pad* também é vulnerável ao CCA. Segurança perfeita só é conseguida quanto ao CPA [Goldwasser and Bellare 2001].

CCA no CTR

Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e $\mathtt{SE} = (K, E, D)$ o esquema de criptografia simétrico R-CTR associado, como já foi descrito. Seja $C[0]C[1]$ um texto cifrado de alguma mensagem M de L-bits e nós invertemos o bit i de $C[1]$, resultando em um novo texto cifrado $C[0]C'[1]$. Seja M' a mensagem obtida pela decriptação do novo texto cifrado. Então M' é igual a M com o i-ésimo bit invertido. Essa idéia pode ser usada para quebrar o sistema, dizendo em qual mundo um adversário foi colocado [Goldwasser and Bellare 2001].

Proposição 2. 31. *Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e $\mathtt{SE} = (K, E, D)$ o esquema de criptografia simétrico R-CTR associado, como já foi descrito. Então*

$$\boldsymbol{Adv}_{SE}^{ind-cca}(t, 1, L, 1, l + L) = 1$$

para $t = O(l + L)$ mais o tempo de uma aplicação de F. A vantagem do adversário é 1 mesmo usando pouco os recursos: somente uma consulta para cada oráculo. Isso mostra que o esquema é inseguro [Goldwasser and Bellare 2001].

Prova da Proposição 2.31: Vamos apresentar um algoritmo adversário A, com complexidade de tempo t, fazendo 1 consulta para o oráculo de cifragem-lr, a consulta tem tamanho L, fazendo também 1 consulta ao oráculo de decriptação, esta última com tamanho $l + L$, e tendo

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cca} = 1.$$

O objetivo de A é determinar o valor de b.

Adversário $A^{\mathtt{E}_K(LR(.,.,b)),\mathtt{D}_K(.)}$
$M_0 \leftarrow 0^L; M_1 \leftarrow 1^L$
$C[0]C[1] \leftarrow \mathtt{E}_K(LR(M_0, M_1, b))$
$C'[1] \leftarrow C[1] \oplus 1^L; C' \leftarrow C[0]C'[1]$
$M \leftarrow \mathtt{D}_K(C')$
Se M = M_0 então responda 1 então responda 0.

Então, nós afirmamos que

$$\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cca-1} = 1\right] = 1$$

$$\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cca-0} = 1\right] = 0.$$

Assim

$$\mathbf{Adv}_{\mathrm{SE},A}^{ind-cca} = 1 - 0 = 1.$$

No mundo 1, quando $b = 1$, faça $C[0]C[1]$ denotar o texto cifrado retornado pelo oráculo cifragem-lr, e $R = StN(C[0])$. Então

$$C[1] = F_K(NtS_l(R+1)) \oplus M_1 = F_K(NtS_l(R+1)) \oplus 1^L.$$

Note que

$$\begin{aligned} M &= \mathtt{D}_K(C[0]C'[1]) \\ &= F_K(NtS_l(R+1)) \oplus C'[1] \\ &= F_K(NtS_l(R+1)) \oplus C[1] \oplus 1^L \\ &= F_K(NtS_l(R+1)) \oplus (F_K(NtS_l(R+1)) \oplus 1^L) \oplus 1^L \\ &= 0^L \\ &= M_0. \end{aligned}$$

Assim o oráculo de decriptação irá retornar M_0, e A irá retornar 1. No mundo 0,

$$C[1] = F_K(NtS_l(R+1)) \oplus M_0 = F_K(NtS_l(R+1)) \oplus 0^L.$$

Note que

$$\begin{aligned} M &= \mathtt{D}_K(C[0]C'[1]) \\ &= F_K(NtS_l(R+1)) \oplus C'[1] \\ &= F_K(NtS_l(R+1)) \oplus C[1] \oplus 1^L \\ &= F_K(NtS_l(R+1)) \oplus (F_K(NtS_l(R+1)) \oplus 0^L) \oplus 1^L \\ &= 1^L \\ &= M_1. \end{aligned}$$

Assim, o oráculo de decriptação irá retornar M_1, e A irá retornar 0. O ataque no C-CTR é similar.

CCA no CBC

Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e $\mathtt{SE} = (K, E, D)$ o esquema de criptografia simétrico CBC associado, como já foi descrito. Seja $C[0]C[1]$ um texto cifrado de alguma mensagem M de L-bits, e nós invertemos o bit i do IV $C[0]$, resultando em um novo texto cifrado $C'[0]C[1]$. Seja M' a mensagem obtida pela decryptação do novo texto cifrado. Então M' é igual a M com o i-esimo bit invertido. Essa idéia pode ser usada para quebrar o sistema, dizendo em qual mundo um adversário foi colocado [Goldwasser and Bellare 2001].

Proposição 2. 32. *: Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^L$ uma família de funções e $\mathtt{SE} = (K, E, D)$ o esquema de criptografia simétrico CBC associado, como já foi descrito. Então*

$$\boldsymbol{Adv}_{SE}^{ind-cca}(t, 1, l, 1, 2l) = 1$$

para $t = O(l + L)$ mais o tempo de uma aplicação de F. A vantagem do adversário é 1 mesmo usando pouco os recursos: somente uma consulta para cada oráculo. Isso mostra claramente que o esquema é inseguro [Goldwasser and Bellare 2001].

Prova da Proposição 2.32: Vamos apresentar um algoritmo adversário A, com complexidade de tempo t, fazendo 1 consulta para o oráculo cifragem-lr. A consulta tem tamanho l, fazendo também 1 consulta ao oráculo de decriptação, esta última com tamanho $2l$, e tendo

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cca} = 1$$

O objetivo de A é determinar o valor de b.

Adversário $A^{\mathtt{E}_K(LR(.,.,b)),\mathtt{D}_K(.)}$
$M_0 \leftarrow 0^L; M_1 \leftarrow 1^L$
$C[0]C[1] \leftarrow \mathtt{E}_K(LR(M_0, M_1, b))$
$C'[0] \leftarrow C[0] \oplus 1^L; C' \leftarrow C'[0]C[1]$
$M \leftarrow \mathtt{D}_K(C')$
Se M = M_0 então responda 1 senão responda 0

Então, nós afirmamos que

$$\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cca-1} = 1\right] = 1$$

$$\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cca-0} = 1\right] = 0.$$

Assim,

$$\mathbf{Adv}_{\mathrm{SE},A}^{ind-cca} = 1 - 0 = 1$$

No mundo 1, quando $b = 1$, faça $C[0]C[1]$ denotar o texto cifrado retornado pelo oráculo cifragem-lr. Então

$$C[1] = E_K(C[0] \oplus M_1) = E_K(C[0] \oplus 1^l).$$

Note que

$$\begin{aligned} M &= \mathtt{D}_K(C'[0]C[1]) \\ &= E_K^{-1}(C[1]) \oplus C'[0] \\ &= E_K^{-1}(E_K(C[0] \oplus 1^l)) \oplus C'[0] \\ &= (C[0] \oplus 1^l) \oplus C'[0] \\ &= (C[0] \oplus 1^l) \oplus (C[0] \oplus 1^l) \\ &= 0^l \\ &= M_0 \end{aligned}$$

Assim o oráculo de decriptação irá retornar M_0, e A irá retornar 1. No mundo 0,

$$C[1] = E_K(C[0] \oplus M_0) = E_K(C[0] \oplus 0^l).$$

Note que

$$\begin{aligned} M &= \mathtt{D}_K(C'[0]C[1]) \\ &= E_K^{-1}(C[1]) \oplus C'[0] \\ &= E_K^{-1}(E_K(C[0] \oplus 0^l)) \oplus C'[0] \\ &= (C[0] \oplus 0^l) \oplus C'[0] \\ &= (C[0] \oplus 0^l) \oplus (C[0] \oplus 1^l) \\ &= 1^l \\ &= M_1 \end{aligned}$$

Assim o oráculo de decriptação irá retornar M_1, e A irá retornar 0. O ataque no C-CTR é similar.

Esses ataques mostram que devemos evitar que o adversário tenha acesso ao computador que decripta as mensagens cifradas. Permitir tal acesso compromete, severamente, a segurança de qualquer esquema.

2.3 Considerações Finais sobre o Modelo de Segurança

Este capítulo apresentou o modelo de função vantagem. Esse modelo aborda o conceito de segurança perfeita (ou segredo perfeito), objetivo da segurança da Teoria da Informação. Essa abordagem norteia os esforços de modelagem e análise de esquemas de criptografia.

O modelo de função vantagem é validado através de teoremas, o que mostra a sua força. Vários teoremas têm grande valor para aplicação do modelo, dentre eles os Teoremas 2.25 e 2.29. O que faz esses teoremas merecerem destaque é que a desigualdade é menor-ou-igual ($\leq$), não maior-ou-igual ($\geq$). Quando dizemos que um esquema tem uma vantagem maior-ou-igual, significa que não sabemos qual é o valor máximo e isso é suficiente para provar insegurança, quando o valor é grande, mas não é suficiente para provar segurança, quando o valor é pequeno. Quando dizemos que um esquema tem uma vantagem menor-ou-igual estamos impondo um valor máximo não importa quão inteligente é o adversário e isso é suficiente para provar segurança, mesmo que como é o caso aqui, em relação ao algoritmo usado. Os teoremas citados dizem que o esquema é tão seguro quanto o algoritmo que ele usam para cifrar os dados, e isto é uma afirmação importante.

Por se tratarem de conceitos complexos, o capítulo é extenso em exemplos e demonstrações. Os exemplos visam o máximo entendimento dos conceitos e aumentam a confiança no modelo.

O perfeito entendimento dos conceitos apresentados aqui é essencial para assimilar a aplicabilidade do modelo e o aproveitamento dos resultados.

Capítulo 3

Segurança das Redes Locais sem Fios

As redes sem fios são caracterizadas por não necessitarem de cabos para que haja comunicação entre os vários dispositivos da rede. Elas podem existir isoladas ou como uma extensão da rede cabeada, podendo ser rapidamente implementadas em ambientes que não possuam infra-estrutura de cabeamento, além de possibilitar a mobilidade. No final da década de 90, as corporações começaram a se interessar por tais redes. Então, para atender a esse fomento, em 1999 o IEEE (*Institute of Electrical and Electronic Engineers*) publicou o Padrão IEEE 802.11. Tal padrão descreve, para as redes locais sem fios, o ambiente e os protocolos das camadas físicas e de enlace, bem como aspectos de segurança, como confidencialidade[1] e integridade[2] do dados transmitidos [Silva 2005]. O protocolo *Wired Equivalent Privacy* (WEP), por exemplo, foi lançado junto com o 802.11 e objetiva a privacidade. No entanto, muitas vulnerabilidades foram detectadas no WEP, especialmente a apresentada por Fluhrer, Mantin e Shamir que possibilita a recuperação da chave secreta usada na cifragem dos dados [Fluhrer et al. 2001]. Esta e outras vulnerabilidades tornaram o WEP impróprio para a garantia de confidencialidade. Em resposta, o IEEE montou o grupo de trabalho IEEE 802.11i para editar novos padrões para segurança nas redes 802.11. Como padrões demoram para ficar prontos, a *Wi-Fi Alliance*, uma organização sem fins lucrativos, decidiu lançar o *Wi-Fi Protected Access* (WPA). O WPA é um padrão de mercado, cujo objetivo é sanar as vulnerabilidades conhecidas do protocolo WEP sem, no entanto, descartar o hardware existente. O grupo 802.11 continuou trabalhando e em junho de 2004 aprovou o Padrão IEEE 802.11i. O 802.11i cobre a segurança do 802.11, do WPA

[1] Assegura que a informação transmitida ou armazenada é acessível apenas para leitura pelas pessoas autorizadas, incluindo simplesmente revelar a existência, ou não, de algo em seu conteúdo [Stallings 1998].

[2] Assegura que somente pessoas autorizadas possam modificar a informação transmitida ou armazenada [Stallings 1998].

e acrescenta mais proteção. Logo após, a *Wi-Fi Alliance* lançou um padrão de mercado totalmente compatível com a especificação do 802.11i e o chamou de WPA2 [Sankar et al. 2004].

O Comitê IEEE 802.11 é responsável pelos padrões das Redes Locais sem fios (*Wireless* LANs - WLANs) e é subdividido em diversos subcomitês, conhecidos como *Task Groups* (TG). Os TGs são encarregados de desenvolver os padrões. O *Task Group e* (TGe), por exemplo, é responsável por prover em redes 802.11 suporte para aplicações que requerem QoS (Qualidade de Serviço), tais como aplicações de áudio e vídeo. Já o *Task Group i* (TGi) foi criado em março de 2001 como uma partição do TGe, para reforçar os mecanismos de segurança e autenticação do 802.11. O TGi trabalhou no Padrão IEEE 802.11i que foi finalizado e aprovado em junho de 2004 [Sankar et al. 2004].

O Padrão 802.11i reforça o Padrão 802.11, acrescentando novos mecanismos de segurança para garantir confidencialidade e integridade às mensagens. As características adicionadas são:

- dois novos tipos de redes, *Transition Security Network* (TSN) e *Robust Security Network* (RNS);
- novos métodos de criptografia e integridade de dados, *Temporal Key Integrity Protocol* (TKIP) e *Counter mode/CBC-MAC* [3] *Protocol* (CCMP);
- novos mecanismos de autenticação usando o *Extensible Authentication Protocol* (EAP);
- gerenciamento das chaves via protocolos de *handshake* seguros através do 802.1x.

Uma rede RSN é aquela que permite somente máquinas usando TKIP/Michael ou CCMP. Já uma rede TSN permite também máquinas usando WEP. Redes TSN têm uma fraqueza na transmissão de quadros de *broadcast*, já que o emissor deverá usar WEP para todos os destinatários [Sankar et al. 2004].

Este capítulo está organizado da seguinte maneira: as seções 3.1, 3.2, 3.3 e 3.4 apresentam respectivamente os padrões de segurança WEP, WPA, WPA2 e 802.1x, bem como suas vulnerabilidades. A seção 3.5 descreve o processo de ataque utilizado pelos *hackers* para atacar as redes locais sem fios e relaciona alguns dos ataques mais conhecidos.

[3] *Aqui MAC é acrônimo de* Message Authentication Code. *O código gerado será referenciado como MIC (*Message Integrity Check*) para evitar confusões. Com exceção do CBC-MAC, o acrônimo MAC fará referência a* Medium Access Control.

3.1 *Wired Equivalent Privacy* - WEP

O protocolo WEP foi lançado junto com o Padrão IEEE 802.11 e objetiva a confidencialidade dos dados, a proteção contra alterações de quadros na transmissão e o controle de acesso dos usuários à rede.

O WEP usa o algoritmo *Rivest Cipher 4* (RC4) para gerar uma chave de fluxo que é misturada aos dados com o objetivo de obter o texto cifrado que será transmitido. Um mecanismo de integridade chamado *Integrity Check Vector* (ICV)[4] também é especificado. Para controlar o acesso dos usuários à rede, o WEP é dotado de um mecanismo de desafio-resposta[5] [Sankar et al. 2004].

3.1.1 Encapsulamento do WEP

Encapsulamento é o processo de transformar os dados de uma camada de rede para outra camada mais baixa poder usar. Isso pode envolver cifragem, cálculo do ICV, fragmentação (se necessário) e concatenação de cabeçalhos. O processo contrário é o desencapsulamento, no qual o processo de encapsulamento é revertido e o quadro é passado para a camada de rede superior, o que pode envolver as tarefas de remover os cabeçalhos, decifrar, remontar quadros fragmentados e verificar o ICV [Sankar et al. 2004].

Para cifrar os dados, o algoritmo do WEP primeiro escolhe um Vetor de Inicialização (IV) (como o IV será escolhido o padrão não especifica). O mais comum é escolher o primeiro IV de forma randômica e obter os seguintes por incremento. Isso diminui a chance de haver repetição de IVs. Iniciar o IV com zero não é uma boa opção, porque ao religar o dispositivo os IVs serão reusados. Em seguida o IV é concatenado com a chave do WEP e o resultado é a semente do WEP. A semente será usada pelo RC4 para embaralhar o vetor S-box[6]. O RC4 gera então como saída uma chave de fluxo do tamanho dos dados mais quatro octetos para o ICV. Agora o ICV é calculado e concatenado à mensagem (dados que devem ser transmitidos). Os dados mais o ICV são combinados com a chave de fluxo através de operações de ou-exclusivo (XOR) e produzem um texto cifrado que será então enviado [Sankar et al. 2004].

A Figura 3.1 mostra o esquema de encapsulamento do WEP. O símbolo || significa concatenação

[4]Um tipo de *Cyclic Redundancy Check* (CRC) de 32 bits.

[5]Mecanismo para autenticação de cliente, no qual o autenticador envia um número aleatório (em aberto) que o cliente deve cifrar e devolver. O autenticador decifra a resposta e compara com o seu número, constatando que cliente sabe a chave secreta.

[6]No caso do WEP, o S-box é um vetor com valores seqüenciais de 0 a 255 que serão misturados para gerar a seqüência randômica.

de *strings* de bits, XOR indica uma operação lógica bit-a-bit de ou-exclusivo e a implementação do RC4 é identificada como "WEP PRNG" (*Pseudo Random Number Generator*).

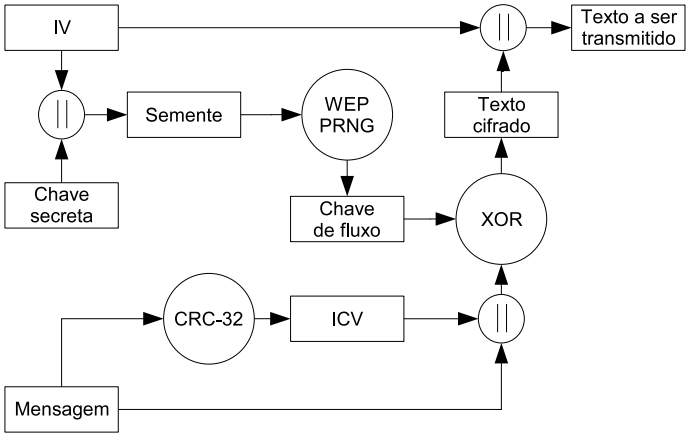

Figura 3.1: Esquema de encapsulamento do protocolo WEP

O IV é enviado de forma não cifrada, como também o cabeçalho MAC e o identificador da chave, para permitir ao recebedor decifrar os dados e verificar a sua integridade.

Quadro do WEP

O encapsulamento do WEP irá criar o quadro mostrado na Figura 3.2.

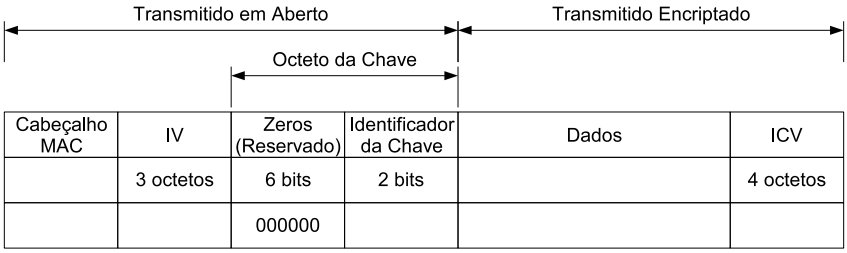

Figura 3.2: Formato do quadro do WEP

O cabeçalho MAC é seguido de três octetos do vetor de inicialização. Do octeto da chave, apenas dois bits são utilizados, eles especificam qual das quatro chaves permitidas foi utilizada na cifragem do quadro atual. O WEP permite o armazenamento de até quatro chaves diferentes no ponto de acesso (AP), então, o usuário pode especificar qual chave usou. Os dados e o ICV são transmitidos de forma cifrada [Sankar et al. 2004].

Algoritmo RC4

O algoritmo RC4 é um cifrador de fluxo que utiliza criptografia simétrica. Ele é considerado seguro, quando usado de forma apropriada, ou seja, nunca reusando a semente (a chave do RC4) e não revelando parte da semente. O algoritmo do RC4 é simples e pode ser implementado em *software* sem comprometer o desempenho. Ele produz uma chave de fluxo (*stream key*) de mesmo tamanho dos dados que devem ser cifrado. No WEP, essa chave de fluxo é altera os dados por meio de operações *exclusive-OR* (XOR) e assim é produzido o texto a ser transmitido [Sankar et al. 2004].

O RC4 trabalha com um *array* de valores chamado S-box. Os valores são misturados no S-box por uma série de operações de troca. O algoritmo é dividido em duas partes: o KSA (*Key Scheduling Algorithm*), que embaralha o S-box usando uma chave aleatória (a semente do RC4) através de várias permutações entre os elementos, e o PRGA (*Pseudo Random Generation Algorithm*), que utiliza a saída do KSA para gerar uma seqüência de saída pseudo-aleatória (a chave de fluxo), enquanto continua a embaralhar o S-box [Sankar et al. 2004]. A Figura 3.3 mostra os algoritmos internos do RC4, segundo [Silva 2005]. A semente é o vetor de bytes K e tem um tamanho de L bytes. O tamanho do S-box é representado por n, que no WEP é 256. Todas as operações são módulo n, com exceção da operação de acesso aos bytes do vetor K que, como indicado no próprio algoritmo, é módulo L.

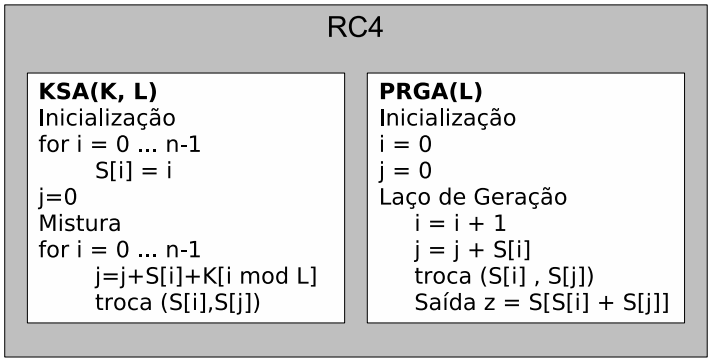

Figura 3.3: Algoritmo interno do RC4

No KSA, o S-box é criado com a seqüência de 0 a *n-1*. Os elementos são misturados por meio de operações de troca. Nesse momento a semente é usada, introduzindo assim a aleatoriedade. As variáveis *i* e *j* indicam quais elementos do vetor S-box serão trocados. Já o PRGA recebe o S-box embaralhado e calcula então novos *i* e *j*, uma nova troca é então realizada e uma saída é produzida. O laço é repetido até que se obtenha uma chave de fluxo de mesmo tamanho dos dados a serem

cifrados.

Implementações vulneráveis do KSA[7] não geram uma distribuição perfeitamente randômica. A recomendação é que algumas saídas do PRGA sejam descartadas. Segundo Paul e Preneel, para se obter uma distribuição uniforme, sugere-se desprezar de $3n$ a $12n$ bytes da saída [Paul and Preneel 2004].

