MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE MESTRADO EM SISTEMAS E COMPUTAÇÃO

CAP LEANDRO DE MATTOS FERREIRA

A APLICAÇÃO DE WAVELETS NO RECONHECIMENTO DE PADRÕES CRIPTOGRÁFICOS

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA

CAP LEANDRO DE MATTOS FERREIRA

A APLICAÇÃO DE WAVELETS NO RECONHECIMENTO DE PADRÕES CRIPTOGRÁFICOS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.

Orientador: Prof. José Antonio Moreira Xexéo - D.Sc.

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP 22290-270

005.82 Ferreira, Leandro de Mattos
F383a A Aplicação de Wavelets no Reconhecimento de Padrões Criptográficos / Leandro de Mattos Ferreira, orientado por José Antonio Moreira Xexéo - Rio de Janeiro: Instituto Militar de Engenharia, 2017.

137p.: il.

Dissertação (mestrado) - Instituto Militar de Engenharia, Rio de Janeiro, 2017.

1. Curso de Sistemas e Computação - teses e dissertações. 1. Criptografia. 2. Recuperação da informação. I. Xexéo, José Antonio Moreira . II. Título. III. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

CAP LEANDRO DE MATTOS FERREIRA

A APLICAÇÃO DE WAVELETS NO RECONHECIMENTO DE PADRÕES CRIPTOGRÁFICOS

Dissertação de Mestrado apresentada ao Curso de Mestrado em Sistemas e Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Mestre em Ciências em Sistemas e Computação.

Orientador: Prof. José Antonio Moreira Xexéo - D.Sc.

Aprovada em 14 de Dezembro de 2017 pela seguinte Banca Examinadora:

Prof. José Antonio Moreira Xexéo - D.Sc. do IME - Presidente

Prof. Flávio Luis de Mello - D.Sc. da POLI/UFRJ

Prof. Geraldo Bonorino Xexéo - D.Sc. da COPPE/UFRJ

Prof. Julio Cesar Duarte - D.Sc. do IME

Prof. Anderson Fernandes Pereira dos Santos - D.Sc. do IME

Rio de Janeiro
2017

Ao Instituto Militar de Engenharia, alicerce da minha formação e aperfeiçoamento.

AGRADECIMENTOS

Agradeço a todas as pessoas que me incentivaram, apoiaram e possibilitaram esta oportunidade de ampliar meus horizontes.

Meus familiares, cônjuge e mestres.

Em especial ao meu Professor Orientador Dr. José Antônio Moreira Xexéo, por sua disponibilidade e atenção.

"Humanity has the stars in its future, and that future is too important to be lost under the burden of juvenile folly and ignorant superstition. "

ISAAC ASIMOV

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

ECB	Electronic Codebook
CBC	Cipher Block Chaining
AES	Advanced Encryption Standard
3DES	Triple Data Encryption Standard
NIST	National Institute of Standards and Technology
TF	Term Frequency
IDF	Inverse Document Frequency

RESUMO

Este trabalho combina o uso de transformadas *wavelet* e técnicas de Recuperação de Informação, como ferramentas para realizar o agrupamento e classificação de criptogramas gerados em modo ECB por sistemas criptográficos distintos. Resolver os problemas de agrupamento e classificação servem como primeiro passo numa análise criptográfica, e portanto são relevantes em situações reais. Os sistemas analisados foram o AES, Twofish, 3DES, RC6 , Serpent e DES. Uma base de criptogramas foi gerada com estes sistemas utilizando textos em claro em inglês presentes na base Reuters-21578. As transformadas *Wavelet* foram utilizadas sobre os blocos do criptograma, juntamente com a aplicação de técnicas de Recuperação de Informação para agrupamento e classificação dos criptogramas. A divisão dos criptogramas em blocos menores do que o bloco de cifragem foi analisada, resultando em tempos menores de execução do agrupamento, e como principal resultado levando à criação de um classificador binário para 3DES. O uso de *wavelets* trouxe resultados similares aos do emprego apenas de técnicas de RI, traduzindo-se em vantagem apenas na redução no uso de espaço em disco.