É interessante observar que todo o algoritmo do RC4 é determinístico, exceto a semente que é uma escolha aleatória das partes. A semente é de conhecimento tanto do emissor quanto do receptor. Isso permite que o processo seja refeito para gerar a mesma chave fluxo novamente, o que torna possível a decifragem, como será visto a seguir.

3.1.2 Desencapsulamento do WEP

O processo de desencapsulamento é o oposto do encapsulamento (Figura 3.4).

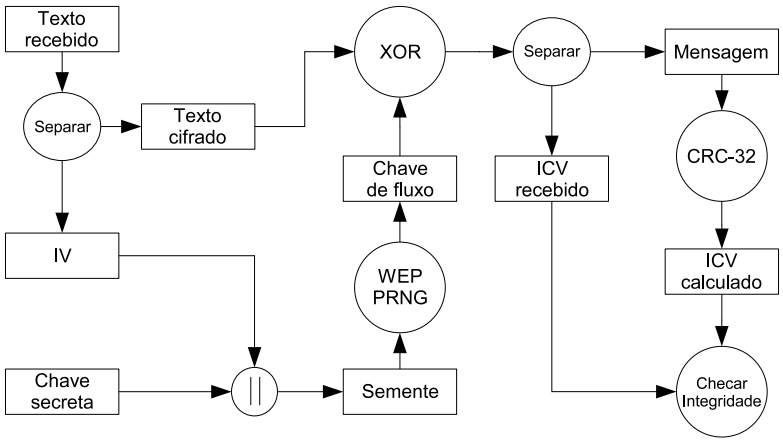

Figura 3.4: Esquema de desencapsulamento do protocolo WEP

Após a retirada do cabeçalho MAC, o IV e o identificador da chave são extraídos e separados dos dados cifrados. O IV é então concatenado à chave para formar novamente a semente e alimentar o RC4. O RC4 (indentificado como "WEP PRNG" na Figura 3.4) gera a mesma chave de fluxo do encapsulamento. A chave de fluxo é combinada com o texto cifrado, usando operações de ou-exclusivo (XOR), e surgem, assim, os dados e o ICV. No último passo, o ICV e os dados (a mensagem) são separados. O cálculo do ICV é repetido e o resultado comparado com o ICV recebido. Se os ICVs forem iguais, o destinatário assume que a mensagem não foi modificada, caso contrário, descarta a mensagem.

[7]Essas vulnerabilidade dizem respeito aos IVs fracos, como pode ser visto na Seção 3.5.2 no item Ataque ao Cifrador de Fluxo.

A garantia de que o recebedor que conhece a chave e conseguirá obter a mensagem a partir do texto cifrado é baseada nas propriedades da operação XOR. A álgebra abaixo mostra isso. Dado que:

$$TextoCifrado = Mensagem \oplus ChavedeFluxo,$$

onde $\oplus$ significa a operação XOR. É possível afirmar que:

$$\begin{aligned} TextoCifrado \oplus ChavedeFluxo &= (Mensagem \oplus ChavedeFluxo) \oplus ChavedeFluxo \\ &= Mensagem \oplus (ChavedeFluxo \oplus ChavedeFluxo) \\ &= Mensagem \oplus 0 \\ &= Mensagem \end{aligned}$$

3.1.3 Vulnerabilidades do WEP

São muitas as vulnerabilidades no WEP, isto é comprovado pelo grande número de ataques possíveis.

Os ítens abaixo sintetizam as vulnerabilidades levantadas através de ataques conhecidos ao WEP [Suriyajan 2006] [Silva 2005]:

- **gerenciamento da chave é precário:** o WEP usa uma chave estática, que deveria ser trocada pelo administrador da rede periodicamente. Essa atividade demanda muito tempo e esforço, e portanto não é feita, tornando a chave permanente. Os IVs foram acrescentados para variar a chave para cada quadro cifrado, mas como só usam 24 bits e são transmitidos sem criptografia, um atacante pode muito bem esperar até que os IVs comecem a ser repetidos;

- **manipulação dos dados:** como a manipulação dos dados é feita através de operações XOR, bits da mensagem podem ser alterados facilmente alterando alguns bits do texto cifrado. O ICV não é forte o bastante para evitar esse ataque, sendo mais apropriado para detectar erros de distorções no canal de comunicação;

- **ataque passivo para descobrir a chave secreta:** como o IV é transmitido sem criptografar e, além disso, tem tamanho limitado, todos os IVs possíveis serão utilizados e então reutilizados. A partir de dois textos cifrados com o mesmo IV e mesma chave (já que a chave é estática), o atacante pode fazer um ataque estatístico para descobrir a chave de fluxo e expandi-la para out-

ros IVs. O atacante poderia também transmitir alguns pacotes pela rede cabeada, ou até pela Internet, para algum membro da rede capturá-la cifrada pelo ponto de acesso e, assim, proceder tal ataque. A fraqueza descrita por [Fluhrer et al. 2001] pode ser usada para recuperar a chave secreta e já existem várias ferramentas que o fazem. Em redes com muito tráfego, ou usando técnicas para forçar o AP gerar mais tráfego, o WEP de 128 bits pode ser quebrado em questão de 3 a 10 minutos;

- **re-roteamento de pacotes:** se o atacante conhece padrões nos pacotes, ele pode alterar alguns bits para mudar o destino dos pacotes para alguma máquina sob o seu controle. Depois de o AP decriptar o pacote ele roteará o pacote para a máquina controlada e então o atacante terá acesso à mensagem;

- **o uso do WEP não é obrigatório pelo padrão:** assim muitas redes ficam totalmente abertas por descuido de administradores.

- **o modelo de autenticação do WEP autentica apenas o cliente:** o ponto de acesso não é autenticado perante ao cliente;

- **o esquema de comunicação utilizado não impede um ataque do tipo homem-do-meio**[8].

Na tentativa de contornar tais vulnerabilidades e encontrar mercado para seus produtos, os fabricantes de dispositivos adicionaram por conta própria alguns mecanismos que não apareciam no padrão [Suriyajan 2006] [Silva 2005]:

- **chave WEP extendida:** alguns fabricantes estenderam o tamanho da chave WEP dos 40 bits original para 104, 128 e 232 (Lucent, Agrere e US Robotic respectivamente). Este aumento da chave faz o atacante precisar de mais tempo, mas não resolve o problema já que a fraqueza está relacionada com o tamanho dos IVs, que continua limitado aos mesmos 24 bits;

- **WEP dinâmico:** o WEP dinâmico tenta gerar chaves de "tempo de vida curto" e distribuí-las na rede por *broadcast*, para dificultar o trabalho do atacante;

- **integrando uma rede privada virtual (VPN):** esse método pode ser efetivo, mas é dispendioso. Além disso, VPN não foi desenvolvido para Redes Locais sem Fios;

[8]Um ataque-do-homem-do-meio, ou MITM (*man-in-the-middle attack*), é um ataque no qual um atacante consegue ler, inserir e modificar as mensagem que são trocadas entre as duas partes, sem que elas saibam.

- **o uso da "rede fechada":** na qual o SSID não é enviado por *broadcast*, obrigando os cliente a conhecer tal identificador previamente;
- **a filtragem MAC:** onde o endereço MAC do cliente é verificado contra uma base de dados de endereços autorizados. Tanto o SSID quanto o endereço MAC são enviados de forma não-cifrada nos quadros da rede e por isto não ajudam muito na segurança da rede.

3.2 *WI-FI Protected Access* - WPA

Baseado nas vulnerabilidades do WEP, o IEEE propôs o Padrão IEEE 802.11i, que deu origem ao protocolo TKIP (*Temporal Key Integrity Protocol*). O TKIP, na sua proposta inicial, deveria resolver todas as vulnerabilidades identificadas no WEP, mas sem exigir a troca do *hardware*. A idéia era aproveitar as interfaces de rede já fabricadas. Como grande parte do WEP é implementada em *hardware*, o TKIP utiliza os mecanismos básicos do WEP, incluindo o vetor de inicialização, a criptografia do RC4 e o ICV [Sankar et al. 2004].

Enquanto a IEEE trabalhava no 802.11i, o mercado necessitava de um padrão, já que o WEP não atendia ao mínimo exigido de segurança. Foi nesse cenário que a *Wi-Fi Alliance*, uma organização sem fins lucrativos, decidiu lançar o WPA. O WPA é um sub-conjunto do IEEE 802.11i, pois foi especificado a partir do rascunho deste último, existente na época. O WPA utiliza o TKIP, que possui um bom desempenho e não exige a troca do *hardware*, apenas as atualizações do *firmware* e *software* controlador (*driver*) são necessárias [Sankar et al. 2004].

O protocolo WPA foi desenvolvido para resolver as falhas conhecidas do protocolo WEP, como as resumidas a seguir:

- Uma chave única é usada por todos os pontos de acesso e clientes.
- As chaves podem ser recuperadas com utilitários já disponíveis.
- As chaves recuperadas expõem a rede a ataques e a monitoramento passivo.
- A falta da gerência automatizada de chaves contribui para extensão da vida da chave estática de forma infinita em grandes redes.
- Quando o protocolo WEP estava disponível, nem sempre estava ligado.
- O protocolo WEP não oferece nenhuma proteção contra a falsificação de dados.

- O protocolo WEP não oferece nenhuma proteção contra a repetição de pacotes.
- O protocolo WEP emprega de forma inadequada o algoritmo de criptografia RC4 e permite assim o ataque de vetores IVs fracos.
- O protocolo WEP usa o vetor de inicialização como parte da chave e quando o vetor IV repete, os dados podem ser facilmente recuperados.

O WPA aborda as fraquezas do WEP através das seguintes características [Suriyajan 2006]:

- **estendendo o IV para 48 bits e usando regras de seqüenciamento:** o IV do WEP tem 24 bits de comprimento; assim, todas as possibilidades são utilizadas muito rapidamente e depois são usados IVs repetidos. O WPA extendeu o IV para 48 bits, aumentando assim seu ciclo de vida ($2^{48} = 281.474.976.710.656$). Além disso, o IV agora é sequencial por padrão e usado para impedir a retransmissão de quadros;
- **implementando o Michael *Message Integrity Code* (MIC):** com um MIC de 64 bits, calculado pelo algoritmo Michael, o WPA consegue detectar alterações originadas de manipulações intensionais dos bits da mensagem;
- **derivando e distribuindo as chaves:** o WPA deriva e distribui automaticamente as chaves que serão utilizadas na cifragem. Assim, é resolvido o problema de chaves estáticas do WEP;
- ***Temporal Key Integrity Protocol* (TKIP):** o TKIP foi introduzido no WPA para reforçar o cifrador do RC4 implementado no WEP. Esse protocolo gera uma chave para cada quadro usando várias entradas, como será visto nesta seção. Ele garante que o mesmo IV não será usado duas vezes.

O WPA também permite o reforço no controle de acesso, pelo uso do padrão 802.1X EAP (*Extensible Authentication Protocol*) com um servidor de autenticação, ou usando uma PSK (*Pre-Shared Key*) para ambientes SOHO (*Small Office and Home Office*), onde é inviável o uso de um servidor de autenticação.

3.2.1 Encapsulamento do TKIP

O encapsulamento do TKIP é o processo de cifrar, fragmentar, calcular os verificadores de integridade e construir o quadro. A Figura 3.5 mostra o esquema de parte do processo [Suriyajan 2006].

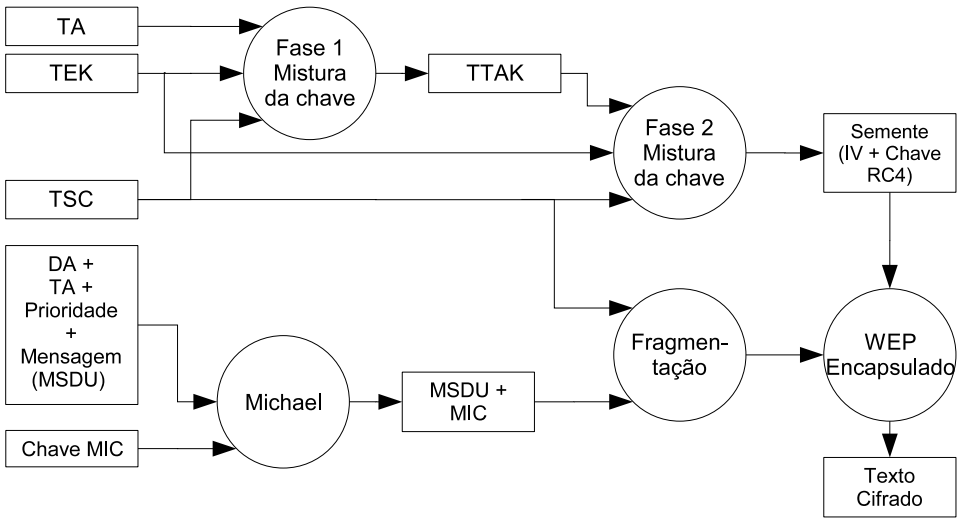

Figura 3.5: Esquema do protocolo TKIP

Primeiro o emissor calcula, o MIC através de um *hash* criptográfico chamado Michael. O MIC é um código de oito octetos calculado a partir do *MAC Service Data Unit* (MSDU)[9] concatenado com o endereço do destinatário (DA), o endereço do transmissor (TA) e um campo de prioridade. O MIC é combinado com MSDU e fragmentado (se necessário) para o tamanho do *MAC Protocol Data Unit* (MPDU)[10] que será transmitido. Por outro lado, o TKIP *Sequence Counter* (TSC) é incrementado e a semente para o WEP é gerada usando o algoritmo de mistura da chave. Cada MPDU é processado pelo algoritmo do WEP e encapsulado no quadro MAC [Sankar et al. 2004]. O formato do quadro é mostrado na Figura 3.6.

TKIP *Sequence Counter* (TSC)

A especificação do WEP não obriga os implementadores a evitar o reuso de IVs, permitindo, assim, que um atacante reuse uma parte da chave de fluxo que consiga decifrar. No WEP a chave tem um tempo de vida longo e o IV não é grande o bastante para evitar o reuso até a próxima troca de chave. Como não existe um mecanismo de gerenciamento de chaves especificado, no WEP este trabalho fica a cargo do administrador da rede [Sankar et al. 2004].

O TSC é um mecanismo que foi desenvolvido para evitar, no WPA, ataques de reenvio de quadros. Ele é um contador de quadros de 48 bits que inicia em 0 e é incrementado de 1 a cada

[9]MSDU é o conjunto de informações que desejamos transmitir para a outra estação da rede.

[10]MPDU é o que realmente é transmitido através do 802.11. Se o MSDU é muito grande, ele será fragmentado em mais de um MPDU.

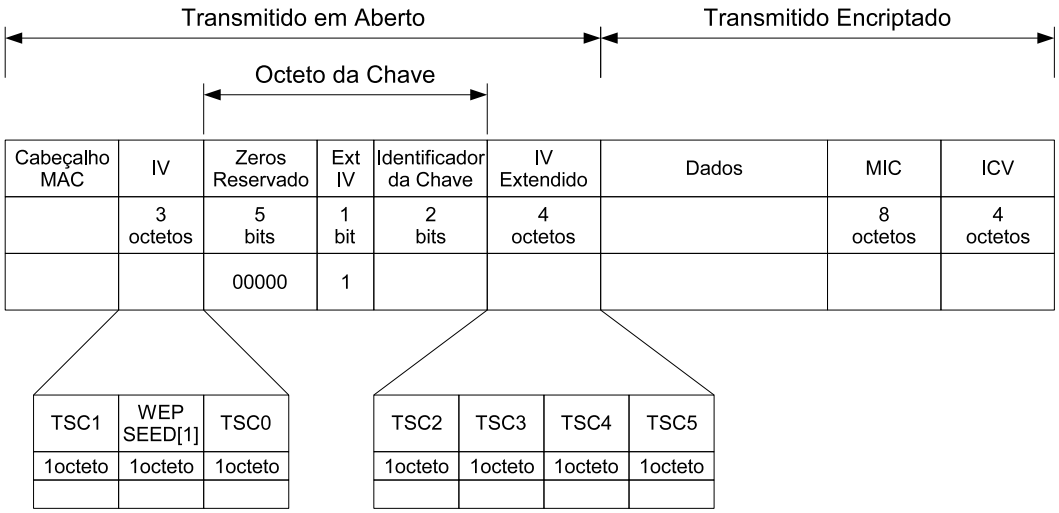

Figura 3.6: Formato do quadro MAC do TKIP

quadro. O TSC não deve se repetir com uma mesma chave, então o recebedor deve armazenar o maior valor de TSC que já foi recebido de cada origem, indentificando-a pelo seu endereço MAC. Se for recebido um quadro com um TSC menor que o último anterior, é assumido que houve um reenvio e ele é descartado [Sankar et al. 2004].

O ICV e o MIC contribuem para evitar ataques de alteração do TSC, na medida em que o TSC é usado na cifragem e decifragem do quadro e sua alteração impede a verificação do ICV e do MIC. Além disso, a especificação diz que o último TSC usado deve ser armazenado depois de verificados o ICV e o MIC, para evitar que um atacante tente enviar quadros com seqüências futuras do TSC e impeça a origem de continuar enviando quadros. A Figura 3.6 mostra o formato do quadro do TKIP. Os octetos do TSC aparecem como TSC0, TSC1, TSC2, TSC3, TSC4 e TSC5 e são transmitidos em aberto (sem criptografia). O TSC é retornado a zero cada vez que uma nova chave é renegociada [Sankar et al. 2004].

Algoritmo de Mistura da Chave

O algoritmo de mistura de chave do TKIP foi desenvolvido para proteger a *Temporal Encryption Key* (TEK). A TEK é uma chave temporária que pode ser trocada, entre estações e o ponto de acesso, através dos algoritmos de gerenciamento de chaves, e é a base para a criação da chave única por quadro. O algoritmo de mistura começa com a TEK, que é de conhecimento do emissor e do recebedor. Ela é combinada com com o TSC e o endereço do emissor (*Transmitter Address* - TA) para criar uma chave única por quadro de 128 bits, que será a semente do algoritmo do

WEP. Como o TSC é incrementado a cada quadro, então a semente do WEP muda a cada quadro também. Além disso, a especificação do algoritmo evita as conhecidas chaves fracas do RC4 (chaves que não geram uma distribuição randômica). O esquema do algoritmo é mostrado na Figura 3.7 [Sankar et al. 2004].

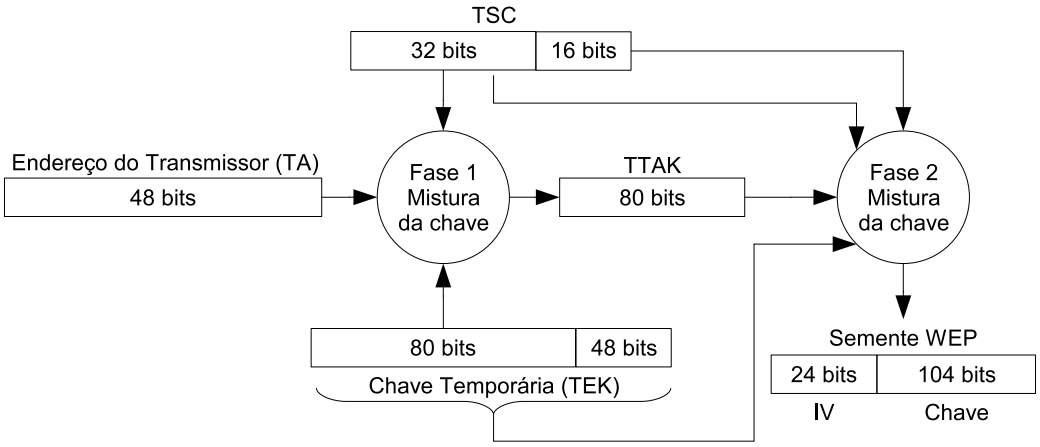

Figura 3.7: Algoritmo de mistura da chave do TKIP

Na fase 1 da mistura, os 32 primeiros bits do TSC são combinados com o TA e os primeiros 80 bits do TEK. Essa fase envolve operações de adição, XOR e AND bit-a-bit com custo baixo de processamento, mais uma S-box similar à do RC4. A fase 1 produz um valor de 80 bits que é chamado *TKIP mixed Transmit Address and Key* (TTAK). Como a única entrada nessa fase que muda é o TSC, e como não foram usados os 16 bits menos significativos dele, essa fase precisa ser executada somente uma vez a cada 64K quadros, se o TTAK for armazenado [Sankar et al. 2004].

Na fase 2, o TTAK vindo da fase 1 é combinado com o TEK completo e como o TSC completo. As operações aqui envolvidas também são de baixo custo de processamento. A saída é uma semente de 128 bits, que será usada como chave do RC4 do mesmo modo que no WEP. A forma como a semente é construída a partir do TSC evita que algumas classes de chaves fracas do RC4 sejam criadas [Sankar et al. 2004].

O algoritmo de mistura do TKIP pode ser encontrado em [Moen et al. 2004].

Michael - *Message Integrity Check* (MIC)

Outro algoritmo do TKIP é o Michael, cujo objetivo é prevenir modificações intencionais nas mensagens. A seção 3.5.2, no Ataque da Modificação da Mensagem, irá mostrar que o ICV do WEP

é um *checksum* linear[11] e, portanto, não é forte o bastante para impedir alterações intencionais nos dados transmitidos. Já o Michael é uma função de *hash* criptográfico[12] que produz um resultado que depende de uma chave. Além disso, ele também é não linear, o que quer dizer que não é possível para um atacante modificar partes da mensagem e saber quais partes do *hash* devem ser alteradas para se obter um quadro válido [Sankar et al. 2004].

É comum o uso de um algoritmo de *hash* forte como o *Secure Hash Algorithm* (SHA1) em aplicações de criptografia. Mas o SHA1 exige um certo volume de processamento e seria difícil de implementar nos pontos de acesso já existentes. Assim os desenvolvedores optaram por um algoritmo mais simples chamado Michael. O processamento exigido pelo Michael depende do tamanho do quadro, mas seu funcionamento é baseado em operações de deslocamento de bits e operações XOR, que são facilmente calculadas. Ele usa uma chave chamada chave MIC, cuja derivação pode ser vista no Apêndice A [Sankar et al. 2004].

A tamanho da saída da função *hash* deve ser tão grande quanto for necessário para que seja muito difícil criar uma outra combinação de dados que, quando executado o algoritmo do *hash*, obtenha o mesmo resultado como saída. Assim, o atacante terá uma grande dificuldade de gerar um quadro com um MIC correto sem conhecer a chave [Sankar et al. 2004].

Medidas de contenção para o Michael

O Padrão IEEE 802.11i diz que o algoritmo Michael tem uma fraca proteção contra ataques ativos (aqueles em que o atacante altera os dados e os reenvia). Assim, medidas de contenção são necessárias para combater tais ataques. Duas medidas são empregadas no TKIP: *Logging* e desabilitar/desautenticar. A primeira medida sugere que as falhas do Michael sejam registradas em um *log*, como indicação de um ataque. O ICV deve ser checado antes do Michael para evitar que o atacante crie, intencionalmente, quadros que causem falhas no Michael. A segunda medida diz que se duas falhas do Michael ocorrerem em um minuto, todas as comunicações devem ser desabilitadas e todas as estações devem se reassociar, para que sejam negociadas novas chaves. As reassociações poderão ser feitas depois de 60 segundos [Sankar et al. 2004].

[11] Estas funções podem facilmente detectar falhas acidentais, porém caso a integridade dos dados seja uma questão de segurança uma função mais elaborada ainda é necessária.

[12] Uma função de *hash* criptográfico torna difícil encontrar outro texto, de modo que ambos resultem no mesmo valor de verificação, o que dificulta muito a manipulação maliciosa da informação.

Algoritmo Michael

O MIC é calculado sobre uma espécie de MSDU ajustado. O MSDU do Michael é ajustado e nunca é transmitido, sendo usado apenas como entrada para calcular o MIC. O MSDU do Michael é formado pelo MSDU real mais alguns campos extras. Os campos adicionados protegem os dados contra modificações, pois o MIC é verificado pelo recebedor. Os campos incluídos são: os endereços MAC de origem e de destino, alguns octetos reservados que são preenchidos com zeros e um octeto de prioridade. Após adicionar o MSDU real, um octeto hexadecimal de valor 0x5A é adicionado como sinal de fim e um preenchimento de 4 a 7 octetos zerados para que o tamanho total do MSDU seja divisível por quatro octetos e termine com pelo menos quatro octetos iguais a zero. A Figura 3.8 mostra o MSDU do Michael em detalhe [Sankar et al. 2004].

Endereço da Origem	Endereço do Destino	Zeros (Reservado)	Prioridade	MSDU	Octeto de Fim	Preenchi-mento
6 octetos	6 octetos	3 octetos	1 octeto		1 octeto	4 a 7octetos
		0x000000	0x00		0x5A	0x00

Figura 3.8: MSDU ajustado do Michael

O Michael usa uma chave de 64 bits separada em duas palavras de 32 bits, Key[0] e Key[1]. O MSDU é ajustado e as duas palavras da chave são passadas como entrada para o Michael. O Algoritmo do Michael é como segue [Sankar et al. 2004]:

```
Michael(Key, MSDU)
   Key:  array[2] of int32
   MSDU: array [Length] of int32
   Length:  integer # tamanho do MSDU ajustado em palavras de 32 bits
   left := Key[0]; right := Key[1]
   for i := 0 to (Length - 1)
      left := left XOR MSDU[i]
      (left, right) := Block (left, right)
   end for
   return (left, right)
end Michael
```

A função Block é definida como:

```
Block(left, right)
    left, right:  int32
    right := right XOR (left « 17)
    left := (left + right) MOD 2^32
    right := right XOR (XSWAP(left))
    left := (left + right) MOD 2^32
    right := right XOR (left « 3)
    left := (left + right) MOD 2^32
    right := right XOR (left » 2)
    left := (left + right) MOD 2^32
    return (left, right)
end Block
```

Para os algoritmos acima, considere os símbolos e respectivos significados:

Operador	**Descrição**
:=	Atribuição
«	Rotação circular de bits à esquerda (os bits são deslocados com o bit mais à esquerda sendo inserido a partir da direita).
»	Rotação circular à direita (o oposto da rotação à esquerda).
MOD	O resto da divisão inteira
XOR	Uma operação XOR bit-a-bit
XSWAP	troca dos octetos 3 e 4 entre si e dos octetos 1 e 2 entre si
+	Adição normal de 32 bits
#	Início de comentários

3.2.2 Desencapsulamento do TKIP

O desencapsulamento é o oposto do encapsulamento, acrescentando-se a verificação de integridade. A Figura 3.9 mostra o esquema de parte do processo.

O destinatário deve recuperar o TSC que é transmitido em aberto nos campos IV e IV extendido do MPDU, como pôde ser visto no quadro MAC TKIP na Figura 3.6. Ele verifica se o TSC é maior que o TSC do MPDU anteriormente recebido. O algoritmo de mistura é executado, como no

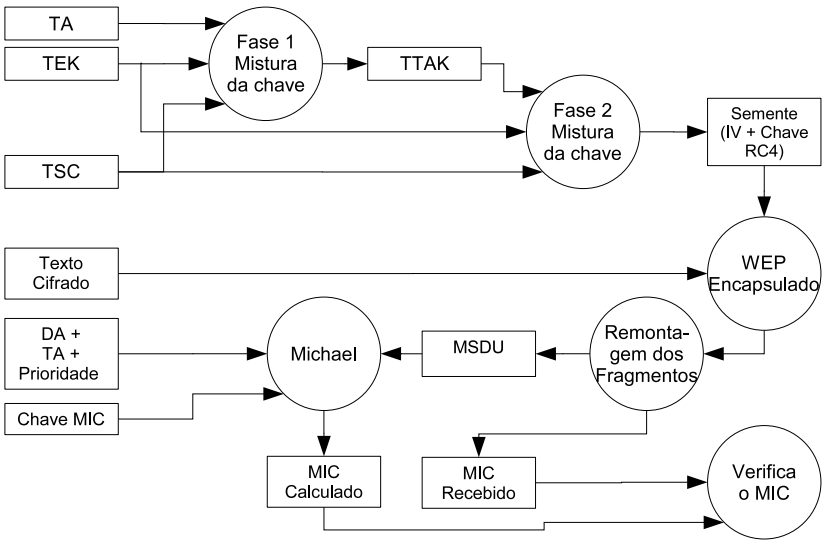

Figura 3.9: Esquema do desencapsulamento do TKIP

encapsulamento, para calcular a semente do WEP. O WEP decifra então o MPDU via RC4 e verifica o ICV. Depois de remontados os quadros, tem-se o MSDU + MIC. O MIC é então recalculado, usando o algoritmo Michael, e pode ser verificado com o MIC recebido. MICs iguais indicam sucesso na transmissão e decifragem [Sankar et al. 2004].

3.2.3 Variantes do WPA

Para prover um nível ainda maior de segurança as empresas, o WPA prevê uma autenticação em duas fases, num modo conhecido como WPA *Enterprise*:

- a primeira é um sistema aberto de autenticação;
- a segunda usa o 802.1X e o método de autenticação EAP (*Extensible Authentication Protocol* (EAP). Exige o uso de um servidor RADIUS (*Remote Authentication Dial In User Service*).

A autenticação no WPA usando IEEE 802.1X com EAP será discutida na seção 3.4.

Como alternativa mais simples, WPA também fornece autenticação usando apenas uma chave pré-compartilhada, modo conhecido como WPA *Personal*. O objetivo é reduzir custos e complexidade para pequenas empresas e uso em residências. Uma chave pré-compartilhada deve ser manualmente configurada tanto no cliente quanto no ponto de acesso. Quando o cliente conecta no AP, ele tem que provar que conhece a chave, senão ficará impedido de acessar a rede. Caso consiga provar, o ponto de acesso pode então trocar dados entre a rede e o cliente e vice-versa. O modo PSK

(*Preshared Key*) do WPA *Personal* simplifica a autenticação, mas a cifragem continua idêntica à do WPA *Enterprise*.

O WPA usa um processo chamado *4-Way-Handshake* (aperto de mão em 4 vias) para estabelecer uma conexão após um cliente requisitar a autenticação(veja a seção A.3 para mais detalhes).

3.2.4 Vulnerabilidades do WPA

O WPA apresenta vulnerabilidades menos graves do que o WEP [Suriyajan 2006] [Silva 2005]:

- **fraqueza na autenticação usando PSK:** o PSK torna fácil a configuração do WPA, mas introduz fraquezas. O PSK é um número de 256 bits ou uma frase de 8 a 63 caracteres, e é uma das entradas usadas para gerar a PMK (*Pairwise Master Key*), junto com o MAC e os números aleatórios (*nonces*). O MAC e os *nonces* podem ser obtidos durante o *4-way-handshaking*. Enviando uma mensagem de *DISASSOCIATE*, o atacante pode forçar um cliente a se reassociar e então capturar as informações necessárias. Se a PSK é pequena e fácil, pode ser conseguida através de um ataque de dicionário. Para evitar esse ataque, a PSK deve ter mais de 20 caracteres [Fleishman 2003].

- **fraqueza no *Hash* da chave temporal:** [Moen et al. 2004] mostra que é possível recuperar uma TK (*Temporal Key*) uma vez que se possua algumas chave RC4 geradas pelo TKIP. O algoritmo descrito em [Moen et al. 2004] reduz a complexidade de um ataque força bruta de $O(2^{128})$ para $O(2^{105})$. Apesar de $O(2^{105})$ ser considerado impraticável, é um ganho considerável.

- **fraqueza no MIC:** Existe ainda um ataque de criptoanálise diferencial, que explora fraquezas no algoritmo Michael. Tal ataque é considerado impraticável com o poder de computação atual [Suriyajan 2006].

- **fraqueza no mecanismo de proteção do Michael:** ao receber mais de um quadro de mesma origem com numeração repetida, desativa a operação da rede de forma temporária. Assim, alguém que transmita dois quadros por minuto pode provocar sua desativação permanente.

3.3 Padrão IEEE 802.11i ou WPA2

O protocolo WPA2 foi implementado com base na versão final do IEEE 802.11i, incluindo todas as características disponíveis no WPA e acrescentando outras, como o *Advance Encryption Standard in Counter Mode with CBC-MAC Protocol* (AES-CCMP) [Suriyajan 2006]. O Padrão IEEE 802.11i é considerado o padrão mais seguro com respeito às Redes Locais sem fios.

O nome CCMP é um acrônimo baseado em outros acrônimos [Sankar et al. 2004]:

AES: *Advanced Encryption Standard*. Um método padrão de criptografia muito forte.

CTR: O modo contador usado com o AES para obter sigilo.

CBC-MAC: *Cipher Block Chaining Message Authentication Code*. Um outro tipo de *hash* criptográfico usado com o AES para garantir a integridade das mensagens.

CCM: Abreviação de CTR/CBC-MAC, combina CTR e CBC-MAC e usa o AES para obter sigilo e integridade juntos.

CCMP: CCM Protocol, ou *Counter Mode/CBC-MAC Protocol*. O algoritmo de segurança do Padrão IEEE 802.11i que usa o protocolo CCM.

Os objetivos do projeto do Padrão IEEE 802.11i são relacionados a seguir [Suriyajan 2006]:

- desenvolvimento aberto para todos, sem algoritmos secretos;
- o desenvolvimento deve abordar as característica esperadas pelo mercado, como resolver as vulnerabilidade do WEP, não abordar problemas de comercialização, prover compatibilidade retroativa e posterior e entregar tão rápido quanto possível;
- não duplicar trabalhos feitos por outros, como por exemplo o *Internet Engineering Task Force* (IETF).
- arquitetura flexível adaptável a grandes empresas, pequenos negócios, residências etc;
- realizar revisões externas de projeto para minimizar as chances de um outro WEP.

3.3.1 Encapsulamento do CCMP

O processo de encapsulamento cifra os dados e acomoda-os de forma apropriada nos cabeçalhos para a transmissão. O encapsulamento de CCMP prover sigilo, integridade e previne reenvio de

quadros [Sankar et al. 2004].

O sigilo é alcançado pelo AES no modo CTR e pela proteção da chave, que garantem que alguém sem a chave não consiga ler a mensagem que foi transmitida. O CCM calcula um *Message Integrity Check* (MIC) que assegura a integridade dos dados. O MIC do CCM é diferente do TKIP, pois usa o AES no modo CBC-MAC e é executado sobre uma parte do cabeçalho MAC, o que impede que o cabeçalho MAC seja modificado durante a transmissão. O CCMP usa um contador de quadros incrementado, chamado PN, para evitar reenvio de quadros [Sankar et al. 2004].

Modo Contador com o protocolo CBC-MAC - CCMP

O WPA2 requer suporte ao AES usando o protocolo CCMP. Segundo [Chaplin et al. 2005] , um custo de aproximadamente 40 instruções/byte no software requer novo hardware nos pontos de acesso.

O protocolo CCMP foi desenvolvido para executar a tarefa equivalente a do TKIP. Ele prepara e fornece os dados para o AES cifrar. Por usar o cifrador AES que não necessita de uma chave por quadro, o esquema do CCMP se apresenta mais simples que o TKIP. A Figura 3.10 mostra o diagrama do funcionamento do CCMP segundo [Suriyajan 2006].

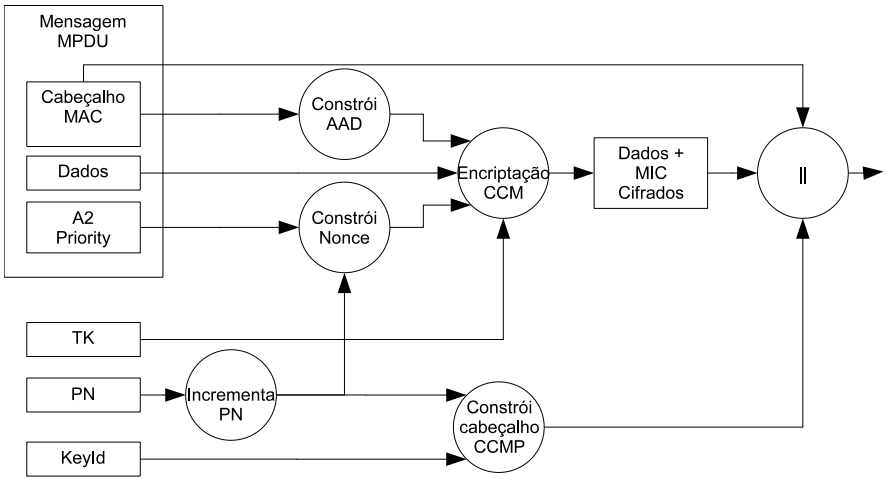

Figura 3.10: Esquema de encapsulamento do protocolo CCMP

O CCMP prepara os elementos necessários a criptografia AES/CCM, que são: um Dado Adicional de Autenticação (AAD), um número aleatório (*nonce*) e uma mensagem. O AAD inclui parte do cabeçalho MAC, enquanto o *nonce* é criado concatenando o campo de prioridade mais o endereço do emissor (A2) e um número de quadro (PN). O cabeçalho CCMP é construído da combinação do

PN e um KeyID, e tudo é concatenado para formar o texto a ser transmitido [Davies 2005].

O 802.11i inclui o CBC-MAC com AES que fornece uma integridade forte dos dados. O algoritmo CBC-MAC calcula um valor de 128 bits, e o WPA2 usa os 64 bits de ordem superior como MIC. O WPA2 criptografa o MIC junto com a mensagem no modo contador do AES [Davies 2005].

O MPUD do CCMP é similar ao usado pelo TKIP, como pode ser visto na Figura 3.11.

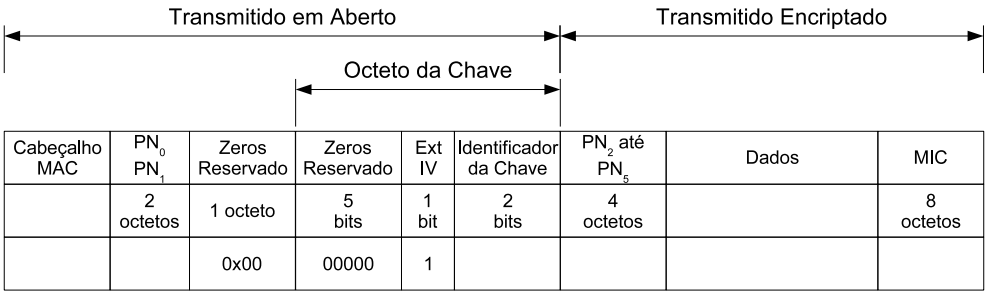

Figura 3.11: Formato do MPDU do CCMP

O tamanho do cabeçalho do CCMP é de oito octetos. O CCMP quebra o PN (Número do Quadro) em duas partes e coloca os dois octetos menos significativos no início do cabeçalho e os restantes no IV extendido, depois do identificador da chave. O PN é transmitido em aberto. Na seqüência, vêm os dados e o MIC, que são transmitidos cifrados [Sankar et al. 2004].

CBC-MAC do AES para calcular o MIC

O WPA2 usa o modo CBC-MAC para calcular o MIC e assim garantir a integridade da mensagem. O CBC-MAC cifra um bloco inicial de 128 bits com o AES e a chave de integridade de dados, produzindo um resultado de 128 bits (Resultado1). Depois é executada uma operação XOR entre Resultado1 e os primeiros 128 bits de dados para os quais o MIC está sendo calculado, o que produz um resultado de 128 bits (XResultado1). O XResultado1 é criptografado com o AES e a chave de integridade de dados, gerando o Resultado2. Um XOR entre Resultado2 e os 128 bits de dados seguintes é executado, e isso resulta no XResultado2. As duas últimas etapas se repetem para os blocos de 128 bits adicionais dos dados. Os 64 bits de ordem superior do resultado final são o MIC do WPA2. A Figura 3.12 mostra o esquema do CBC-MAC.

A Figura 3.13 mostra a entrada do CBC-MAC. O bloco inicial (128 bits) é mostrado na Figura 3.14. O cabeçalho MAC é o cabeçalho MAC 802.11 com os valores dos campos que podem ser

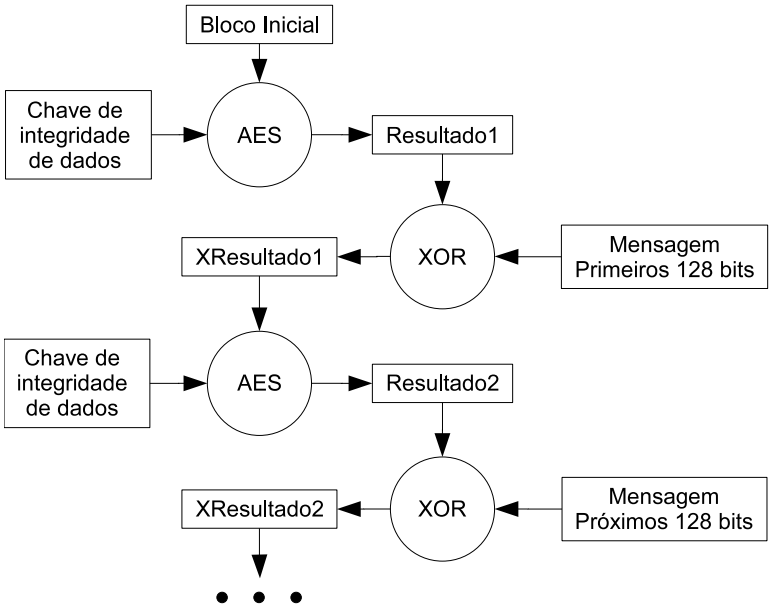

Figura 3.12: Esquema CBC-MAC do AES para calcular o MIC

alterados em trânsito definidos como 0[13]. O cabeçalho CCMP tem 8 bytes para o número do quadro de 48 bits e campos adicionais. Os bytes de preenchimento (definidos como 0) são adicionados para garantir que seja um número integral de blocos de 128 bits [Davies 2005]. Os dados são a parte de texto (mensagem a ser transmitida) não cifrada.

Bloco Inicial	Cabeçalho MAC	Cabeçalho CCMP	Preenchimento	Dados	Preenchimento

Figura 3.13: A entrada do CBC-MAC do AES para calcular o MIC.

Flag	Prioridade	Endereço da Origem	Número do Pacote	Tamanho dos Dados

Figura 3.14: Bloco inicial de 128 bits para o cálculo do MIC.

Modo Contador do AES para criptografar os dados

O WPA2 usa o AES no modo C-CTR para cifrar os dados. Primeiro o WPA2 cifra um contador inicial de 128 bits com o AES usando a chave de criptografia de dados, produzindo um resultado de 128 bits (Resultado1). Em seguida, é executada uma operação XOR entre Resultado1 e o primeiro bloco

[13] Alguns campos do cabeçalho MAC 802.11 podem ser alterados em trânsito, por algum roteador por exemplo, então esses são zerados antes de se calcular o MIC para que tais alterações não tornem o MIC inválido no destinatário.

de 128 bits dos dados que estão sendo criptografados, resultando no primeiro bloco criptografado de 128 bits. Depois o contador é incrementado e criptografado com o AES usando a chave de criptografia de dados, o que produz o Resultado2. Em seguida, é executado um XOR entre Resultado2 e os 128 bits de dados seguintes e obtém-se o segundo bloco criptografado de 128 bits. O modo de contador do AES repete as duas últimas etapas para os blocos de 128 bits adicionais da mensagem, até o bloco final. Para o bloco final, o modo de contador do AES executa o XOR do contador criptografado com os bits restantes, produzindo dados criptografados do mesmo comprimento que o último bloco de dados [Davies 2005]. A Figura 3.15 mostra o esquema do CCTR-AES.

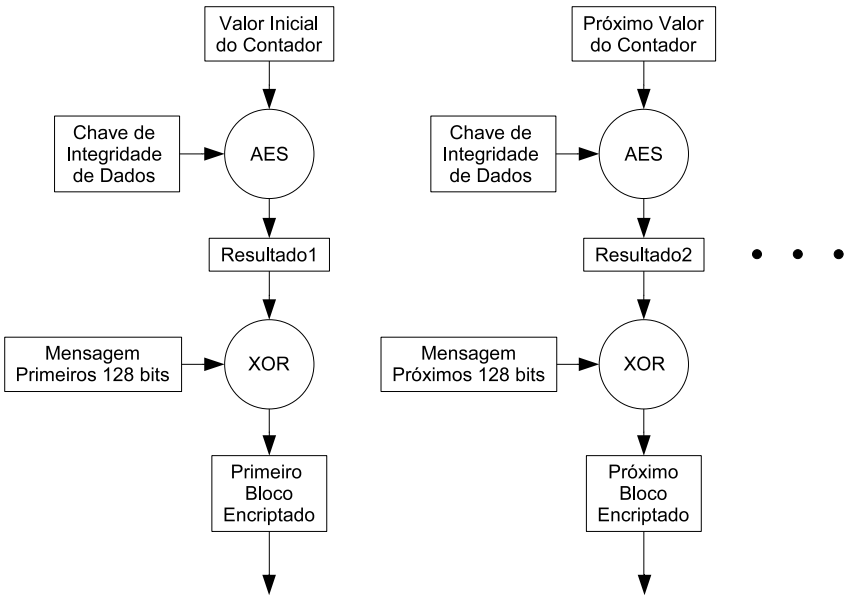

Figura 3.15: Esquema do modo contador com o AES para cifragem dos dados

O valor inicial do contador do modo de contador do AES é mostrado em detalhe na Figura 3.16. O campo *Flag* (8 bits) é definido como 01011001, que é o mesmo valor de *Flag* usado para o cálculo do MIC, seguido do campo de Prioridade de 8 bits (reservado para finalidades futuras e definido como 0). Logo após, vem o endereço de origem do cabeçalho MAC 802.11 (48 bits) e o número do quadro do cabeçalho CCMP (48 bits). No final, aparece um campo Contador (16 bits) que é definido como 1 e será incrementado apenas se uma carga do 802.11 for fragmentada em cargas menores [Davies 2005].