ABSTRACT

This work combines the application of wavelet transforms and Information Retrieval techniques, as tools to implement the grouping and classification of ciphers creatd by distinct algorithms in ECB mode. Solving the grouping and classification problem serves as a first step in a cryptanalysis, and therefore are relevant in real world applications. The cryptographic systems analyzed were AES, Twofish, 3DES, RC6 , Serpent and DES. A database of ciphers was created using these systems applied over plaintexts in English contained on the Reuters-21578 database. The wavelet transforms were applied over the cipher blocks, along with the use of Information Retrieval techniques. Also, the division of the cipher into blocks smaller than the encryption block was analysed, resulting in shorter execution times for grouping, and producing, as the main result, a binary classifier for the 3DES algorithm. The use of wavelets brought only results similar to those obtained using only Information Retrieval techniques, bringing only a reduction of disk space usage as a positive result.

1 INTRODUÇÃO

A criptografia torna uma mensagem legível (chamada de texto em claro) em uma mensagem ilegível (chamada criptograma) que apenas o destinatário da mensagem pode tornar novamente legível. A criptografia utiliza um algoritmo criptográfico e uma chave. A dificuldade de um atacante em descobrir a mensagem a partir do criptograma se dá pelo fato do atacante não conhecer a chave, ao contrário do destinatário, que a possui. Através da história, diversos algoritmos para criptografar foram desenvolvidos, desde a mais simples cifra de substituição monoalfabética (conhecida como cifra de César) até os modernos sistemas computacionais como o *Advanced Encryption Standard* (AES). Após o surgimento de computadores capazes de testar por força bruta um grande número de chaves em pouco tempo, a criptografia passou a gerar algoritmos criptográficos que criam um problema de difícil solução computacional sem a chave correta, mas de fácil solução caso se possua.

A linguagem natural em qualquer idioma apresenta redundância. Essa característica, explicada por (SHANNON, 1948) faz com que padrões emerjam nos textos escritos. As cifras de bloco atuais separam um texto em claro em blocos de tamanho iguais para cifragem e possuem alguns modos de operação. O modo de operação ECB preserva esses padrões pois cada bloco cifrado depende apenas do bloco de entrada e da chave. Portanto blocos em claro iguais, cifrados com a mesma chave, geram blocos cifrados iguais.

Segundo (SHANNON, 1949), a presença de redundância no texto em claro é propagado pelo processo de cifragem, de tal modo que os padrões repetitivos do texto se propagam para os criptogramas. Esses padrões todavia se encontram ocultos como resultado da confusão e difusão presentes no processo de cifragem. A presença desses padrões permitiu que as cifras fossem agrupados pelo par (algoritmo, chave) por diversos trabalhos anteriores. Este trabalho vai além, buscando realizar a classificação de cifras, preferencialmente de acordo apenas com o algoritmo que as geraram.

Confusão é a propriedade do criptograma ser altamente distinto do texto em claro, preferencialmente através de uma transformação não-linear, dificultando a análise reversa do criptograma. A difusão consiste na dependência de diversos bits do criptograma para cada bit do texto em claro, assim difusão total implica que todos os bits da cifra dependem de todos os bits do texto em claro.

A presença de padrões permite que quando identificados se consiga agrupar cifras geradas pelo mesmo algoritmo de cifragem. Possuindo-se um conjunto de treinamento de

cifras de um determinado algoritmo pode-se então classificar uma nova cifra de acordo com o algoritmo que a gerou. A esses processos dão-se os nomes de Problema de agrupamento e classificação de criptogramas, respectivamente. Esses problemas estão caracterizados na Seção 1.2 abaixo.

1.1 MOTIVAÇÃO

Criptoanálise busca recuperar a mensagem original detectando falhas no algoritmo de cifragem ou descobrindo a chave correta por algum meio. Segundo os princípios descritos em (KERCKHOFFS, 1883) a segurança do sistema deve estar contida na chave e não no desconhecimento do método de cifragem. Dessa forma é assumido que o atacante sabe tudo sobre o algoritmo de cifragem, suas peculiaridades e vulnerabilidades. Em situações reais, para que o atacante seja capaz de explorar essas deficiências é necessário saber qual algoritmo foi utilizado na cifragem.