Flag	Prioridade	Endereço da Origem	Número do Pacote	Contador

Figura 3.16: Valor inicial do contador de 128 bits para o CCTR-AES.

3.3.2 Desencapsulamento do CCMP

O processo de desencapsulamento é essencialmente o oposto do encapsulamento e pode ser visto na Figura 3.17.

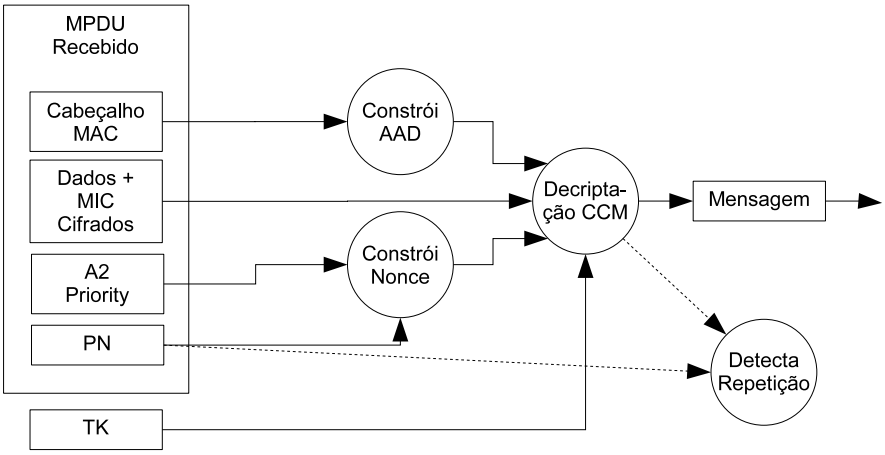

Figura 3.17: Desencapsulamento do CCMP

O recebedor pode extrair o PN do cabeçalho do CCMP e verificar se foi incrementado e, assim, detectar ataques de reenvio de quadros. O recebedor calcula o *nonce* e o AAD. A chave temporal, que é de conhecimento de ambas as partes, é utilizada no algoritmo CCM para decriptar os dados e o MIC. O CCM pode recalcular o MIC e conferir com o MIC recebido.

3.3.3 Vulnerabilidades do WPA2

Segundo [Suriyajan 2006], apesar de o IEEE802.11i (também conhecido como WPA2) ser a abordagem mais segura em uso para Redes Locais sem fios, ainda existe a possibilidade de ataques DoS (*Denial of Service*).

3.4 Padrão IEEE 802.1X

O IEEE 802.1X é um padrão que implementa o *Extensible Authentication Protocol* (EAP) para identificação do cliente. O EAP pode usar muitos métodos de autenticação tais como: *Transport Layer Security* (TLS), *Challenge-Handshake Authentication Protocol* (CHAP), *Message-Digest algorithm 5* (MD5), *Kerberos*, *certificates*, *onetime passwords*, etc. O 802.1X suporta autenticação para grandes redes, usando um servidor de autenticação, tal como RADIUS[14], para dispensar o trabalho do ponto de acesso, já que este tem limitações de capacidade e de processamento. A princípio nenhum cliente tem acesso a rede, apenas pacotes EAP são permitidos. O esquema funciona como OSA - *Open System Authentication*, permitindo que qualquer cliente se conecte ao ponto de acesso, para posterior autenticação. Depois de ser autenticado, o cliente pode enviar pacotes não-EAP pela rede. A Figura 3.18 mostra o controle de portas exercido pelo ponto de acesso [Suriyajan 2006].

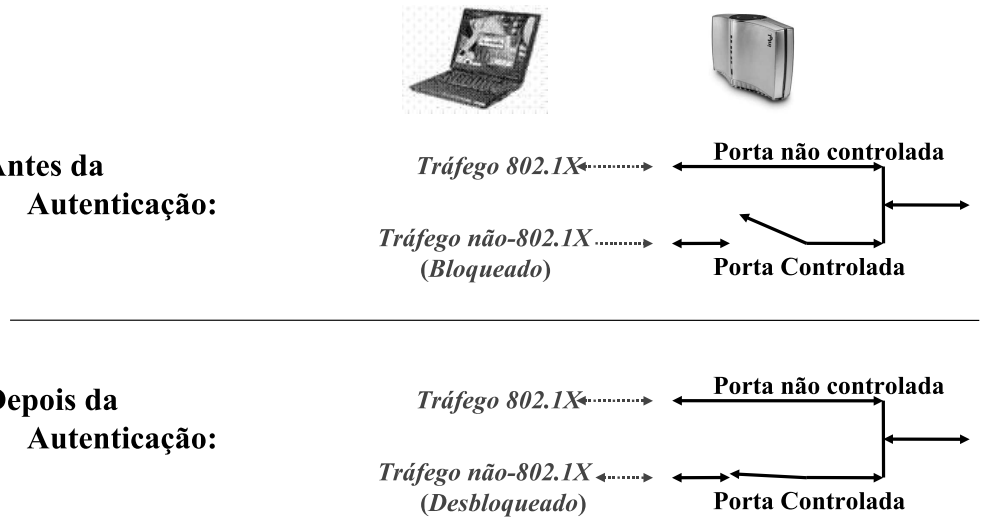

Figura 3.18: Controle de portas do 802.1X.

Quando um cliente que quer ter acesso a uma Rede Local sem fios usando 802.1X, envia um *EAPstart* para o ponto de acesso, que responde com um *EAPrequest identity*. O cliente envia então um *EAPidentity*, que o ponto de acesso repassa para um servidor de autenticação. O servidor irá verificar a identificação do cliente para o ponto de acesso. Se o cliente está autorizado a se juntar à rede, o servidor informa ao ponto de acesso. O ponto de acesso envia um *EAPsuccess* para o cliente e muda o estado do cliente para autorizado, o que permite o tráfego de pacotes normais entre o cliente e a rede cabeada. Caso o cliente não esteja autorizado o ponto de acesso envia um *EAPreject*

[14]O Radius (Remote Authentication Dial-In User Service) é um sistema de autenticação de utilizadores de redes.

ao cliente. No caso das credenciais do cliente serem válidas, o servidor já distribui automaticamente chaves de cifragem para o AP e o cliente, usando para isto um processo *4-way-handshake*, que pode ser visto no Apêndice A. A Figura 3.19 ilustra a troca das mensagens.

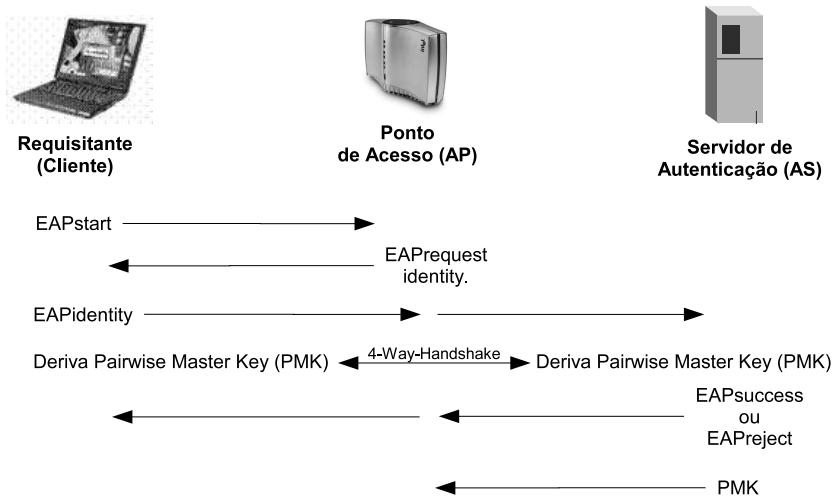

Figura 3.19: Fluxo de mensagens do 802.1X.

O Padrão IEEE 802.1X foi criado para controla o acesso por portas em redes cabeadas, sendo possível também a sua utilização em redes sem fio. No entanto, como não existe a porta e o cabo fisicamente ligados, no caso de redes sem fios, torna-se possível a captura, adulteração e repetição de pacotes de validação, o que é mais improvável em redes cabeadas. A segurança de dados trafegados entre o cliente e o ponto de acesso pode ser reforçada pelos fabricantes, mas não fazem parte do padrão [Silva 2005].

Análises dos aspectos de segurança do 802.1X podem ser encontradas em [MISHRA and W. 2002].

3.5 Ataques às Redes Locais sem Fios

3.5.1 Processo de Ataque às Redes Locais sem Fios

Um atacante pode ter muitas razões para ter as redes locais sem fios como alvo. Ele pode querer acessar os recursos oferecidos pela rede, como arquivos confidenciais. Alguém pode simplesmente querer usar a rede para acessar a Internet por exemplo, se estiver viajando e não quiser pagar pelo acesso, ou se quiser enviar milhares de e-mails sem ser rastreado, ou para espalhar um vírus. Em uma outra situação, o atacante pode querer derrubar a rede, por simples vandalismo, revanchismo ou para

tirar um adversário da caminho. Também é possível que haja a combinação de várias destas razões [Sankar et al. 2004]. Sejam quais forem as razões, os atacantes (*hackers*) seguem um processo. Para uma rede local sem fios o processo de ataque bem aceito na comunidade *hacker* tem sido baseado em: primeiro encontrar uma rede, depois planejar um ataque e, finalmente, retornar para tentar quebrá-la [Earle 2006].

Recolhendo as Informações

O primeiro passo é entender como os *hackers* encontram uma rede-alvo e como encontram informações sobre a rede-alvo. Muito provavelmente um *hacker* interessado em redes locais sem fios irá procurar alvos com um *notebook* dotado de uma placa de rede sem fios, dirigindo um carro (ataque conhecido como *wardriving*). Ele pode querer testar a segurança das redes ou apenas localizar uma conexão gratuita para acessar a Internet. Para detalhes sobre *wardriving*, veja o capítulo 2 de [Earle 2006].

Após alguns desses *wardrivings*, o atacante pode decidir planejar um ataque a uma determinada rede. Para recolher informações, ele irá usar a grande quantidade de informações já colocada na Internet. Muitas pessoas não têm consciência das possibilidades que a Internet oferece para se encontrar informações sobre as empresas e as pessoas que trabalham nelas. Um atacante procura por funcionários de TI (Tecnologias da Informação), que precisam de ajuda para algum produto. Isso pode revelar quais produtos de rede a empresa está usando. Se ele descobre que uma empresa tem ligação com outra, esta segunda pode ter um sistema de segurança fraco e pode servir de porta dos fundos para a rede-alvo. As informações de registros de domínios são públicas e podem ser usadas pelo atacante. Ferramentas como *whois*, *nslookup*, *dig* e *Sam Spade* podem executar buscas e encontrar este tipos de dados [Earle 2006].

Enumerando a Rede

Depois de coletar informações, o *hacker* procura por qualquer coisa que possa conectá-lo, tentando entender qual tipo de produto é, o que a empresa faz com ele e qual versão de *software* ou *firmware* ele está rodando. O *wardriving* é novamente usado aqui para pesquisar a rede. A enumeração pode ser limitada pela capacidade do atacante pesquisar a rede. É possível que o atacante tenha que ir para a próxima fase para comprometer um dispositivo que esteja impedindo que se prossiga na enumeração e tenha que voltar a esta mais tarde para continuar. Na enumeração, um atacante

pode usar um *scanner* de portas para encontrar os dispositivos de rede e suas portas TCPs ou UDPs abertas, ou pode usar um outro tipo de *scanner*, como os que usam ICMP e SNMP, para mapear a rede e seus dispositivos [Earle 2006].

Comprometendo os Dispositivos

Agora o *hacker* ataca a rede local sem fios ou um dispositivo da rede para se apoderar dele, isto é, comprometê-lo. Se o atacante pode comprometer um ponto de acesso da rede sem fios, ele pode entrar na rede e ir em frente com seus objetivos. As ferramentas existentes para esta fase são baseadas nas vulnerabilidades já conhecidas. O problema é que depois que uma vulnerabilidade é descoberta e publicada, em algum tempo os dispositivos terão solução para o problema, e estas ferramentas não terão mais efeito. Para se ter sucesso nessa fase, o atacante pode precisar ele mesmo construir tal ferramenta, ou buscar em *Web sites* de *hackers* ou canais de bate-papo [Earle 2006].

Expandindo os Privilégios e a Acessibilidade

O processo de expandir os privilégios se dá pelo ataque a um dispositivo (um servidor por exemplo), como um usuário restrito e tirando vantagens de falhas de segurança para elevar o nível de privilégios para irrestrito. Para expandir a acessibilidade, o atacante pode inserir *backdoors* (portas dos fundos), com o objetivo de facilitar o acesso externo. Ferramentas como VNC e NCAT podem permitir que o atacante acesse um *prompt* de comandos remotamente [Earle 2006].

Apagando os Rastros

Se a empresa detecta a existência de um atacante, ela irá vasculhar os arquivos de *log* na tentativa de rastrear e identificar o *hacker*. Para não deixar rastros que possam ser seguidos, o atacante deve limpar os arquivos de *log* e realizar outras técnicas de limpeza para apagar qualquer sinal de sua existência. O atacante tentará encobrir seus rastros, alterando os arquivos de *log* dos pontos de acesso. Qualquer servidor que o *hacker* tentar acessar terá *logs* que registrarão suas tentativas. Se existir um servidor de *log*, significa que todos os dispositivos enviarão seus *logs* para ele e o atacante terá que comprometer este servidor para apagar todos os seus rastros [Earle 2006].

3.5.2 Ataques ao WEP

O WEP falha em cada uma das três áreas: confidencialidade, disponibilidade e integridade. Quanto à confidencialidade, existem os ataques da mensagem conhecida, ataque da autenticação da chave compartilhada, ataque da dupla cifragem, ataque do homem do meio e um ataque de dicionário, dentre outros. Quanto à disponibilidade existe uma série de ataques de *denial-of-service* (DoS). E quanto à integridade, existem alguns ataques de modificação de mensagens [Earle 2006].

Ataque ao Cifrador de Fluxo (*Stream Cipher Attack*)

Esse primeiro ataque à confidencialidade do WEP é também conhecido como FSM, as iniciais dos autores do artigo que revelou a vulnerabilidade utilizada [Fluhrer et al. 2001]. O ataque já foi implementado nas ferramentas *WEPCrack*, *Airsnort*, *BSD-AIR Tools* e outras. Como o ataque é feito em modo passivo, ele é praticamente impossível de ser detectado. Os dispositivos que executam esse ataque não necessitam transmitir nada, apenas escutar o tráfego da rede no ar [Earle 2006].

Para o ataque funcionar, ele tira proveito de uma quantidade de Vetores de Inicialização (IVs), chamados fracos. O IVs são usados pelo WEP para gerar um chave de fluxo diferente por quadro. Existem aproximadamente 9000 números de IVs interessantes de um total de 16.777.216 disponíveis. Os números de IVs são considerados interessantes se eles têm o valor FF no meio dos seus três grupos de dois dígitos hexadecimais, como por exemplo: 3A:FF:5E. Se alguém olhar o pacote IP verá que um cabeçalho 802.2 SNAP é obrigatório. O cabeçalho tem o valor 0x88. Se a mensagem é conhecida e seu correspondente cifrado também, isso permite um ataque de texto plano conhecido, como explicado na próxima seção, e resulta na recuperação de parte da chave. Uma vez que os primeiros bits da chave são conhecidos, descobrir os próximos bits se torna um jogo de adivinhação. O jogo começa com 5 por cento de chance de acerto, mas à medida em que mais bytes da chave são apresentados, a chance de acerto aumenta rapidamente. O artigo [Fluhrer et al. 2001] apresenta mais detalhes sobre a matemática da solução e o artigo [Stubblefield et al. 2002] apresenta como implementar a solução.

Ataque da Mensagem Conhecida (*Known Plaintext Attack*)

Esse ataque é semelhante ao ponto inicial do ataque FSM. Quando se tem tanto a mensagem aberta como sua correspondente cifrada, é possível aplicar esse ataque e então derivar a chave.

Quando a mensagem é cifrada, o WEP executa operações de XOR para misturar a chave com os dados. O mesmo processo usado para cifrar é usado para decifrar. Se ambas as mensagens são conhecidas, a aberta e a correspondente cifrada, uma operação XOR entre as duas irá revelar a chave. Isto é conhecido como *plaintext cryptanalysis attack*. Veja a demonstração na Tabela 3.1 [Earle 2006].

10011101010010101010001010101 0	<-	Mensagem em Aberto
		XOR
11101001011001110110101101010 1	<-	Chave de Fluxo
		é igual a
01110100001011011100100111111 1	<-	Mensagem Cifrada
10011101010010101010001010101 0	<-	Mensagem em Aberto
		XOR
01110100001011011100100111111 1	<-	Mensagem Cifrada
		é igual a
11101001011001110110101101010 1	<-	Chave de Fluxo

Tabela 3.1: Detalhe da Criptoanálise da Mensagem Conhecida

Um cenário para esse ataque é a autenticação de chave compartilhada, onde o autenticador envia um desafio em texto aberto e espera um texto cifrado para verificar se a estação que quer se autenticar tem a chave compartilhada. Veja o exemplo abaixo, no qual Alice é o usuário legítimo que deseja conectar-se a rede e Malice é um invasor [Earle 2006]:

1. Alice tenta se conectar com a rede.
2. O ponto de acesso envia um desafio em aberto.
3. Alice recebe o desafio, cifra com a chave WEP e envia de volta para o ponto de acesso.
4. O Malice extrai o IV (enviado em aberto) e obtém a chave através de operações de XOR entre o desafio e a resposta de Alice.
5. Agora o Malice tenta conectar na rede.
6. O ponto de acesso envia um desafio para o Malice.
7. O Malice cifra o desafio com a chave obtida e responde para o ponto de acesso que aceita a resposta como correta.
8. Agora o Malice está conectado à rede.

Ataque da Construção de um Dicionário (*Dictionary Building Attack*)

Executando o ataque anterior repetidas vezes, é possível coletar informações suficientes para montar um dicionário de chaves de fluxo do WEP. Tendo feito isso, qualquer quadro criptografado com um IV repetido pode ser decriptografado, se a mesma chave secreta do WEP se mantiver por um período longo de tempo. Um dicionário necessitaria de um banco de dados de 24 Gbytes para armazenar todas as possíveis chaves de fluxos para uma única chave secreta WEP, o que é possível armazenar até em um *notebook* [Earle 2006].

Ataque da Dupla cifragem (*Double Encryption Attack*)

Esse ataque é possível porque a mesma chave, e também o mesmo processo, é usado tanto para cifrar com para decifrar. Primeiro um quadro com alguma informação valiosa é capturado no ar. A atacante muda o cabeçalho alterando o endereço MAC do destinatário para um outro cliente da rede local sem fios e espera até que o IV seja reiniciado e chegue a um número menor que o IV original. Então o quadro é novamente enviado pelo ar. Quando o ponto de acesso vê tal quadro, com o IV esperado, ele executa o processo de cifragem, o que resultará na decriptação do quadro e o envia pelo ar de forma aberta podendo ser capturado pelo atacante [Earle 2006].

Ataque da Modificação da Mensagem (*Message Modification Attack*)

No processo do WEP, o ICV é usado para verificar a integridade na mensagem. Por padrão, o ponto de acesso irá desconsiderar um quadro com um ICV errado. Como o ICV é independente da chave secreta ou do IV, fica fácil fazer modificações na mensagem e recalcular o ICV.

Primeiro, o atacante deve capturar um pacote destinado a uma sub-rede diferente da atual. Tal pacote é examinado pelo roteador e se houver algum problema no formato ele é desconsiderado e uma resposta padrão é retornada ao emissor. O atacante altera um único bit na mensagem e a reenvia. O ponto de acesso (AP) irá desconsiderar o pacote e não manterá qualquer registro do ocorrido. O atacante pode então fazer várias tentativas de alterações de bits até que encontre um que o AP aceite como ICV válido. O AP aceita o pacote como válido e o encaminha para o roteador. Este último considera o formato do quadro inválido e responde com uma resposta padrão, que será enviada pelo AP através do ar de forma cifrada. Agora o atacante sabe a resposta padrão e também tem a mensagem cifrada e pode executar um Ataque da Mensagem Conhecida para recuperar a chave de fluxo [Earle 2006].

Ataque de *Denial-of-Service* (DoS)

Uma das características encontradas em todas as rede 802.11 é a existência de quadros de controle. Esses quadros dizem aos clientes da rede que eles podem conectar ou que eles devem desconectar, por exemplo. Um quadro de desautenticação irá desassociar um cliente do seu ponto de acesso. No WEP, estes quadros são transmitidos em aberto. Qualquer um pode então forçar usuários legítimos da rede a se desassociarem. Alguém pode simplesmente capturar o último quadro de desautenticação e reenviá-lo. Tudo o que é requerido é a habilidade de emitir um quadro de desautenticação a um cliente *wireless*. Uma ferramenta chamada WLAN-JACK pode executar tal tarefa. Repetindo o reenvio com uma certa freqüência, o atacante pode impedir que o cliente tenha acesso à rede. [Earle 2006].

3.5.3 Ataques ao WPA

Atacando o Michael

O procedimento de medida de contenção apresentado na seção 3.2.1 diz que se o ponto de acesso receber dois MICs errados em menos de 1 segundo, o ponto de acesso deve parar as transmissões e desautenticar todos os clientes. Depois ele deve esperar 60 segundo e solicitar novas autenticações. Um atacante pode enviar dois quadros errados a cada minuto, derrubando a rede até que seja detectado, o que é um trabalho difícil, já que o sinal enviado é pequeno para ser rastreado [Earle 2006] [Silva 2005].

Atacando o WPA e 802.11i

Em 2003, Robert Moskowitz descobriu que um ataque de dicionário é possível no WPA e 802.11i, quando configurados para usar uma chave secreta pré-compartilhada. Nesse cenário, durante o *4-way-handshake*, são criadas as chaves de sessão a partir de uma chave mestra (a seção A.3 detalha bem o processo). O processo envolve a chave mestra, dois *nonces* e os endereços MAC do emissor e do recebedor. Para o atacante refazer a operação, ele precisa conhecer o SSID o tamanho do SSID e a chave secreta pré-compartilhada. Ele deve então observar o *4-way-handshake* usado para criar as chaves de sessão. Na segunda mensagem, são enviados dois valores PTK e KEK depois de passarem pela função *hash* MD5. O atacante pode tentar várias chaves até encontrar uma que tenha o mesmo MD5 [Earle 2006]. Para evitar esse ataque, a PSK deve ter mais de 20 caracteres [Fleishman 2003].

3.5.4 Ataques ao WPA2

Ataque da mensagem 1 no *4-way-handshake*

É possível fazer um ataque do tipo DoS quando o processo *4-way-handshake* pode ter múltiplas instâncias executando em paralelo. O atacante pode simplesmente enviar a mensagem 1 forjada repetidamente, fazendo com que a *Pairwise Master Key* (PTK), da mensagem forjada, substitua a original, causando uma falha na verificação do MIC do AP pelo cliente quando ele receber a mensagem 3. A estação cliente ficaria impedida de se associar ao AP.

Outros ataques DoS possíveis

Um atacante pode enviar frames de desassociação para forçar o cliente a se reassociar ao AP. Uma mensagem *EAP-Failure* pode colocar o cliente em estado de *HELD* (preso). Também é possível encher o AP com o máximo de associações permitidas usando endereços MAC aleatórios.

3.5.5 Ataques ao Padrão IEEE 802.1X

Atacando o 802.1x

Originalmente, o protocolo 802.1x foi desenvolvido para ser usado em redes cabeadas, onde todos os equipamentos de rede estão trancados em um *rack* dentro de um centro de processamento de dados ou sala de telecomunicações. Nessas redes, um conector na parede é considerado seguro, pois está ligado fisicamente a apenas um *switch*. O usuário necessita de um acesso físico às instalações da empresa para poder se plugar ao conector e obter o acesso à rede. Na rede local sem fios, os pontos de acesso podem ser espalhados pela empresa para facilitar a mobilidade dos usuários. Qualquer pessoa interessada em se conectar à rede pode enviar sinais e solicitar o acesso. Um atacante pode instalar pontos de acesso não autorizados. Quando o protocolo 802.1x foi criado, não foi levada em consideração a necessidade de validar o ponto de acesso, mas na rede sem fios isso se torna necessário.

A falta de um mecanismo de validação do ponto de acesso permite vários ataques do tipo homem-do-meio. O objetivo do atacante é instalar um ponto de acesso não autorizado e capturar as credenciais de algum usuário legítimo da rede. Usando o protocolo 802.1x, ao se conectar, o cliente deve digitar as suas credenciais para obter acesso à rede. O *Windows XP* no *Service Pack 2*, por exemplo,

é dotado da habilidade de conectar-se sozinho a rede sem fios assim que o *notebook* é ligado, o que pode facilitar a ação do *hacker* que ataca a rede.

O exemplo apresentado aqui pode ser executado para comprometer uma rede sem fios que usa o MS PEAP 802.1x. O atacante precisa primeiro encontrar a rede que usa o MS PEAP 802.1x, o que não deve ser muito difícil, pois o custo de implementar tal rede não é grande para quem já utiliza o *Windows*. O atacante pode executar um *wardriving* pelos edifícios a procura de sinais de redes locais sem fios. Ele deve capturar os dados e procurar por alguma SSID e verificar que a rede usa o MS PEAP com 802.1x. O atacante pode usar uma antena direcional de longo alcance, ou pode, equipado de um *notebook*, aproximar-se do local para capturar os dados no ar. Pode acontecer de encontrar alguma implementação em que os quadros 802.1x estejam cifrados. Neste caso o atacante deverá esperar por quadros de *probe request* e *probe responses* onde irá trafegar uma SSID em aberto.

Tendo em mãos o SSID, o atacante deve configurar um servidor e um ponto de acesso não autorizado. Um *laptop* pode ser usado para tal tarefa. Serão necessários os serviços de DHCP (*Dynamic Host Configuration Protocol*), DNS (*Domain Name System*) e servidor Web. O atacante configura o ponto de acesso para usar a mesma SSID capturada, e não incluir nenhum mecanismo de segurança. Agora o atacante deve instalar seu ponto de acesso no local e conectar uma antena de longo alcance. Algumas pessoas irão fatalmente se conectar ao seu falso ponto de acesso. Ao se conectar, uma mensagem irá aparecer para o usuário, informando que a rede não é segura. Alguns usuário podem desistir de continuar. Mas algum usuário, ao identificar o SSID como sendo válido, simplesmente optará por continuar.

Depois de configurar o IP, *gateway* padrão e DNS através do servidor DHCP, o que no *Windows* é feito de forma automática, o usuário irá tentar conectar a algum serviço da rede, tal como *e-mail* ou acessar a Web. Ao tentar acessar a Web, um aplicativo em Java Script irá abrir uma janela semelhante à janela padrão de autenticação do 802.1x, anteriormente preparada pelo atacante, na qual serão solicitadas as credenciais do usuário. Ao clicar no botão Ok, as credenciais são enviadas para o atacante que desligará o ponto de acesso. O usuário será então direcionado para o ponto de acesso original e será reautenticado. O usuário tentará novamente e não pensará em chamar o *helpdesk* para um problema já resolvido. A única forma de detectar o ataque é usando um software de detecção de pontos de acesso não autorizados. É possível evitar este ataque pelo uso de EAP-TLS, mas a maioria das empresas são relutantes em implantar PKI (*Public Key Infrastructures*) [Earle 2006].

Atacando o LEAP da Cisco

O LEAP (*Lightweight Extensible Authentication Protocol*) foi desenvolvido pela empresa Cisco para permitir um *roaming* rápido de célula para célula. Mas foram descobertas tantas vulnerabilidades que a Cisco o substituiu por um novo EAP chamado EAP-FAST. Dentre as ferramentas existentes para atacar o LEAP, existem a *anwap* e a *asleap*. A ferramenta *anwap* tenta um grande número de autenticações no ponto de acesso. Já a ferramenta *asleap* faz o mesmo que a anterior, mas acrescenta um ataque *off-line* nos quadros LEAP, e pode recuperar uma chave fraca de LEAP com um ataque de dicionário.

Atacando o RADIUS

O segredo compartilhado (*shared secret*), usado para comunicação entre um servidor RADIUS e os outros dispositivos RADIUS ou clientes, é criado através de da função *hash* MD5. São usados como entrada do MD5 um código, um ID, o tamanho, o autenticador de requisições e os atributos do autenticador de resposta. Como o MD5 é uma função unidirecional, ela não pode ser quebrada, podendo-se usar uma ferramenta de força bruta para tentar todas as possíveis chaves. Um atacante pode capturar o processo de autenticação e executar uma ferramenta de força-bruta para conseguir a chave. Como os quadros de autenticação não são autenticados pelo RADIUS, qualquer um pode iniciar o processo de autenticação e capturar os quadros necessários. Servidores RADIUS que seguem a RFC 2869 autenticam as estas mensagens, o que impede que o servidor entregue o segredo compartilhado a qualquer um que o solicitar.

É comum usar o RADIUS para algum tipo de acesso remoto. Nessa situação, se o acesso remoto tem vulnerabilidades, o atacante pode usar destas vulnerabilidades para comprometer o segredo compartilhado.

Outra questão é se o administrador do servidor RADIUS tem as habilidades necessárias para corretamente configurar a segurança do próprio servidor RADIUS [Earle 2006].

Ataque DoS ao EAP

Um atacante pode enviar repetidamente quadros de EAP *Stat* para um ponto de acesso, se ele não puder processar todos estes quadros, talvez ele reinicie ou se torne inoperante. Um outro ataque é enviar quadros EAP mal-formados, o que pode em alguns casos tirar do ar um ponto de acesso ou um servidor RADIUS. Isso já foi provado no Free RADIUS, quando são enviados quadros EAP-

TLS com alguns bits alterados. Uma outra opção seria encher o ponto de acesso com um grande número de conexões, já que o EAP permite 256 IDs *tags* de clientes [Earle 2006].

3.5.6 Outros Ataques às Redes Locais sem Fios

Atacando a Filtragem MAC

A filtragem MAC foi adotada por muitos profissionais de TI como sendo um remédio para as várias vulnerabilidades do WEP publicadas. O problema é que o endereço MAC pode facilmente ser alterado. Algumas versões de sistemas operacionais permitem que se faça a mudança do endereço MAC. Se algum sistema não proporcionar isso, ainda é possível usar uma ferramenta disponível na Internet. O atacante deve capturar um endereço MAC no ar (ele é transmitido em aberto), alterar o seu próprio MAC e então assumir o lugar de um cliente legítimo. Ele pode por exemplo enviar um quadro de desautenticação e em seguida se autenticar [Earle 2006].

Atacando os *Gateways* da Rede sem Fios

Para suportar uma grande variedade de clientes, os *gateways* de redes sem fios usam muitos meios de autenticação. Em contrapartida, uma outra quantidade de soluções de mercado não são acomodadas. Isso deixa brechas para ataques sobre os meios de autenticação fracos.

Para suportar clientes leves (com pouco poder de processamento), freqüentemente a opção é usar o SSL (*Secure Sockets Layer*). Quando o SSL é usado o *gateway* apresenta para o usuário uma página de Web padrão para autenticação. Uma ferramenta de *proxy* com SSL pode ser usada para executar o ataque. O primeiro passo é configurar um servidor *proxy* e colocá-lo entre o *gateway* e o cliente. Quando o cliente solicitar uma conexão, ele irá conectar no *proxy*. O cliente e o *proxy* estabeleceram uma conexão SSL. O *proxy* solicita uma conexão ao *gateway* e estabelecem uma conexão SSL. O cliente então autentica o *gateway*. O *proxy* irá receber o tráfego de autenticação criptografado, decriptografar, criptografar novamente e por fim enviar ao *gateway*. A ferramenta Achilles pode executar este ataque [Earle 2006].

Um ataque ao *gateway* pode também ser executado com um ponto de acesso não-autorizado, como já foi explicado antes nesta seção.

Atacando o Telnet

O Telnet é usado para executar sessões de terminal remoto nos dispositivos. Ele permite ao administrador configurar os dispositivos de rede através de do seu IP. Esse protocolo foi criado nos anos 60, e muitas características de segurança foram omitidas. A maior fraqueza está na execução da autenticação usando tráfego em aberto. O atacante pode capturar os quadros e ver em aberto o usuário e a senha usadas na autenticação [Earle 2006].

Atacando o HTTP

O meio mais fácil de gerenciar o ponto de acesso é através de um navegador Web. Mas páginas HTTP são vulneráveis a um ataque automatizado para adivinhar senhas. Isso pode ser executado tentando palavras em um dicionário ou usando "força bruta". Por estas possibilidades, algumas empresas têm desabilitado as interfaces baseadas na Web na administração de seus equipamentos de rede [Earle 2006].

Atacando o SNMP

O *Simple Network Management Protocol* (SNMP) pode ser uma ferramenta valiosa para resolver problemas e modelar a rede. Mas na mão de um atacante pode lhe dar poderes para mudar a configuração de diversos equipamentos da rede. O SNMP tem muitos mecanismos básicos de segurança, esta é baseada em duas senhas, uma para leitura e uma para escrita. De posse da senha de leitura, o atacante pode ter acesso a atual configuração dos dispositivos, sem no entanto poder alterá-la. Já a senha de escrita, confere-lhe o poder de reiniciar um dispositivo, alterar a configuração ou até mesmo alterar as senhas. Existem algumas ferramentas para executar ataques ao SNMP que podem tentar um ataque de dicionário ou "força bruta". O SNMP tem três versões. A versão 3 tem aprovação do *Internet Engineering Steering Group* (IESG), mas poucos equipamentos a suportam. Muitos equipamentos suportam a versão 1 e alguns já suportam a versão 2. Nestas duas faltam algumas características necessárias à segurança [Earle 2006].

3.6 Considerações Finais sobre a Segurança das Redes Locais sem Fios

Este capítulo apresentou os padrões de encriptação e integridade usados no Padrão IEEE 802.11, suas propriedades, vulnerabilidades e ataques conhecidos. Esse padrão será usado como objeto de estudo na análise do Capítulo 4, onde serão referenciadas as características, pontos fortes e vulnerabilidades apresentadas aqui. O objetivo é comparar a análise feita pela aplicação do Modelo de Função Vantagem com a segurança percebida pelos usuários.

Capítulo 4

Aplicação do Modelo às Redes Locais sem Fios

Conforme análise apresentada no capítulo 2, é mais fácil provar insegurança do que provar segurança de um esquema. Isto ocorre porque é possível identificar ações que implicam que o esquema é inseguro. Os modelos apresentados no capítulo 2, por outro lado, não estão vinculados às limitações da implementação ou da praticidade e, por isto, são bem eficazes e poderosos nas demonstrações de segurança.

A função vantagem modela de forma apropriada os esquemas de segurança. No caso de PRFs e PRPs, o problema envolve descobrir discernidores poderosos o bastante para "quebrar" o esquema, conseguindo resolver o problema de dizer se uma seqüência é verdadeiramente randômica ou se foi gerada por uma função ou família de funções (veja a seção 2.1). No caso de esquemas de criptografia, a função vantagem é usada para modelar a segurança e, além disto, podemos relacionar a segurança do esquema com a segurança do cifrador de blocos utilizado e então verificar se o esquema insere fraquezas ou não (veja a seção 2.2).

Este capítulo usa a abordagem da função vantagem, apresentada no capítulo 2, para modelar os protocolos de segurança das redes sem fio apresentadas no capítulo 3. Usando esta abordagem, a seção 4.1 mostra que o algoritmo RC4, utilizado como uma PRF, tem discernidores fortes o bastante para quebrá-lo, além disso, o esquema utilizado no protocolo WEP é ruim. A seção 4.2 mostra que apesar de o esquema do protocolo WPA ser mais forte que o esquema do protocolo WEP, ele ainda sofre de vulnerabilidades por usar o algoritmo RC4, mesma PRF usada no protocolo WEP. Na seção 4.3, é apresentada uma análise do protocolo WPA2, cuja função vantagem mostra características de

segurança que o faz ser considerado o mais seguro e recomendado.

4.1 A Função Vantagem no Protocolo WEP

Como já foi dito, o WEP usa uma implementação do algoritmo RC4 como gerador da chave de fluxo. Segue então a análise do RC4 e depois a análise do esquema do WEP.

4.1.1 A Segurança do RC4 como uma PRP

O algoritmo RC4, cujo funcionamento foi apresentado na seção 3.1.1, é o software cifrador de *streams* mais utilizado nos dias de hoje. Ele já foi integrado a várias aplicações como TLS/SSL e o WEP. O RC4 foi desenvolvido por Ron Rivest, em 1987, e foi mantido em segredo até 1994, quando seu algoritmo foi descoberto e publicado [Paul and Preneel 2004]. Entre outras aplicações que receberam implementações do RC4 estão, Microsoft Windows, Lotus Notes, Apple AOCE, Oracle Secure SQL [Mantin and Shamir 2001].

O RC4 vem sendo examinado de forma minuciosa desde que foi publicado. Paul e Preneel apresentaram, em 2004, uma relação de ataques ao RC4 [Paul and Preneel 2004]. Aqui será utilizada a modelagem da função vantagem mostrada no capítulo 2. Como já foi mostrado no capítulo 2, para modelar uma função pseudo-randômica, ou uma permutação pseudo-randômica usando a função vantagem, é necessário apresentar discernidores. Os discernidores são algoritmos que conseguem dizer se uma determinada *stream* é verdadeiramente randômica ou se é resultado de uma função pseudo-randômica (veja o capítulo 2).

O melhor discernidor anteriormente mencionado na literatura, distingue o RC4 de fontes randômicas analisando $2^{30.6}$ palavras da saída [Mantin and Shamir 2001]. Este discernidor é chamado de **fraco**, pois ele analisa uma *stream* simples. Os novos discernidores são chamados **fortes**, porque analisam várias *streams*. Aqui serão apresentados dois discernidores fortes para o RC4. O primeiro executa sua função de discernir, mas só pode ser usando em aplicações de *broadcast*, como será explicado na seção 4.1.2. O segundo tem a característica de se aproveitar de uma tendência na distribuição dos bytes da saída do RC4 e é apresentado na seção 4.1.4.

4.1.2 Discernidor do RC4 segundo Mantin e Shamir

Mantin e Shamir, em 2001, descreveram um discernidor que obtém uma vantagem considerável em distinguir entre o RC4 e uma função randômica [Mantin and Shamir 2001]. Eles observaram que a segunda palavra da saída do RC4 tinha uma tendência em ser igual ao valor 0 (zero) com uma probabilidade de duas vezes a esperada ($1/128$ ao invés de $1/256$ para $n = 8$). Segundo Mantin e Shamir, esta tendência não havia sido observada antes porque os discernidores anteriores analisavam apenas uma única *stream* e para perceber tal tendência é necessário observar várias *streams*.

Teorema 4. 33. *Assuma que a permutação inicial S no algoritmo PRGA() é uma escolha randômica do conjunto de todas as permutações possíveis de $0, ..., N-1$, isto é, do conjunto de todos os resultados possíveis do KSA(). Então a probabilidade de a segunda palavra da saída do RC4 ser igual a 0 é aproximadamente $2/N$.*

Prova do Teorema 4.33: Considerando o algoritmo PRGA(), seja S_t a permutação S, após sua atualização na rodada t (sendo S_0 a primeira permutação) e z_t a saída desta rodada. Primeiro será mostrado que se $S_0[2] = 0$ e $S_0[1] \neq 2$, então a segunda saída é 0 com probalidade de 1. A Figura 4.1 mostra os dois primeiros *rounds* das permutações. A saída no segundo *round* é $S = [X + 0]$, que é o valor da posição X, posição esta que acabou de receber o zero que estava de início na posição 2. Observe que o valor de Y não importa e o de X deve ser diferente de 2.

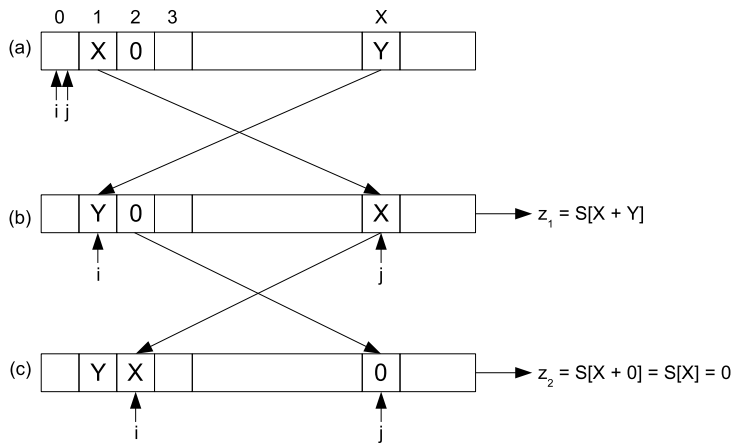

Figura 4.1: Os dois primeiros *rounds* do RC4 com $S_0[2] = 0$ e $S_0[1] \neq 2$. (a) posição inicial da geração das saídas, (b) a primeira saída e (c) a segunda saída.

A probabilidade de $S_0[2] = 0$ é igual a $1/N$ e de $S_0[2] \neq 0$ é igual a $1 - 1/N$. Então:

$$\begin{aligned} P[z_2 = 0] &= P[z_2 = 0|S_0[2] = 0] \cdot P[S_0[2] = 0] + P[z_2 = 0|S_0[2] \neq 0] \cdot P[S_0[2] \neq 0] \\ &\approx 1 \cdot 1/N + 1/N \cdot (1 - 1/N) \\ &\approx 1/N \cdot (1 + 1 - 1/N) \\ &\approx 2/N \end{aligned}$$

Sendo duas vezes a probabilidade esperada.

Outro resultado que é obtido aplicando a regra de Bayes, é:

$$\begin{aligned} P[S_0[2] = 0|z_2 = 0] &= \frac{P[S_0[2] = 0]}{P[z_2 = 0]} \cdot P[z_2 = 0|S_0[2] = 0] \\ &\approx \frac{1/N}{2/N} \cdot 1 = \frac{1}{2} \end{aligned}$$

Distinguindo RC4 de Fontes Randômicas

O discernidor apresentado por Mantin e Shamir é **forte**, porque ele analisa várias *streams*. Ele requer apenas $O(N)$ segundas palavras de *streams* com chaves distintas para distinguir o RC4 de uma fonte verdadeiramente randômica, como pode ser visto no Teorema 4.34.

Teorema 4. 34. *Sejam X, Y distribuições, e suponha que o evento e acontece em X com uma probabilidade de p e em Y com uma probabilidade de $p(1 + q)$. Então, para p e q pequenos, $O\left(\frac{1}{pq^2}\right)$ exemplos são suficientes para distinguir X de Y como uma probabilidade constante de sucesso.*

Prova do Teorema 4.34: Sejam X_e, Y_e variáveis randômicas que especificam o número de ocorrências de e em t exemplos. Então X_e e Y_e têm uma distribuição binomial com parâmetros (t, q) e $(t, p(1_+q))$, e seus valores esperados, variâncias e desvios padrões são [Mantin and Shamir 2001]:

$$\begin{aligned} E[X_e] = tp \quad &, \quad E[Y_e] = tp(1 + q) \\ V[X_e] = tp(1 - p) \approx tp \quad &, \quad V[Y_e] = tp(1 + q)(1 - p(1 + q)) \approx tp(1 + q) \\ \sigma(X_e) = \sqrt{V(X_e)} \approx \sqrt{tp} \quad &, \quad \sigma(Y_e) = \sqrt{V(Y_e)} \approx \sqrt{tp(1 + q)} \approx \sqrt{tp} \end{aligned}$$

Conforme a teoria de probabilidades temos que se $E[Y_e] - E[X_e] \geq \sigma(X_e)$ então é possível distinguir

as distribuições X_e e Y_e. Mas,

$$\begin{aligned} E[Y_e] - E[X_e] &\geq \sigma(X_e) \\ &\Updownarrow \\ tp(1+q) - tp &\geq \sqrt{tp} \\ &\Updownarrow \\ tpq &\geq \sqrt{tp} \\ &\Updownarrow \\ t &\geq \frac{1}{pq^2} \end{aligned}$$

Assim, $O(\frac{1}{pq^2})$ exemplos são suficientes para o discernidor (a constante depende da probabilidade desejada do sucesso) [Mantin and Shamir 2001].

Considere X a distribuição probabilística da segunda saída em *streams* uniformemente distribuídas, e Y a distribuição probabilística da segunda saída de streams produzida pelo RC4 para chaves escolhidas randomicamente. O evento e denota uma saída de valor 0 no segundo byte, a qual acontece com probabilidade de $1/N$ em X e $2/N$ em Y. Usando o Teorema 4.34 com $p = 1/N$ e $q = 1$, podemos concluir que precisamos de aproximadamente $\frac{1}{pq^2} = N$ saídas para seguramente distinguir entre as duas distribuições [Mantin and Shamir 2001].