No contexto de estudos de ataques recentes a algoritmos criptográficos, dois modelos de ataques também são relevantes: o *grey-box* e *white-box*. Em ambos os casos, o conhecimento do tipo de algoritmo utilizado para cifragem é fundamental para que os ataques sejam bem sucedidos. O modelo tradicional, onde o atacante não tem acesso ao processo de cifragem, apenas suas entradas e saídas (textos em claro e cifras respectivamente) ficou então conhecido como *black-box*.

No modelo *grey-box* o atacante tem acesso a pelo menos uma das informações secundárias do processo de cifragem, como por exemplo: tempo de processamento, temperatura, consumo de corrente do processador ou radiação eletromagnética. Essas informações são chamadas de *Side Channel Information* (Informação de canal alternativo).Segundo (CHOW et al., 2003) o conhecimento do processo de cifragem em si pode ser considerando um tipo de informação pertinente ao contexto *grey-box*, portanto um processo de descoberta do sistema criptográfico (conforme apresentado neste trabalho) pode ser considerado uma forma de permitir o uso de técnicas específicas dependentes do algoritmo, efetivamente fazendo um ataque que utiliza o contexto *grey-box* em um ambiente *black-box* (onde o atacante não sabe qual sistema criptográfico foi utilizado).

Já no modelo *white-box* o atacante tem potencialmente acesso total ao processo de cifragem, incluindo acesso pleno à memória, podendo ver os subconjuntos das chaves nela instanciados. Nesse modelo o mero conhecimento de qual método criptográfico está sendo utilizado já pode permitir a obtenção da chave, caso o processo de cifragem não tenha sido preparado para lidar com esse tipo de ataque. Esse modelo não foi explorado nos

experimentos que deram origem à esta dissertação.

Neste trabalho, foi analisado o uso de transformadas integrais (especialmente as *wavelets*) juntamente com técnicas de Recuperação de Informação (RI) para auxiliar na tentativa de descobrir qual sistema criptográfico foi utilizado para gerar determinado criptograma. Ao interpretar a cifra como um sinal digital, o uso dessas transformadas permitiu a identificação de padrões através da mudança de domínio da função representada por esse sinal. Com essa nova forma de analisar a informação presente no criptograma, prosseguiu-se então para resolver os problemas de agrupamento e classificação de criptogramas.

1.2 CARACTERIZAÇÃO DO PROBLEMA

Estudos anteriores apontaram que diferentes textos (que contêm redundância), quando cifrados por um mesmo algoritmo e chave usando o modo *Electronic CodeBook* (ECB), fazem padrões emergir nos criptogramas resultantes. Através da semelhança entre esses criptogramas (ou seja a repetição de padrões entre eles) pode-se então agrupá-los segundo os algoritmos que os geraram.

O objetivo do algoritmo de separação de criptogramas é, dado um conjunto de criptogramas como entrada, separar esses criptogramas em diferentes grupos, onde cada grupo conterá os elementos cifrados por um determinado algoritmo. Por exemplo, caso o conjunto de entrada seja composto dos seguintes elementos :

a) C1(T1,AES,k1)

b) C2(T2,AES,k1)

c) C3(T2,3DES,k2)

d) C4(T3,3DES,k3)

e) C5(T1,RSA,k3)

f) C6(T2,RSA,k4)

g) C7(T1,Serpent,k1)

h) C8(T3,Serpent,k5)

Onde C1(T1,AES,k1) significa a cifra C1 gerada usando AES e chave k1 sobre o texto T1. O problema está representado na Fig. 1.1. Os 4 grupos que devem ser encontrados estão dispostos na Tabela 1.1.

Grupo 1	Grupo 2	Grupo 3	Grupo 4
C1(T1,AES,k1)	C3(T2,3DES,k2)	C5(T1,RSA,k3)	C7(T1,Serpent,k1)
C2(T2,AES,k1)	C4(T3,3DES,k3)	C6(T2,RSA,k4)	C8(T3,Serpent,k5)