Um Ataque *Ciphertext-Only* em uma Aplicação de *broadcast* com RC4

Uma aplicação de *broadcast* com o RC4 é algo análogo ao problema clássico dos Generais Bizantinos, onde a mesma ordem de "atacar" ou "recuar" deve ser enviada a cada general em *broadcast*. Cada cópia é cifrada, com chaves diferentes, usando RC4. Um inimigo que captura todos os *ciphertexts* pode facilmente deduzir qual das duas possíveis mensagens foi enviada, independente do tamanho das chaves.

Teorema 4. 35. *Seja M uma mensagem e $C_1, C_2, ..., C_k$ as cifragens de M usando o RC4, com k chaves uniformemente distribuídas. Então se $k = \Omega(N)$, o segundo byte de M pode ser seguramente extraído de $C_1, C_2, ..., C_k$ [Mantin and Shamir 2001].*

Prova do Teorema 4.35: Para cada chave de criptografia, $M[2]$ tem probabilidade $\frac{2}{N}$ de ser XORed com 0, e probabilidade $\frac{1}{N}$ de ser XORed com outro byte qualquer. Assim, uma fração de $\frac{2}{N}$

bytes da segunda posição terão o mesmo valor nos textos cifrados, e assim o caracter mais freqüente em $C_1[2], ..., C_k[2]$ é igual ao caracter original $M[2]$.

Muitos protocolos de *broadcast* são utilizados hoje, como por exemplo usuários que enviam o mesmo e-mail para múltiplos destinatários cifrados com diferentes chaves. Em muitas aplicações de *groupware*, que permitem aos usuários sincronizar seus documentos por *broadcasting*, listas de modificações cifradas, são enviadas para todos os membros do grupo. Todas estas aplicações são vulneráveis a este ataque [Mantin and Shamir 2001].

4.1.3 A Vantagem do Discernidor de Mantin e Shamir para o Algoritmo RC4

Esta seção propõe uma análise da vantagem do discernidor de Mantin e Shamir para o algoritmo RC4. Esta análise faz parte das contribuições deste trabalho. As aplicações mais comuns do algoritmo RC4 na criptografia, são implementações que combinam as saídas com a mensagem através de operações OR-exclusivo (XOR), obtendo como resultado o texto cifrado. Tal operação é considerada uma permutação, pois o resultado da operação XOR em um byte, resulta um outro byte do mesmo domínio.

A análise que se segue foi idealizada com o objetivo de aplicar o modelo de função vantagem ao algoritmo RC4. Este algoritmo é uma função pseudo-randômica. Como indicado no esquema da Figura 4.2, o algoritmo RC4 é encapsulado, ficando no interior do esquema apresentado. Neste caso, apesar do algoritmo RC4 ser uma função pseudo-randômica, o esquema da Figura 4.2 é uma permutação pseudo-randômica. Utilizando o discernidor de Mantin e Shamir, demonstramos a seguir que o uso do algoritmo RC4, como uma função pseudo-randômica, é inapropriado.

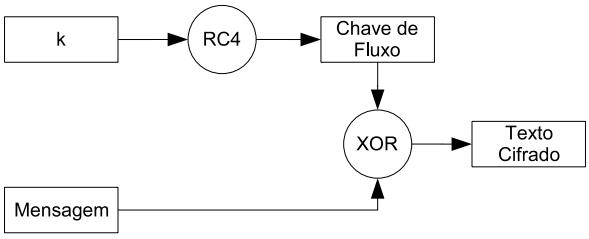

Figura 4.2: Esquema criptográfico de exemplo de uso do RC4.

Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ uma família de permutações, que utiliza a saída do algoritmo RC4, como no esquema da Figura 4.2. Neste esquema k (semente) é uma escolha randômica que define qual das permutações possíveis será usada. Na Figura 4.2, a saída do RC4 é combinada com

a mensagem através de operações do tipo XOR. Por outro lado, seja $Perm$ uma função verdadeiramente randômica, como foi definido na seção 2.1.2. Assuma então, a existência de duas funções g^0 e g^1 como segue:

Function $g^1(x)$	Function $g^0(x)$
$y \xleftarrow{R} F(x)$	$y \xleftarrow{R} Perm(x)$
Return y	Return y

Onde $\xleftarrow{R}$ indica uma escolha randômica a cada chamada de g^1 ou g^0. No caso de F, significa um k randômico a cada chamada. A função g^1 recebe um valor x (mensagem), usa uma função da família F, escolhida randomicamente através de k, para permutar o valor x por y e retorna o valor. Já a função g^0 recebe o valor x, usa uma função verdadeiramente randômica para permutá-lo por y e retorna o valor.

Considere os experimentos:

Experiment $\mathbf{Exp}_{F,A}^{prp-1}$	Experiment $\mathbf{Exp}_{F,A}^{prp-0}$
$d \leftarrow A^{g^1}$	$d \leftarrow A^{g^0}$
Return d	Return d

onde A é o adversário como definido na seção 2.1.3. Ele recebe um acesso a um oráculo (g^1 ou g^0). A não sabe qual oráculo recebeu, isto ele deve adivinhar enviando mensagens e observando as saídas geradas pelo oráculo.

A prp-advantage de A é dada por

$$\mathbf{Adv}_{F,A}^{prp} = \mathbf{P}\left[\mathbf{Exp}_{F,A}^{prp-1} = 1\right] - \mathbf{P}\left[\mathbf{Exp}_{F,A}^{prp-0} = 1\right]$$

que significa a probabilidade de A, recebendo o oráculo g^1, responder que está no mundo 1, menos a probabilidade de, recebendo o oráculo g^0, responder que que está no mundo 1. Para qualquer t, q, μ, conforme definido na seção 2.1.3, a prp-advantage de F é

$$\mathbf{Adv}_{F}^{prp}(t, q, \mu) =_{A}^{max} \left\{\mathbf{Adv}_{F,A}^{prp}\right\}$$

onde o máximo de todos os As que têm complexidade de tempo t e fazem no máximo q consultas ao oráculo e a soma dos tamanhos das consultas seja μ bits.

Um adversário é definido aqui e trabalha da seguinte forma:

Adversário A^g

Faça $X \leftarrow 1^{2l}$

Faça $C \leftarrow 0$

Para i de 0 até 255

Faça $Y_1Y_2 \leftarrow g(X)$ // onde $Y = Y_1Y_2$

Se $Y_2 = 1^l$ então $C \leftarrow C + 1$

Se $C \geq 2$ então retorne 1 senão retorne 0

O algoritmo começa atribuindo uma *string* de *bits* 1^{2l} a X. Esta é a mensagem que será cifrada. Ela tem o tamanho de duas palavras porque a tendência apresentada ocorre justamente na segunda palavra e todos os *bits* são fixados no valor 1. O tamanho da palavra é de l *bits*, no caso do protocolo WEP l é igual a 8. Um contador C recebe zero, ele conta quantas vezes o RC4 gera zero na segunda palavra. A variável i é apenas para limitar o laço em 256 vezes, pois $N = 256$ é o número de segundos *bytes* suficientes para discernir entre o mundo 0 e mundo 1, de acordo com o Teorema 4.34. O laço chama o oráculo g para a entrada X e separa o resultado em dois *bytes*. A partir da análise deste resultado, podemos ter informação sobre a entrada. Caso o segundo byte seja uma *string* de *bits* igual a 1^l, então mais uma vez o oráculo fez XOR da mensagem com zero. Neste caso, o segundo *byte* da mensagem foi retornado. Além disso, a variável C é incrementada. No final das iterações, o valor de C é verificado. Se C é maior ou igual a 2, significa que o oráculo fez XOR com zero no segundo *byte* pelo menos duas vezes. Veja as regras da operação XOR na Tabela **??**.

Numa função verdadeiramente randômica, a chance de sair um zero no segundo byte de uma única escolha é de $1/N$, que neste caso é $1/256$. Já a chance de não sair zero é de $255/256$. Em duas escolhas independentes a chance de não sair zero é o produto das chance de cada escolha, isto é, $(255/256) \times (255/256)$. Assim, em i escolhas independentes, a chance de não sair zero é de $(255/256)^i$ e a chance de sair zero é de $1 - (255/256)^i$. Tendo então saído um zero em uma das escolhas, a chance de sair um zero em uma outra escolha é de $1 - (255/256)^{i-1}$. E podemos concluir que a chance de sair dois zeros em i escolhas independentes é de $[1 - (255/256)^i] \times [1 - (255/256)^{i-1}]$.

Nosso adversário A tenta 256 escolhas, então $i = 256$. Assim,

$$\begin{aligned}
[1-(255/256)^i] \times [1-(255/256)^{i-1}] &= [1-(255/256)^{256}] \times [1-(255/256)^{255}] \\
&= 0,632840 \times 0,631400 \\
&= 0,399575176 \\
&\approx 0,40.
\end{aligned}$$

Segundo o Teorema 4.33 a chance de sair zero no segundo byte de uma única saída do RC4 é de $2/N$, isto é duas vezes a de uma função verdadeiramente randômica. Seguindo o raciocínio anterior, a chance é de $2/256$. Já a chance de não sair zero é de $254/256$. Em duas escolha independentes a chance de não sair zero é o produto das chance de cada escolha, isto é, $(254/256) \times (254/256)$. Assim em i escolhas independentes a chance de não sair zero é de $(254/256)^i$ e a chance de sair zero é de $1-(254/256)^i$. Tendo então saído um zero em uma das escolhas, a chance de sair um zero em uma outra escolha é de $1-(254/256)^{i-1}$. E podemos concluir que a chance de sair dois zeros em i escolhas independentes é de $[1-(254/256)^i] \times [1-(254/256)^{i-1}]$.

Nosso adversário A tenta 256 escolhas, então $i = 256$. Assim,

$$\begin{aligned}
[1-(254/256)^i] \times [1-(254/256)^{i-1}] &= [1-(254/256)^{256}] \times [1-(254/256)^{255}] \\
&= 0,865723 \times 0,864666 \\
&= 0,748561 \\
&\approx 0,75
\end{aligned}$$

Então é possível afirmar que

$$\mathbf{P}\left[\mathbf{Exp}_{F,A}^{prf-1} = 1\right] = 0,75$$

$$\mathbf{P}\left[\mathbf{Exp}_{F,A}^{prf-0} = 1\right] = 0,40.$$

Na primeira equação acima, temos a probabilidade de o adversário A conseguir mais de uma saída igual a zero no segundo *byte* e retornar 1. Neste caso, o adversário recebe o oráculo g^1 que faz acesso ao algoritmo RC4. Na segunda equação, também temos a probabilidade de o adversário A obter mais de uma saída igual a zero no segundo *byte* e retornar 1. Mas neste caso, o adversário

recebeu o oráculo g^0, que faz acesso a uma função perfeitamente randômica.

Então, a vantagem do adversário A ao atacar o algoritmo RC4 é:

$$\mathbf{Adv}_{F,A}^{prf-RC4} = 0,75 - 0,40 = 0,35$$

Considere t a complexidade de tempo do algoritmo do adversário A. O algoritmo A^g possui um laço com N passos, ou seja, ele é executado N vezes. O custo da execução de cada passo deste laço é o custo de execução da função g(x), que é igual a $O(N)$, quando estamos utilizando o algoritmo RC4. Portanto, o custo de A é igual a $N \cdot O(N)$. Isto é, $t = O(N^2)$.

Se número de consultas feitas por A é somente 256, então a soma dos tamanhos das consultas é de $512l$. Assim,

$$\begin{aligned} \mathbf{Adv}_{F}^{prp-RC4}(t, 256, 512l) &= \max_{A} \left\{ \mathbf{Adv}_{F,A}^{prp-RC4} \right\} \\ &\geq \mathbf{Adv}_{F,A}^{prp-RC4} \\ &\geq 0,35. \end{aligned}$$

Neste caso, o adversário consegue esta vantagem com poucos recursos. Isto torna o RC4 inseguro como uma permutação pseudo-randômica. Apesar de o ataque apresentado ser possível apenas em aplicações de *broadcast* isto mostra que o algoritmo RC4 tem graves falhas de segurança.

4.1.4 Discernidor do RC4 segundo Paul e Preneel

Um outro discernidor para o RC4 foi apresentado em 2004 por Paul e Prennel. Eles identificaram através da observação de várias *streams* de saída do RC4, que a distribuição dos dois primeiros bytes não é uniforme. Foi observado que a probabilidade dos dois primeiros bytes da saída do RC4 serem iguais é de $\frac{1}{N}(1 - \frac{1}{N})$, enquanto em uma distribuição perfeitamente randômica a probabilidade deste evento é de $\frac{1}{N}$.

Seja $S_t[l]$ o l-ésimo elemento do S-box após a troca (*swapping*) do *round* t do RC4, e z_t o byte de saída gerado neste *round*. Sejam também, i_t e j_t os valores das variáveis de índices do RC4 no *round* t (varáveis i e j do algoritmo RC4 na Seção 3.1.1 representando posições no *array* S).

Teorema 4. 36. *Se $S_0[1] = 2$, então os dois primeiros bytes da saída do RC4 são sempre diferentes [Paul and Preneel 2004].*

Prova do Teorema 4.36: A Figura 4.3 apresenta a execução dos dois primeros *rounds* da geração das saídas do RC4. Note que, $z_1 = S_1[X + 2]$ e $z_2 = S_2[Y + 2]$. Observe também que $X + 2$ e

$Y + 2$ apontam para duas posições diferentes no *array*, porque $X \neq Y$. Assim, as duas primeiras saídas só podem ser iguais se $X = 0$ e $Y = 2$ ou se $X = 2$ e $Y = 0$, mas isto é impossível porque $X \neq Y \neq 2$, pois o *array* contém permutações de elementos e não tem elementos iguais.

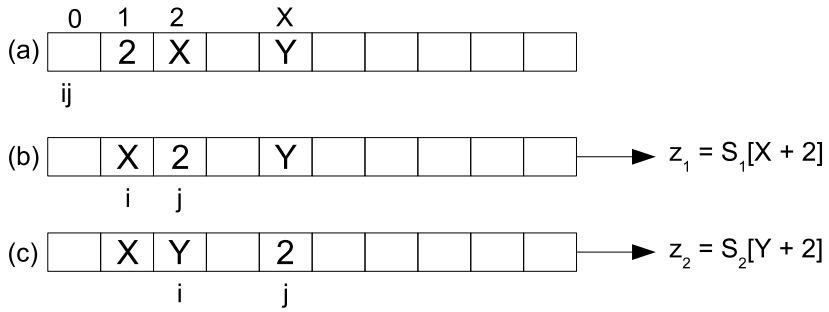

Figura 4.3: Os dois primeiros *rounds* do RC4 com $S_0[1] = 2$. (a) Posição inicial da geração das saídas. (b) A primeira saída. (c) A segunda saída.

Observe agora as conclusões extraídas do Teorema 4.36 através dos corolários.

Corolário 4. 37. *Se os dois primeiros bytes da saída do RC4 são iguais, então* $S_0[1] \neq 2$.

Segundo Paul e Preneel [Paul and Preneel 2004], este fato pode ser usado para melhorar um algoritmo de força bruta em um fator de $\frac{N}{N-1}$.

Corolário 4. 38. *A probabilidade de os dois primeiros bytes serem idênticos é igual a* $(1-1/N)/N$ *(assumindo que* $S_0[1] = 2$ *com probabilidade de* $1/N$ *e que para o restante das permutações, nas quais* $S_0[1] \neq 2$*, os dois primeiros bytes da saída do RC4 são iguais com probabilidade de* $1/N$*) [Paul and Preneel 2004].*

Prova do Corolário 4.38: Se $S_0[1] = 2$ ocorre com probabilidade de $1/N$, então os dois primeiros bytes da saída são diferentes para qualquer chave, como visto no Teorema 4.36, para este primeiro caso. Considerando que cada uma das outras possibilidades de $S_0[1]$ ($S_0[1] \neq 2$) seja $1/N$, então a probabilidade de se ter $S_0[1] \neq 2$ é igual a $(1 - 1/N)$. Além disso, neste caso, os dois primeiros bytes da saída são iguais com probabilidade $1/N$, onde usamos o corolário 37.

Então podemos deduzir que,

$$\begin{aligned} P[O_1 = O_2] &= P[O_1 = O_2 \mid S_0[1] = 2] \cdot P[S_0[1] = 2] + \\ &\quad P[O_1 = O_2 \mid S_0[1] \neq 2] \cdot P[S_0[1] \neq 2] \\ &= 0 \cdot \frac{1}{N} + \frac{1}{N} \cdot (1 - \frac{1}{N}) \\ &= \frac{1}{N} \cdot (1 - \frac{1}{N}) \\ &= (1 - 1/N)/N. \end{aligned}$$

Distinguindo RC4 de Fontes Randômicas

O discernidor de Paul e Preneel também é **forte** porque ele analisa várias *streams*.

Seguindo o Teorema 4.34, seja X a distribuição dos dois primeiros bytes da saída de uma fonte perfeitamente randômica, Y a distribuição dos dois primeiros bytes da saída do RC4 e e a ocorrência de bytes consecutivos. Assim, $O(N^3)$ é o número de exemplos necessários para distinguir X de Y com uma probabilidade não desprezível ($p = 1/N$ e $q = -1/N$, veja o corolário 4.38). Segundo [Paul and Preneel 2004], observações experimentais atestam os resultados teóricos. Para $N = 256$, com 2^{24} pares dos dois primeiros bytes da saída, geradas com muitas chaves randômicamente escolhidas, tal discernidor alcançou uma vantagem de 40% com valor de divisa em 65408. Isto é, contabilizando-se o número de ocorrências de dois bytes idênticos (os dois primeiros bytes) nas saídas, um número de ocorrências menor que 65408 indica o uso do algoritmo RC4 e um número maior indica o uso de uma função randômica verdadeira.

4.1.5 A Vantagem do Discernidor de Paul e Preneel para o Algoritmo RC4

Propomos nesta seção uma análise da vantagem do discernidor de Paul e Preneel para o algoritmo RC4. A análise que é desenvolvida aqui é mais uma aplicação do modelo de função vantagem em que consideramos o esquema da Figura 4.2.

Seja $F : \{0,1\}^k \times \{0,1\}^l \rightarrow \{0,1\}^l$ uma família de permutações usando a saída do RC4, como por exemplo o esquema da Figura 4.2. Neste esquema k (semente) é uma escolha randômica que define qual das permutações possíveis é usada. Na Figura 4.2, a saída do RC4 é combinada com a mensagem através de operações do tipo XOR. Por outro lado, seja $Perm$ uma função verdadeiramente randômica, como foi definido na seção 2.1.2. Assuma, então, a existência de duas funções g^0

e g^1 como segue:

Function $g^1(x)$	Function $g^0(x)$
$y \xleftarrow{R} F(x)$	$y \xleftarrow{R} Perm(x)$
Return y	Return y

onde $\xleftarrow{R}$ indica uma escolha randômica a cada chamada de g^1 ou g^0. No caso de F, significa um k randômico a cada chamada. A função g^1 recebe um valor x (mensagem), usa um função da família F, escolhida randomicamente através de k, para permutar o valor x por y e retornar o valor. Já a função g^0 recebe o valor x, usa uma função verdadeiramente randômica para permutá-lo por y e retorna o valor.

Considere os experimentos:

Experiment $\mathbf{Exp}_{F,A}^{prp-1}$	Experiment $\mathbf{Exp}_{F,A}^{prp-0}$
$d \leftarrow A^{g^1}$	$d \leftarrow A^{g^0}$
Return d	Return d

onde A é o adversário como definido na seção 2.1.3. Ele recebe um acesso a um oráculo (g^1 ou g^0). A não sabe qual oráculo recebeu, isto ele deve adivinhar enviando mensagens e observando as saídas geradas pelo oráculo.

A prp-advantage de A é dada por

$$\mathbf{Adv}_{F,A}^{prp} = \mathbf{P}\left[\mathbf{Exp}_{F,A}^{prp-1} = 1\right] - \mathbf{P}\left[\mathbf{Exp}_{F,A}^{prp-0} = 1\right]$$

que é igual a probabilidade de A, recebendo o oráculo g^1, responder que está no mundo 1, menos a probabilidade de, recebendo o oráculo g^0, responder que que está no mundo 1. Para qualquer t, q, μ, conforme definido na seção 2.1.3, a prp-advantage de F é

$$\mathbf{Adv}_{F}^{prp}(t, q, \mu) =_{A}^{max} \left\{\mathbf{Adv}_{F,A}^{prp}\right\}$$

onde o máximo de todos os As que têm complexidade de tempo t e fazem no máximo q consultas ao oráculo e a soma dos tamanhos das consultas seja μ bits.

O adversário é definido aqui e trabalha da seguinte forma:

```
Adversário A^g
    Faça X ← 1^{2l}
    Faça C ← 0
    Para i de 0 até 16777215 // N^3 − 1
        Faça Y_1Y_2 ← g(X) // onde Y = Y_1Y_2
        Se Y_1 = Y_2 então C ← C + 1
    Se C < 65408 então retorne 1 senão retorne 0
```

O algoritmo começa atribuindo uma *string* de *bits* 1^{2l} a X. Esta é a mensagem que será cifrada. Ela tem o tamanho de duas palavras, porque a tendência de vantagens (Teorema 4.36) ocorre quando a primeira palavra é idêntica a segunda. Neste caso, todos os *bits* são fixados no valor 1. O tamanho da palavra é de l *bits*, no caso do WEP o valor de l é igual a 8. Um contador C recebe zero, ele conta quantas vezes o RC4 gera os dois primeiros bytes idênticos. A variável i é apenas para limitar o laço em 256^3 vezes, pois $N = 256$ e o valor de N^3 é o número de saídas suficientes para discernir entre o mundo 0 e o mundo 1, de acordo com o Teorema 4.34. O laço chama o oráculo g para a entrada X e separa o resultado em dois *bytes*. A partir da análise deste resultado, podemos obter informação sobre a entrada. Caso o primeiro byte seja idêntico ao segundo, então mais uma vez o oráculo fez XOR dos dois primeiros bytes com um mesmo valor. Além disso, a variável C é incrementada. No final das iterações, o valor de C é verificado. Se C é menor que 65408, significa que o oráculo fez XOR com a mesma chave de fluxo tanto no primeiro como no segundo *byte* menos de 65408 vezes e segundo [Paul and Preneel 2004] esta é a divisa para uma vantagem de 40%.

Então a vantagem do adversário A ao atacar o algoritmo RC4 é

$$\mathbf{Adv}_{F,A}^{prf} = 0,40.$$

Considere t a complexidade de tempo do algoritmo do adversário A. O algoritmo A possui um laço com N^3 passos, ou seja, ele é executado N^3 vezes. O custo da execução de cada passo é o custo de execução da função $g(X)$, que é igual a $O(N)$ quando estamos utilizando o algoritmo RC4. Portanto o custo de A é igual a $N \cdot O(N^3)$. Isto é, $t = O(N^4)$.

Se o número de consultas feitas por A é N^3, então, a soma dos tamanhos das consultas é de $N^3 l$. Assim,

$$\begin{aligned}
\mathbf{Adv}_{F}^{prp}(t, N^3, N^3 l) &= \max_{A} \left\{ \mathbf{Adv}_{F,A}^{prp} \right\} \\
&\geq \mathbf{Adv}_{F,A}^{prp} \\
&\geq 0,40
\end{aligned}$$

Neste caso, os recursos exigidos no discernidor de [Paul and Preneel 2004] são maiores que os exigidos no discernidor de [Mantin and Shamir 2001], mas ele é perfeitamente praticável. Como no anterior, esse ataque também só é possível em aplicações de *broadcast*, porque para perceber a tendência é necessário observar várias streams, sendo uma característica de tais aplicações.

Descartando os N Primeiros Bytes da Saída do RC4

Ao estudar o discernidor de [Mantin and Shamir 2001] é fácil notar que, se a tendência ocorre no segundo *byte* da saída, podemos simplesmente descartar os dois primeiros bytes e o problema está resolvido. O mesmo vale para o discernidor de [Paul and Preneel 2004], exceto que nesse último uma fraqueza similar à observada se repete, não com a mesma intensidade, ao se descartar os N primeiros bytes da saída.

Seja $t \equiv 0 \bmod N$ e $t > 0$. Assuma que $P[S_t[1] = 2 \cap j_t = 0] = 1/N^2$ e que a probabilidade esperada para se ter $O_{t+1} = O_{t+2}$, para o restante dos estados internos, é igual a $1/N$ [Paul and Preneel 2004]. Então,

$$\begin{aligned}
P[O_{t+1} = O_{t+2}] &= P[O_{t+1} = O_{t+2} \mid S_t[1] = 2 \cap j_t = 0] \cdot P[S_t[1] = 2 \cap j_t = 0] \\
&+ P[O_{t+1} = O_{t+2} \mid S_t[1] \neq 2 \cap j_t \neq 0] \cdot P[S_t[1] \neq 2 \cap j_t \neq 0] \\
&= 0 \cdot \frac{1}{N^2} + \frac{1}{N} \cdot (1 - \frac{1}{N^2}) \\
&= \frac{1}{N} \cdot (1 - \frac{1}{N^2}).
\end{aligned}$$

De acordo com o Teorema 4.34, $O(N^5)$ é o número de exemplos necessário para o discernidor, com $p = 1/N$ e $q = -1/N^2$. Segundo [Paul and Preneel 2004], para se obter uma distribuição uniforme, é sugerido que se despreze de $3 \cdot N$ a $12 \cdot N$ bytes da saída. O que pode significar que descartar saídas não é uma solução definitiva e o RC4 deve ser substituído.

4.1.6 Recuperando a Chave do WEP

Fluhrer, Mantin e Shamir descreveram um ataque passivo de *ciphertext-only* (isto é, o atacante vê o texto cifrado, mas ele não tem o poder de solicitar a cifragem ou decifragem de alguma mensagem escolhida por ele para a tarefa de quebrar o sistema). Tal ataque é capaz de recuperar a chave do

WEP, utilizando-se de fraquezas tanto do RC4, quanto do esquema do WEP. Para detalhes sobre as fraquezas explorada neste ataque veja [Fluhrer et al. 2001]. Para detalhes da implementação do ataque e suas otimizações veja [Stubblefield et al. 2002]. Para detalhes de como executar o ataque veja a seção 3.5.2.

Segundo Stubblefield, Ioannidis e Rubin em [Stubblefield et al. 2002], o ataque necessita de aproximadamente 5 milhões de quadros para ser bem sucedido. Entretanto, as várias otimizações sugerida por estes autores, podem reduzir o número de quadros requeridos para algo entre 1 milhão e 2 milhões de quadros. É possível dizer que a vantagem de um adversário em recuperar a chave do WEP é dado por:

$$Adv_{WEP}^{kr}(t, q, \mu) = 1$$

onde t é a complexidade de tempo do adversário, q é o número de consultas ($q < 2^{21}$) e μ é tamanho de todas as consultas somadas ($\mu < q * 2346$ bytes $*8$ bits, estes dois últimos números são o tamanho máximo do quadro MAC segundo [Silva 2005]).

Na seção 2.1.8, através da proposição 2.6, foi mostrado que se existe um kr-adversário B, então é possível construir um prp-cpa-adversário A_B tal que:

$$Adv_{F,B}^{kr} \leq Adv_{F,A_B}^{prp-cpa} + \frac{1}{2^L - q}$$

Veja na equação acima que a vantagem prp-cpa é sempre maior que a vantagem kr. Como o kr-adversário do WEP apresentado tem vantagem igual a 1, então, a vantagem WEP-cpa também é igual a 1. Portanto o WEP é totalmente impróprio para o uso como esquema de confidencialidade.

4.1.7 Avaliando os Modos do WEP

A análise do modo de criptografia utilizado, pode nos dizer se o esquema mantém a vantagem do cifrador subjacente ou se o esquema faz um mau uso do cifrador. Esta análise pode também ajudar a calcular a vantagem de um esquema, baseado na vantagem do cifrador utilizado, como é apresentado na capítulo 2.

O esquema do WEP pode ser visto de dois modos diferentes, como segue.

O Modo ECB com o Protocolo WEP

Na seção 2.2.2, foi apresentado o modo ECB e a seção 2.2.6 mostrou porque ele é considerado inseguro. O esquema do WEP pode ser considerado um modo ECB se mostrarmos que os IVs podem se repetir depois de esgotadas as possibilidades. Nesta situação, a segurança do WEP não depende mais somente da segurança do cifrador. Em outras palavras, o esquema torna-se inseguro mesmo que o cifrador seja muito seguro. A Figura 4.4 mostra como o esquema do WEP se assemelha ao modo ECB.

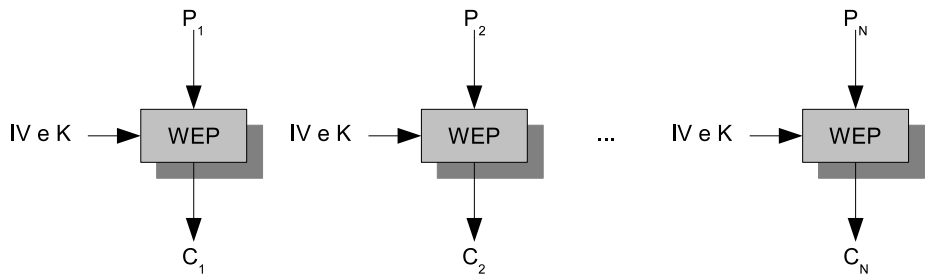

Figura 4.4: O modo ECB com o protocolo WEP

Na Figura 4.4, o bloco de dado P_1 representa um grande bloco formado pelos 2^{24} primeiros pacotes que serão cifrados com a chave K, antes do vetor IV começar a se repetir. O bloco P_2 representa os próximos 2^{24} pacotes com IVs já utilizados no bloco P_1. E os vetores IVs continuam a se repetir sem que a chave seja mudada.

Ao consideramos o uso de todos os vetores IVs possíveis, o WEP trabalha da seguinte maneira:

- O vetor IV é escolhido de forma randômica. A seguir, a cada passo, o vetor IV é incrementado. Esgotadas as possibilidades ele volta ao seu valor inicial.
- Ao executar o reinício podemos pensar em um bloco completo cifrado e o início do próximo bloco.
- Assim temos um bloco grande que é cifrado usando o WEP, tendo como entradas o vetor IV inicial, a chave secreta e o texto plano. E como saída nós temos o texto cifrado.
- A Figura 4.5 ilustra um grande bloco formado de 2^{24} pacotes cifrados com a mesma chave.

A seção 2.2.6 apresenta um adversário que alcança uma vantagem 1 para o modo ECB. Logo, o modo ECB é um esquema de criptografia inseguro, mesmo que o cifrador de blocos utilizado seja seguro. Para os recursos necessários aqui, a Função Vantagem seria

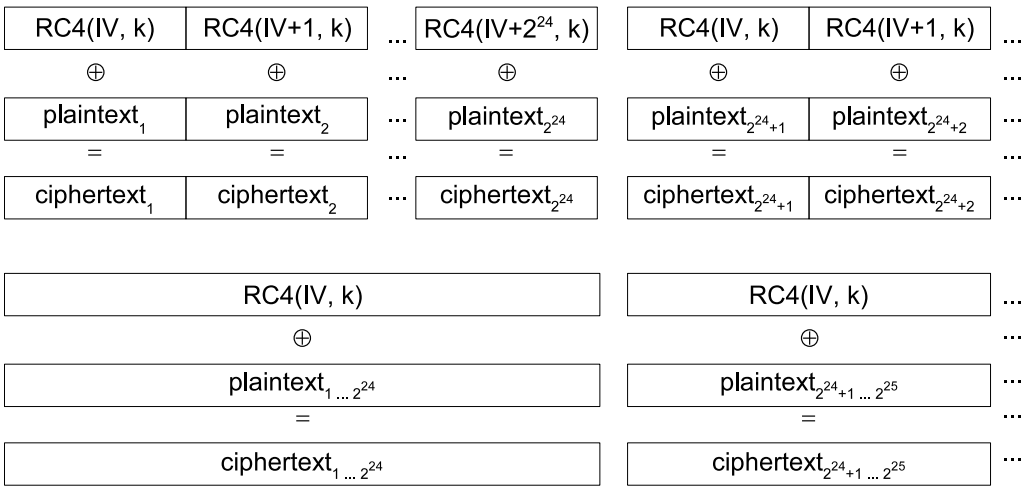

Figura 4.5: O bloco do ECB do protocolo WEP formado por 2^{24} pacotes.

$$\mathbf{Adv}^{ind-cpa}_{\texttt{ECB-WEP}}(t, 2^{24}, 2l \times 2^{24}) = 1$$

E como já foi dito, a fraqueza não está na ferramenta, mas na maneira de utilizá-la. No caso do WEP tanto a ferramenta como o esquema utilizados têm fraquezas.

O Modo C-CTR com o Protocolo WEP

Na seção 2.2.2, foram apresentados vários modos de criptografia, entre eles o C-CTR. O esquema do WEP pode ser bem parecido com o C-CTR, na medida em que é escolhido um valor inicial para um contador e este é incrementado a cada bloco cifrado. A Figura 4.6 foi criada para mostrar o esquema do WEP de uma forma que permite a comparação com o modo C-CTR.

Se um vetor IV qualquer nunca for reutilizado com a mesma chave, o modo C-CTR e o modo C-CTR com o protocolo WEP coincidem. Como já foi dito, o modo C-CTR não insere fraquezas no esquema.

O adversário apresentado para o modo ECB com o protocolo WEP não funciona com o modo C-CTR. Revendo o adversário A, ele recebe um oráculo $\texttt{E}_K(LR(.,.,b))$ que aceita um par de mensagens como entrada e retorna a cifragem de uma das duas entradas, à esquerda ou à direita, dependendo do valor de b. O objetivo de A é determinar o valor de b. O adversário trabalha como segue (veja os detalhes na seção 2.2.6).

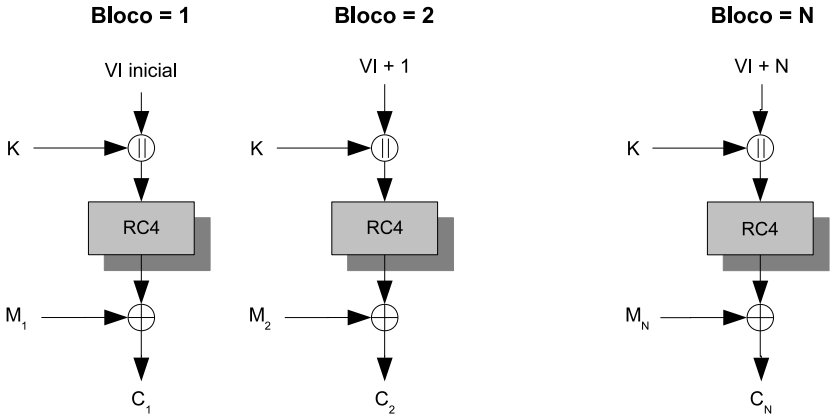

Figura 4.6: O modo C-CTR com o protocolo WEP

Adversário $A^{\mathrm{E}_K(LR(.,.,b))}$
$M_1 \leftarrow 0^{2l}; M_0 \leftarrow 0^l \parallel 1^l$
$C[1]C[2] \leftarrow \mathrm{E}_K(LR(M_0, M_1, b))$
If $C[1] = C[2]$ then return 1 else return 0

No esquema C-CTR com o protocolo WEP, mostrado na Figura 4.6, a criptografia é $C[i] \leftarrow RC4(IV \parallel K) \oplus M[i]$, onde IV não pode ser reutilizado. Neste caso, temos:

No **mundo 1**, $M_{1,0} = 0^l$ é cifrado para $C[1] = RC4(IV \parallel K) \oplus 0^l$
e $M_{1,1} = 0^l$ é cifrado para $C[2] = RC4((IV + 1) \parallel K) \oplus 0^l$
assim,

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}^{ind-cpa-1}_{\mathrm{SE},A} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[RC4(IV \parallel K) = RC4((IV + 1) \parallel K)\right] \\
&= \mathbf{P}\left[IV = IV + 1\right] \\
&= 0.
\end{aligned}$$

No **mundo 0**, $M_{1,0} = 0^l$ é cifrado para $C[1] = RC4(IV \parallel K) \oplus 0^l$
e $M_{1,1} = 1^l$ é cifrado para $C[2] = RC4((IV + 1) \parallel K) \oplus 1^l$

assim,

$$\begin{aligned}
\mathbf{P}\left[\mathbf{Exp}_{\mathtt{SE},A}^{ind-cpa-0} = 1\right] &= \mathbf{P}\left[C[1] = C[2]\right] \\
&= \mathbf{P}\left[RC4(IV \parallel K) = \overline{RC4((IV+1) \parallel K)}\right] \\
&= 2^{-k}
\end{aligned}$$

e assim,

$$\mathbf{Adv}_{\mathtt{SE},A}^{ind-cpa} = 0 - 2^{-k} = -2^{-k}.$$

Este adversário não consegue vantagem alguma no esquema C-CTR com o protocolo WEP. O valor negativo representa apenas a vantagem desse discernidor sendo desprezada no levantamento da vantagem do esquema, já que existem discernidores melhores.

Já que estamos comparando o WEP com o C-CTR, então, a Função Vantagem do esquema obedece a inequação 4.1, com relação a vantagem do cifrador utilizado. Veja na seção 2.2.8 o Teorema 2.25.

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t, q, \mu) \leq 2.\mathbf{Adv}_{F}^{prf}(t, q', lq'), \tag{4.1}$$

onde $q' = \mu/L$.

Considerando $\mathtt{SE}$ o protocolo WEP e F o algoritmo RC4, a equação 4.1 poderia sugerir que sem a reutilização dos vetores IVs o protocolo WEP está sujeito apenas às fraquezas do algoritmo RC4. Mas isto não é verdade pois o protocolo WEP insere outros tipos de fraquezas que poderiam passar despercebidas. No modo C-CTR, o vetor IV é conhecido do atacante. Para o modo C-CTR original isto não implica em problema algum quanto à segurança. Mas no caso do protocolo WEP o vetor IV é concatenado à chave secreta para ser usado como chave semente do RC4. Em outras palavras, uma parte da chave é revelada. Está fraqueza é explorada por várias ferramentas de recuperação da chave do WEP (veja a seção 4.1.6). Uma sugestão para eliminar tal problema é usar o resultado de uma função *hash* com argumentos $IV \parallel K$ no lugar de usá-los como semente do RC4. Como pode ser visto na seção 4.2, o WPA utiliza o algoritmo TKIP para obter essa solução.

4.2 A Função Vantagem no Protocolo WPA

Na prática o protocolo WPA é mais seguro do que o protocolo WEP, pois aborda as fraquezas deste último. Mas o protocolo WPA, assim como no protocolo WEP, utiliza o algoritmo RC4 e portanto está sujeito aos adversários já apresentados.

Moen, Raddum e Hole, em 2004, relataram uma fraqueza no *temporal key hash (the key missing function)*. Segundo eles, com poucas chaves do RC4 é possível recuperar a chave temporal e assim decriptar qualquer pacote que foi cifrado usando tal chave. O ataque tem complexidade de $O(2^{105})$ comparado com o força bruta que é de $O(2^{128})$. Apesar de não ser prático o ataque mostra que partes do protocolo WPA podem ser fracas [Moen et al. 2004].

Além deste ataque, o protocolo WPA ainda é vulnerável a ataques de dicionários já que dados importantes da derivação da chave são revelados durante a fase *4-way-handshaking* (como o MAC e os nonces - números randômicos). Detalhes podem ser encontrados em [Moskowitz. 2004].

Também uma fraqueza do algoritmo Michael já foi relatada por Harkins [Harkins. 2003].

A seguir fazemos a análise do WPA através do Modelo de Função Vantagem.

4.2.1 O Modo C-CTR com o Protocolo WPA

O esquema do protocolo WPA, tal como o esquema do protocolo WEP, também pode ser comparado com o modo C-CTR. A Figura 4.7 mostra o esquema do protocolo WPA de modo a permitir tal comparação.

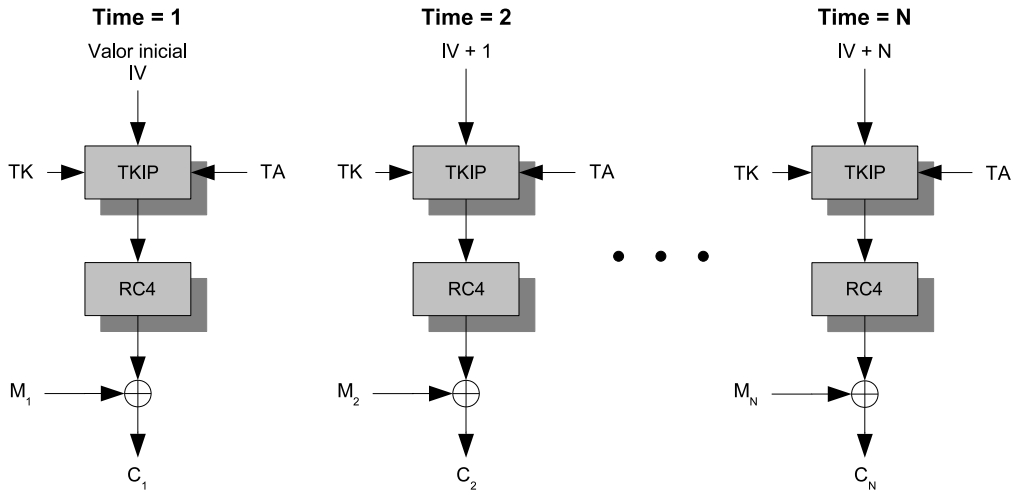

Figura 4.7: O modo C-CTR com o protocolo WPA.

Ao contrário do WEP, o WPA não insere fraquezas. Neste caso, a função vantagem depende das funções vantagens dos algoritmos usados, neste caso o TKIP e o RC4. Veja o Teorema 2.25.

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t, q, \mu) \leq 2.\mathbf{Adv}_{F}^{prf}(t, q', lq'), \tag{4.2}$$

onde $q' = \mu/L$.

Considere que $\mathtt{SE}$ o WPA e F o protocolo TKIP mais o RC4. Isto indica que o WPA está sujeito apenas às fraquezas do TKIP e do RC4.

4.3 A Função Vantagem no Protocolo WPA2

O protocolo WPA2 também conhecido como Padrão IEEE 802.11i é um superconjunto do protocolo WPA. Ele inclui todas as propriedades do protocolo WPA e acrescenta outras; como o algoritmo AES(*Advance Encryption Standard*) em modo contador e o Protocolo CBC-MAC sendo conhecido como protocolo AES-CCMP (*Counter Mode with* CBC-MAC *Protocol*).

4.3.1 Avaliando os Modos do Protocolo WPA2

O protocolo WPA2 usa 2 modos, o modo C-CTR para cifrar os dados e o modo CBC para garantir a integridade dos dados.

O Modo C-CTR com o Protocolo WPA2 (Modo Utilizado para Criptografar os Dados)

O Modo Contador do algoritmo AES para criptografar os dados é, como pôde ser visto na seção 3.3 o modo C-CTR usando o algoritmo AES como cifrador de blocos.

Mais uma vez, segundo Bellare, é possível reduzir o cálculo da vantagem do esquema C-CTR ao cálculo da vantagem do cifrador de blocos utilizado. Veja na seção 2.2.8 o Teorema 2.25.

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t, q, \mu) \leq 2.\mathbf{Adv}_{F}^{prf}(t, q', lq'), \tag{4.3}$$

onde $q' = \mu/L$.

Considerando $\mathtt{SE}$ o protocolo WPA2 e F o algoritmo AES, isto indica que o protocolo WPA2 está sujeito apenas às fraquezas do algoritmo AES. Em outras palavras, o protocolo WPA2 não tem falhas de projeto. Se for usado um bom cifrador de blocos, temos a segurança de que ninguém irá quebrar o esquema criptográfico.

O Modo CBC-MAC com o Protocolo WPA2 (Modo Utilizado para Garantir a Integridade dos Dados Transmitidos)

O modo CBC-MAC também utiliza o algoritmo AES agora para garantir a integridade dos dados transmitidos, como pode ser visto na seção 3.3.

É possível reduzir o cálculo da vantagem do esquema CBC ao cálculo da vantagem do cifrador de blocos utilizado. Veja na seção 2.2.9 o Teorema 2.29.

$$\mathbf{Adv}_{\mathtt{SE}}^{ind-cpa}(t,q,\mu) \leq 2.\mathbf{Adv}_{F}^{prf}(t,q',lq') + \frac{2\mu^2}{l^2 2^l},$$

onde $q' = \mu/L$.

Considerando $\mathtt{SE}$ o modo CBC-MAC e F o algoritmo AES. Isto indica que o modo CBC-MAC está sujeito apenas às fraquezas do algoritmo AES. Em outras palavras, o modo CBC-MAC não tem falhas de projeto. Se for usado um bom cifrador de blocos, é possível estar seguro de que ninguém irá quebrar o esquema criptográfico.

4.4 Considerações Finais sobre a Aplicação do Modelo

Este capítulo mostrou várias formas de aplicação do Modelo de Função Vantagem para análise de esquemas criptográficos. Os esquemas utilizados no Padrão 802.11 serviram de exemplo para a aplicação do modelo.

Mostramos que, de acordo com o modelo utilizado, o WEP é um esquema inseguro com relação à vantagem que um adversário obtém. Este resultado é verificado na prática pelo grande número de ataques possíveis a esse esquema.