TAB. 1.1: Separação em grupos

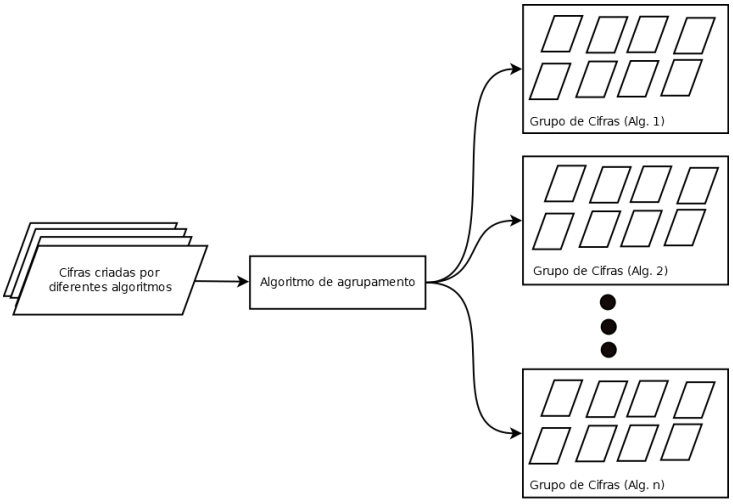

FIG. 1.1: Caracterização do problema de agrupamento

A segunda parte do problema é realizar a classificação. Nesse problema a entrada é um criptograma cujo método de cifragem é desconhecido. Através de uma base de treino composta por um conjunto de textos cifrados por diferentes algoritmos e previamente agrupados, é feita a classificação do criptograma da entrada caso ele corresponda a algum dos padrões treinados. O problema está representado na Fig. 1.2. No caso especifico do classificador apresentado neste trabalho existem duas classificações possíveis: Classe "3DES/DES"ou "Finalistas do AES"(não 3DES/DES). A eficácia do classificador foi avaliada utilizando-se as medidas de precisão, abrangência e acurácia.

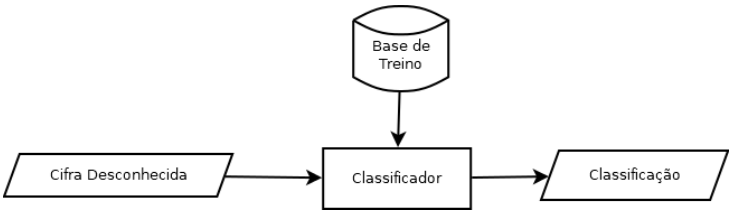

FIG. 1.2: Caracterização do problema de classificação

Visando a resolver esses problemas, inicialmente (CARVALHO, 2006) utilizou um processo de Recuperação de Informação para agrupar criptogramas e o método K-NN para classificação. (SOUZA, 2007) realizou uma análise das métricas de semelhança e distância, além de inserir o uso de redes neurais ao processo de agrupamento. Já (OLIVEIRA, 2011) introduziu o uso de algoritmos genéticos ao problema de agrupamento, reduzindo o esforço

computacional e eliminando a necessidade de conhecimento prévio do número de grupos existentes. Por fim (TORRES, 2011) busca o agrupamento através do uso de grafos, gerando um processo mais eficiente e adicionando uma nova métrica de avaliação.

Esses estudos conseguiram agrupar criptogramas baseados nos pares (algoritmo, chave). Em uma situação real, a hipótese de apenas uma chave ser utilizada é restritiva. Com a presença de múltiplos algoritmos e múltiplas chaves o problema de agrupamento e classificação se torna mais complexo e relevante para uso em ambientes práticos. Deste modo um classificador que distingue entre cifras geradas pelos diferentes algoritmos independente de chave traz um passo importante para criptoanálise. Este trabalho buscou realizar a tarefa de classificação, indo além da tarefa de agrupamento apenas. Além disso utilizou-se de múltiplas chaves. Como principal resultado obteve-se um classificador binário para 3DES, para textos em inglês de tamanho superior a 10KB.

1.3 EMPREGO DE *WAVELETS*

Este trabalho teve como objetivo analisar o emprego de *Wavelets* para auxiliar na resolução desses dois problemas. As transformadas integrais do tipo *wavelet* mudam o domínio da função à qual é aplicada, permitindo análises de frequência e tempo. A função *wavelet* mãe, através do uso de operações básicas de escala e deslocamento, é capaz de representar o sinal (ou função). O seu uso permite a aproximação de um sinal por uma série de funções ajustadas por apenas alguns parâmetros. dessa forma reduzindo a quantidade de informação a ser tratada.

No contexto de Recuperação de Informação, pode-se utilizar um modelo vetorial para documentos, calculando em sequência a matriz de similaridade entre os documentos. Para trabalhar com cifras em bloco, o dicionário de palavras possíveis são as diferentes sequências binárias do tamanho do bloco. Dessa forma existem 2^{m} possíveis palavras, onde m é o número de bits de um bloco. A utilização de *wavelets* se dá então aplicando a transformada integral em cada palavra distinta (cada possível bloco). Em seguida analisa-se a matriz de similaridade procurando os números mais similares entre dois criptogramas quaisquer, buscando agrupá-los por um mesmo conjunto (algoritmo, chave).

1.4 JUSTIFICATIVA

O uso de técnicas de Recuperação de Informação para agrupamento e classificação de criptogramas gerados por sistemas operando em modo ECB foram analisados em trabalhos anteriores por (OLIVEIRA et al., 2006), (SOUZA et al., 2011), e (TORRES; OLIVEIRA,

2011). O problema de classificação de algoritmos criptográficos foi analisado por (NAGIREDDY, 2008), obtendo bons resultados para os executados em modo ECB, mas não conseguindo resultados expressivos em modo *Cypher Block Chaining* (CBC). Além disso, este trabalho testou diversas técnicas para detecção de padrões em algoritmos.

Quanto ao uso de processamento de sinais digitais e *wavelets* no contexto de Recuperação de Informação pode-se citar (SILVA, 2007) que demonstrou a possibilidade do uso desta técnica na representação dos textos e documentos para facilitar a busca e recuperação de informação. Continuando este trabalho, (FERREIRA, 2011) realizou a comparação do uso de *wavelets* distintas no problema de busca e recuperação de informação. Esses dois trabalhos indicaram a possibilidade da aplicação de *wavelets* para o problema de detecção de padrões em criptogramas.

Os resultados obtidos nos trabalhos supracitados (uso de Técnicas de RI sobre criptogramas e uso de transformadas *wavelet* em conjunto com RI) justificam o estudo deste trabalho que visa a combinar estes dois resultados e unir transformadas *wavelet* e técnicas de RI para agrupamento e classificação de criptogramas.

1.5 OBJETIVOS E CONTRIBUIÇÕES ESPERADAS

O objetivo principal deste trabalho é o agrupamento e classificação de criptogramas gerados em modo ECB através do uso de *wavelets*. Para se alcançar esse objetivo organizou-se uma base de dados de criptogramas gerados por diferentes algoritmos e chaves, e ao final obteve-se um sistema capaz de agrupá-los e também de classificar novos criptogramas gerados por um dos métodos conhecidos.

Os objetivos específicos deste trabalho são: avaliar o uso de divisores do bloco e técnicas de RI para agrupamento e classificação de criptogramas; avaliar a adequação do uso de *wavelets* para recuperação de informação em criptogramas; identificar a presença de assinaturas nos criptogramas geradas pelo algoritmo de cifragem; detectar padrões repetitivos gerados pela redundância de textos dentro de criptogramas e investigar a viabilidade do uso da técnica em criptogramas cifrados em modo CBC.

Para realizar esses objetivos elencaram-se as seguintes questões de pesquisa:

a) O uso de técnicas de RI, considerando palavras de tamanho inferior ao bloco de cifragem pode auxiliar na resolução do problema de agrupamento e classificação de criptogramas, principalmente com uso de múltiplas chaves?

b) A representação dos blocos do criptograma através da sua transformada *wavelet*

pode tornar mais eficiente o processo de agrupamento e classificação de criptogramas?

c) A representação do criptograma como um sinal e a subsequente aplicação de diversas transformadas *wavelet* pode facilitar a identificação de padrões emergentes nos criptogramas?

d) A aplicação de um determinado algoritmo de cifragem gera padrões identificáveis ("Assinaturas") ao invés de se aproximar de uma distribuição aleatória uniforme binária?