Mostramos também que o WPA tem um bom esquema, mas utiliza o algoritmo RC4 que tem diversas fraquezas conhecidas. No entanto, o WPA é, ainda, considerado uma boa opção para uso residencial e em pequenos escritórios, observados alguns cuidados com a escolha da chave. Os

ataques identificados até agora para o WPA, para recuperação da mensagem ou recuperação da chave, são impraticáveis. Mas, a existência de fraquezas no RC4 significa que outros ataques podem surgir no futuro.

Já o WPA2 utiliza um esquema forte o suficiente para garantir que se for usado um cifrador de blocos apropriado, como o AES por exemplo, o usuário poderá ficar tranqüilo quanto à segurança de seus dados. O modelo utilizado diz que o WPA2 não tem fraquezas, ele herda as fraquezas do cifrador subjacente. Isto significa que enquanto o AES for considerado seguro o WPA2 também o será e quando o AES não for mais considerado seguro, o esquema permite a troca do cifrador. Isso é verificado na prática pela inexistência de ataques práticos, para recuperação da mensagem ou recuperação da chave, nesse esquema.

Com a análise apresentada aqui, um profissional de TI pode indicar o uso do WPA2, como opção de garantia de privacidade e autenticidade, com a certeza de que o esquema tem as propriedades necessárias e satisfatórias para alcançar noções rigorosas de segurança.

Capítulo 5

Conclusões e Trabalhos Futuros

5.1 Conclusões

Este trabalho mostra como aplicar o modelo de Função Vantagem como uma ferramenta de modelagem e análise de segurança em esquemas de criptografia. O padrão IEEE 802.11 foi utilizado como laboratório por ser aplicado a ambientes bem hostis, onde as informações trafegam pelo ar, o que permite que qualquer pessoa "próxima" tenha acesso ao texto transmitido, diferentemente de redes cabeadas.

A Função Vantagem modela de forma apropriada os esquemas de segurança. No caso de esquemas de criptografia, a Função Vantagem quanto à IND-CPA é usada para modelar a segurança, além disto, podemos relacionar a segurança do esquema com a segurança do cifrador de blocos utilizado e, então, verificar se o esquema insere fraquezas ou não. A noção IND-CCA mostra também que devemos evitar que o adversário tenha acesso ao computador que decriptografa as mensagens cifradas.

Este trabalho apresentou várias formas de aplicação do Modelo de Função Vantagem para análise de esquemas criptográficos. As análises propostas aqui exemplificam a prática da aplicação do modelo. Na maioria dos casos de estudo é possível comparar o resultado das análises como o comportamento observado do uso dos esquemas. Esquemas fracos com relação a Função Vantagem são também considerados fracos pelo número de ataques conhecidos e vulnerabilidades identificadas. Já esquemas fortes com relação a Função Vantagem têm se mostrado resistentes a ataques e não apresentam vulnerabilidades graves.

Este trabalho fez uma releitura do Modelo de Função Vantagem, elucidando conceito complexos

e exemplificando sua aplicação com esquemas práticos de criptografia, o que pode ajudar aos projetistas e desenvolvedores a maximizar a segurança de seus esquemas.

As análises apresentadas aqui podem recobrar o interesse de pesquisadores no estudo e uso de modelos matemáticos para provas de segurança em esquemas criptográficos. Na prática não é possível provar segurança, apenas podemos prova insegurança. Não é possível prever todos os ataques possíveis a um esquema para, assim, garantir que ele é seguro. Mas, como mostrado aqui, os modelos matemáticos podem cumprir tal papel em muitos casos de forma bem relevante.

5.2 Trabalhos Futuros

Os trabalhos futuros incluem os seguintes tópicos:

- Aplicar o modelo a outros esquemas de privacidade de dados, como os usados em redes cabeadas, bancos de dados cifrados etc.
- Aprimorar os estudos de funções pseudo-randômicas, propondo a implementação de uma ferramenta para detecção de tendências em *streams* pseudo-randômicas, o que poderia ajudar em

Referências Bibliográficas

[Bellare 1998] Bellare, M. (1998). Relations among notions of security for public-key encryption schemes. *CRYPTO '98: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology. London, UK: Springer-Verlag, 1998.*, pages 26–45. ISBN 3-540-64892-5.

[Bellare et al. 1997] Bellare, M., Desai, A., Jokipii, E., and Rogaway, P. (1997). A concrete security treatment of symmetric encryption: Analysis of the des modes of operation. *38th IEEE Symp. on Foundations of Comp. Science.*

[Bellare and Rogaway 2004] Bellare, M. and Rogaway, P. (2004). Course note: Introduction to modern cryptography. In <*http://www-cse.ucsd.edu/users/mihir/cse207/*>.

[Chaplin et al. 2005] Chaplin, C., Qi, E., Ptasinski, H., Walker, J., and Li, S. (2005). 802.11i overview. Technical Report IEEE 802.11-04/0123r1. http://www.ieee802.org/16/liaison/docs/80211-05_0123r1.pdf. Acessado em 21 de Dezembro de 2006., IEEE.

[da Silva 2003] da Silva, J. A. (2003). *Curso de Direito Constitucional Positivo.*, pages 204–208. São Paulo: Malheiros, 22.º edition.

[Davies 2005] Davies, J. (2005). Integridade e criptografia de dados do wi-fi protected access 2. *The Cable Guy - Microsoft TechNet. Disponível em https://www.microsoft.com/brasil/technet/Colunas/community/columns/cableguy/cg0805.mspx. Acessado 21 de dezembro de 2006.*

[Earle 2006] Earle, A. E. (2006). *Wireless security handbook.* Auerbach Publications.

[Fleishman 2003] Fleishman, G. (2003). Weakness in passphrase choice in wpa interface. *Wi-Fi Networking News. Acessado 21 de dezembro de 2006 em URL: http://wifinetnews.com/archives/002452.html*.

[Fluhrer et al. 2001] Fluhrer, S., Mantin, I., and Shamir, A. (2001). Weakness in the key scheduling algorithm of rc4. *Eighth Annual Workshop on Selected Areas in Cryptography, Toronto, Canada.*

[Goldreich et al. 1984] Goldreich, O., Goldwasser, S., and Micali, S. (1984). How to construct random functions. *Journal of the ACM.*, pages 33(4):792–807.

[Goldwasser and Bellare 2001] Goldwasser, S. and Bellare, M. (2001). Lecture notes on cryptography. In *<http://www.cs.ucsd.edu/users/mihir/papers/gb.html>*. Massachusetts Institute of Technology (MIT).

[Harkins. 2003] Harkins., D. (2003). Attacks against michael and their countermeasures. *URL: http://www.ieee802.org/11/Documents/DocumentHolder/3-211.zip. Acessado em 29/01/2007.*

[Mantin and Shamir 2001] Mantin, I. and Shamir, A. (2001). A practical attack on broadcast rc4. *FSE 2001, pp152 a 164. Disponível em <http://www.wisdom.weizmann.ac.il/ itsik/RC4/Papers/bc_rc4.ps>. Acessado em 14 de Janeiro de 2007*.

[MISHRA and W. 2002] MISHRA, A. and W., A. (Fevereiro de 2002). An initial security analysis of the ieee 802.1x standard. *Disponível em <http://www.cs.umd.edu/ waa/1x.pdf>. Acesso em 23 de Dezembro de 2006*.

[Moen et al. 2004] Moen, V., Raddum, H., and Hole., K. J. (2004). Weakness in the temporal key hash of wpa. *Department of Informatics, Univ. of Bergen, April 5, 2004. URL: http://www.nowires.org/Papers-PDF/WPA_attack.pdf*.

[Moskowitz. 2004] Moskowitz., R. (2004). Weakness in passphrase choice in wpa interface. *URL: http://wifinetnews.com/archives/002452.html. Acessado em 29/01/2007.*

[Ostrovsky 1995] Ostrovsky, R. (1995). Introduction to cryptography - lecture notes. In *<http://www.cs.ucla.edu/ rafail/PUBLIC/book.pdf>*. U.C. Berkeley.

[Paul and Preneel 2004] Paul, S. and Preneel, B. (2004). A new weakness in the rc4 keystream generator and an approach to improve the security of the cipher. *FSE 2004, pp245 a 259. Disponível*

em <http://www.cosic.esat.kuleuven.be/publications/article-40.pdf>. Acessado em 14 de Janeiro de 2007.

[Sankar et al. 2004] Sankar, K., Sundaralingam, S., Balinsky, A., and Miller., D. (2004). *Cisco Wireless LAN Security.* Cisco Press.

[Silva 2005] Silva, G. M. (2005). Segurança em redes locais sem fio. Master's thesis, Universidade Federal de Uberlândia.

[Stallings 1998] Stallings, W. (1998). *Cryptography and network security: principles and practice*, chapter 3, pages 83–89. Prentice-Hall, Inc., ed. 2 edition.

[Stubblefield et al. 2002] Stubblefield, A., Ioannidis, J., and Rubin, A. D. (2002). Using the fluhrer, mantin, and shamir attack to break wep. *Proceedings of the 2002 Network and Distributed Systems Security Symposium (NDSS'02), San Diego, CA, February 2002. The Internet Society.*

[Suriyajan 2006] Suriyajan, C. (2006). Advance topic in distributed system digital entertainment security mechanisms security in wireless network. *Universität Stuttgart*.

Apêndice A

Manutenção e Troca de Chaves nas Redes Locais sem Fios

A dificuldade do administrador da rede em gerar, distribuir e gerenciar as chaves é um dos grandes problemas do Padrão IEEE 802.11 original. Se um *laptop* é perdido, por exemplo, a segurança da rede pode estar comprometida. As chaves também podem ser comprometidas durante uma distribuição em massa. Isso motiva o desenvolvimento de uma distribuição dinâmica de chaves [Sankar et al. 2004].

O Padrão IEEE 802.11i aborda o problema de gerenciamento de chaves dividindo-o em duas fases: estabelecimento da chave mestra (*master key establishment*) e troca das chaves (*key exchange*). O estabelecimento da chave mestra pode acontecer manualmente via configuração das estações, ou dinamicamente através do protocolo 802.1x usando EAP. Após o estabelecimento da chave mestra as duas partes executam a troca das chaves temporais. Na verdade, a troca das chaves é feita por uma negociação, onde cada lado gera as chaves, não havendo, assim, a necessidade de transmissão de chaves [Sankar et al. 2004].

A.1 Estabelecimento da Chave Mestra

Um método suficiente para estabelecer a chave mestra é o EAP over LAN (EAPOL). O protocolo 802.1x é utilizado para forçar a autenticação de mensagens via EAPOL entre a estação (cliente) e o servidor de autenticação (SA). A seção 3.4 descreveu rapidamente o 802.1x (para mais detalhes veja [Sankar et al. 2004]). Ao se conectar ao ponto de acesso (AP), a estação é primeiro autenticada

usando *open system authentication*, isto é, não há uma autenticação verdadeira. Em seguida, a estação e o AP são conduzidos para uma autenticação mútua usando EAP, em que a verdadeira autenticação acontece. A estação cliente e o servidor de autenticação (AS) negociam e autenticam-se até ambos os lados ficarem convencidos de que cada um está conversando com quem eles realmente esperam e que cada lado conhece as propriedades secretas necessárias. Após isto, o AS envia uma mensagem de *EAP success*, indicando que a autenticação ocorreu com sucesso [Sankar et al. 2004].

O processo de autenticação mútua gera uma chave compartilhada entre o AS e a estação. Após a chave ter sido estabelecida, o AS transfere esta chave para o AP via RADIUS. Esta chave é conhecida como *Pairwise Master Key* (PMK). O AP não participa da negociação da chave, apenas conduz as mensagens. No caso do TKIP o administrador da rede pode optar por não usar o 802.1x, então essa chave deve ser configurada manualmente na estação e no AP, e é conhecida como *Preshared Key (PSK)*. Para o restante do capítulo não faz diferença se a chave mestra é PMK ou PSK, por isso será tratada como PMK [Sankar et al. 2004].

A.2 Hierarquia das Chaves

O 802.11i especifica dois tipos de chaves: *Pairwise* (para tráfego *unicast*, envio de mensagem entre uma estação e um AP e vice-versa) e *Group* (para tráfego *multicast*, envio de uma mesma mensagem entre um AP e muitas estações). O nó raiz das chaves *unicast* é a PMK e das *multicast* é a *Group Master Key* (GMK). O tempo de vida da PMK pode ser longo e se conservar entre muitas associações a um AP. A GMK por outro lado, deve ser reconfigurada toda vez que uma estação se desassocia, ou em intervalos regulares de tempo. O motivo é que todas as estações conhecem a GMK e isto pode representar um risco. Estas chaves mestras são usadas para derivar várias outras que são apresentadas a seguir [Sankar et al. 2004]. A Tabela A.1 resume os diferentes tipos de chaves usadas no 802.11i.

A.2.1 Hierarquia das Chaves de *Pairwise*

A PMK é a raiz da hierarquia de todas as chaves do tipo *pairwise* do 802.11i. Somente uma única PMK pode existir entre cada estação cliente e o AP ao qual está associada. A Figura A.1 ilustra a hierarquia de chaves formada a partir da PMK.

A partir da PMK, a estação e o AP derivam três chaves usando uma função pseudo-randômica. A função pseudo-randômica gera primeiro uma chave intermediária, chamada PTK. Ela usa dois

Chave	Uso	Origem
Preshared Key (PSK)	Usada em TKIP, WEP ou CCMP	Configurada
Pairwise Master Key (PMK)	Usada como duração longa para derivar outras	Criada na negociação EAP
Pairwise Transient Key (PTK)	Usada para derivar outras chaves de unicast	Derivada da PMK ou PSK através do *4-way handshake*
Group Transient Key (GTK)	Usada para derivar outras chaves multicast	Derivada da PMK através do 4-way-handshake.
Temporal Key (TK)	Para o TKIP é a combinação de uma TEK e uma chave MIC. No CCMP é somente a TK.	Derivada do PTK
Temporal Encryption Key (TEK)	No TKIP é usada para cifrar os pacotes	Derivada da PTK ou GTK
MIC (Michael) Key	No TKIP é usada para calcular o MIC.	Derivada do PTK ou GTK.
EAPOL Key Encryption Key (KEK)	Usada na negociação da nova GTK	Derivada do PTK.
EAPOL Key Confirmation Key (KCK)	Usada na negociação da nova GTK para prover integridade para as mensagens	Derivada da PTK

Tabela A.1: Tipos de Chaves do 802.11i

nonces (números únicos) originados um do suplicante (estação) e outro do autenticador (SA). O PTK é então dividido em três chaves: a EAPOL KCK, a EAPOL KEK e a TK. No TKIP, a TK tem 256 bits e é particionada em TEK e a chave Michael, mas no CCMP ela tem só 128 bits. A função pseudo-randômica é o SHA-1 [Sankar et al. 2004].

A.2.2 Hierarquia das Chaves de *Group*

O GMK pode ser usado como raiz para as chaves do tipo group. A Figura A.2 ilustra essa hierarquia.

Ao contrário da PMK, que é derivada por ambos os lados, a GMK é gerada pelo AP. O AP usa a GMK para gerar a GTK que tem o tamanho de 256 bits e é aplicada a várias chaves que são usadas em pacotes *multicast*. A GTK pode ser usada com WEP, TKIP ou CCMP. Os tamanhos utilizados pelos protocolo são mostrados na Figura A.2. Caso alguma estação não suporte um protocolo de cifragem, será usado o menor denominador comum. Como por exemplo, se algumas estações só aceitam o TKIP e algumas aceitam o CCMP, para esse caso o menor denominados comum é o TKIP e portanto ele será usado para comunicações de *multicast* [Sankar et al. 2004].

Após estabelecer a chave mestra (PMK), os dois lados estão prontos para negociar as chaves transitórias, que serão usadas apenas por uma sessão. Isto é feito através de um processo seguro conhecido como *4-way-handshake* (aperto de mãos em quatro passos). No caso das chaves de grupo, temos o *group key handshake*. As chaves criadas nestes processo são usadas até que sua validade termine e em seguida são destruídas [Sankar et al. 2004].

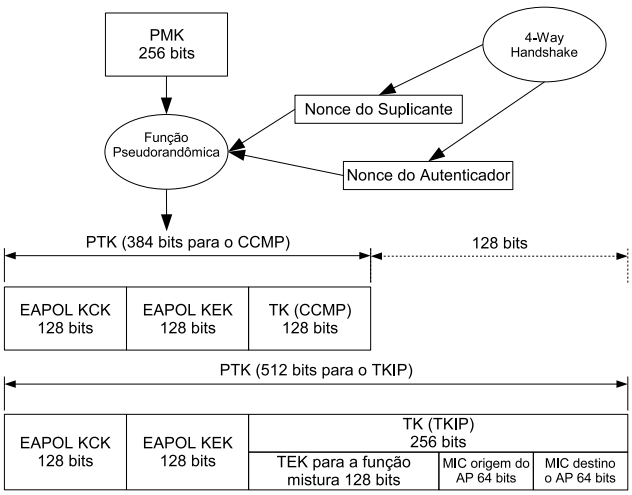

Figura A.1: Hierarquia das chaves de *Pairwise*

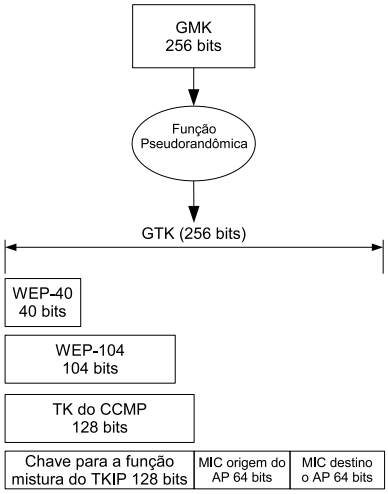

Figura A.2: Hierarquia das chaves de *Group*

A.3 O *4-Way-Handshake*

Após a autenticação e o estabelecimento da PMK, a estação deve usar o *4-way-handshake* para estabelecer, junto com o AP, as chaves transitórias. O *4-way-handshake* é composto de quatro trocas de pacotes de mensagens EAPOL. Ele assegura que os dois lados conhecem a PMK, troca os *nonces* (números únicos que são usados para construir a hierarquia de chaves) e distribui a GTK. A Figura A.3 ilustra bem o processo.

O *4-way-handshake* inicia com o autenticador (AP) gerando um *nonce* (número único). Este *nonce* nunca deve ter sido usado com a mesma PMK para prover proteção contra reenvio de pacotes nesse processo. O *nonce* é enviado na mensagem 1, identificado como *Anonce* na Figura A.3. O

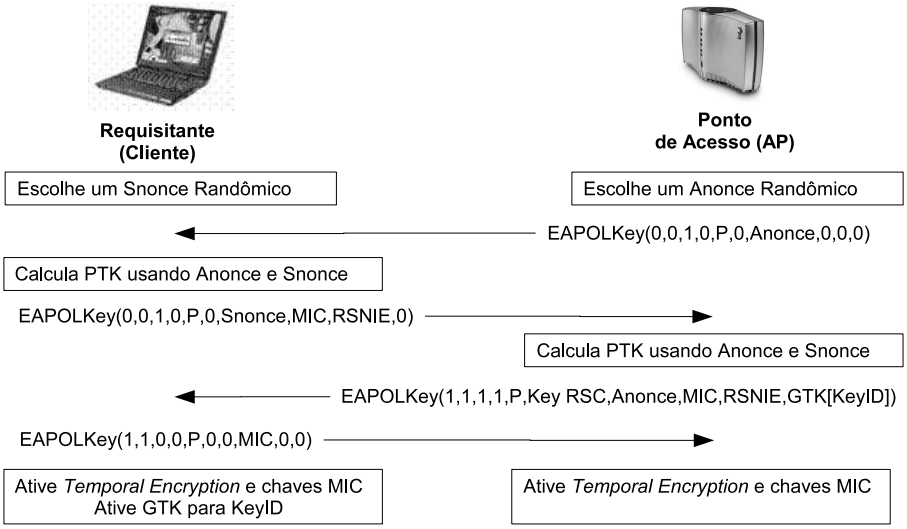

Figura A.3: Esquema do *4-Way-Handshake*

suplicante (cliente) gera seu próprio *nonce* (*Snonce*) e usa os dois nonces junto com a PMK para gerar a PTK, como mostrado na seção A.2.1. O suplicante responde de volta com seu próprio nonce e um MIC feito com sua PMK, na mensagem 2. O autenticador agora tem os dois *nonces* e pode gerar a PTK. Ele verifica o MIC da mensagem 2. Se a verificação foi bem-sucedida, ele gera o GTK se necessário e o envia na mensagem 3, incluindo o contador de seqüência de recebimento (RSC) para as mensagens GTK. O suplicante verifica o MIC na mensagem 3, instala as chaves e envia a mensagem 4 com a confirmação. O autenticador recebe a mensagem 4 e instala as mesmas chaves. Neste momento, ambos se autenticaram e cada um tem certeza que o outro conhece a PMK [Sankar et al. 2004] [Suriyajan 2006].

A.4 O *Handshake* da Chave de Grupo

O *group-key-handshake* é usado quando é necessário prover uma nova chave de grupo, pois esta chave já é distribuída uma vez no *4-way-handshake*. Ele usa apenas duas mensagens, já que o AP é que gera a GTK. Nesse processo, são aplicadas as chaves EAPOL KEK e o EAPOL KCK que ambos os lados derivaram da PMK. Veja a Figura A.4.

O autenticador envia a mensagem 1 que contém a nova GTK cifrada com a KEK. A mensagem tem também um MIC que foi criado usando a KCK e o *Receive Sequence Counter*(RSC). O cliente responde com o RSC e um MIC que será verificado pelo autenticador.

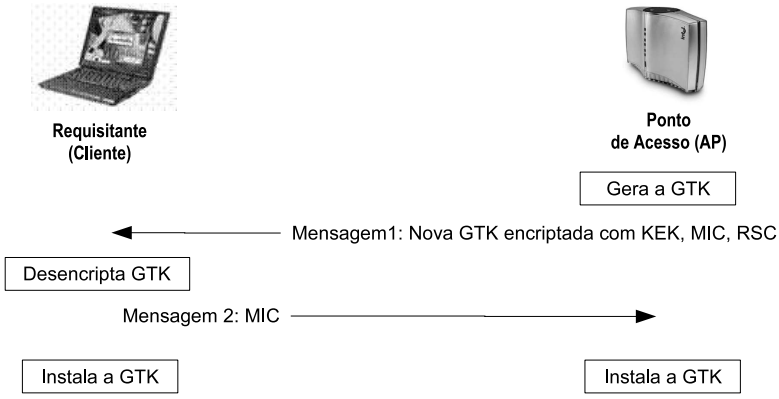

Figura A.4: Esquema do *Group Key Handshake*