As contribuições esperadas para este trabalho são:

a) Análise quantitativa do uso de técnicas de RI utilizando divisores do bloco de cifragem como palavra para resolução problema de agrupamento e classificação de criptogramas, principalmente com uso de múltiplas chaves.

b) Análise quantitativa do uso de transformadas *wavelet* para resolução do problema de agrupamento e classificação de criptogramas, associadas ao algoritmo de cifragem e modo de operação.

c) Comparação do uso de diferentes *wavelets* para a resolução do problema de agrupamento e classificação de criptogramas.

d) Determinação da capacidade de uso de *wavelets* para detecção de padrões gerados pelo método de cifragem utilizando a interpretação direta da cifra como um sinal.

e) Criação de uma base de dados de cifras geradas com múltiplos algoritmos aplicados sobre textos em claro de uma base de textos de fácil acesso e amplo emprego.

1.6 METODOLOGIA

A pesquisa proposta empregou o método indutivo, e realizou uma abordagem quantitativa e experimental. Para atingir os objetivos propostos, os seguintes passos foram seguidos:

(i) Definição do conjunto de *wavelets* a ser utilizado no trabalho, mediante a análise das características de cada uma e seleção das aparentemente mais aptas para utilização neste trabalho.

(ii) Definição do conjunto de algoritmos criptográficos a ser usado no trabalho. Preferencialmente foram escolhidos algoritmos de amplo uso e reconhecidos como importantes pela comunidade acadêmica.

(iii) Confecção do banco de dados de criptogramas. Com a correta escolha de base de dados é possível garantir a reprodutibilidade dos experimentos, além de facilitar a aceitação dos resultados. Preferencialmente foram analisados bancos de dados com amplo uso pela comunidade acadêmica; por fim decidiu-se pela confecção do banco de criptogramas, utilizando textos em claro escolhidos de uma base de dados similarmente reconhecidos pela comunidade acadêmica.

(iv) Desenvolvimento e implementação de versão do algoritmo de agrupamento e posterior classificação de criptogramas utilizando as *wavelets* escolhidas.

(v) Realização de experimentos para medir a influência do uso de diferentes tamanhos de palavras, relevância do uso de chaves distintas, eficiência do classificador obtido, principalmente perante variação do tamanho da cifra a ser classificada.

(vi) Realização de experimentos para aferir a viabilidade do emprego de transformadas *wavelet* em conjunto com as técnicas de RI, novamente variando-se tamanho de palavra, uso de chaves distintas e também o uso de diferentes bases vetoriais.

(vii) Comparação da eficiência do algoritmo implementado com os resultados obtidos anteriormente pelo GSI/IME com outros algoritmos. Foram consideradas as medidas: Precisão, abrangência, acurácia e tempo de execução.

(viii) Confecção da dissertação contendo os experimentos e os resultados obtidos.

1.7 ORGANIZAÇÃO DA DISSERTAÇÃO

O capítulo 2 apresenta os conceitos fundamentais da pesquisa envolvida neste trabalho, além de outros trabalhos relacionados de forma direta a esta dissertação.

O capítulo 3 apresenta os experimentos com o uso de técnicas de Recuperação de Informação e divisão de criptogramas em blocos de tamanhos inferiores ao tamanho do bloco de cifragem, apresentando os resultados encontrados e também a descrição da base de dados utilizada em todos os experimentos deste trabalho.

O capítulo 4 apresenta o desenvolvimento, através de experimentação, da principal contribuição deste trabalho: um classificador binário para 3DES/DES capaz de separar cifras geradas pela aplicação desses algoritmos sobre textos em inglês.

O capítulo 5 descreve os experimentos realizados com o uso de *Wavelets* sob o paradigma da Recuperação de Informação, apresentando também os resultados desses experimentos.

Por fim, no capítulo 6 são apresentadas as conclusões e contribuições do trabalho realizado e as possibilidades de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo trata dos fundamentos teóricos necessários para o acompanhamento dos conceitos apresentados posteriormente nesta dissertação. Nas seções seguintes serão tratados os seguintes assuntos: Criptografia simétrica de bloco, técnicas de Recuperação de Informação, transformadas *wavelet*, classificador K-NN e por fim uma análise de trabalhos relacionados ao tema desta dissertação.

2.1 CRIPTOGRAFIA SIMÉTRICA DE BLOCO

A criptografia simétrica está embasada no uso de uma única chave por ambas as partes envolvidas na comunicação. Tanto o emissor quanto o receptor utilizam a mesma chave para realizar a cifragem e deciframento da mensagem respectivamente. O tamanho da chave empregada no processo de cifragem é fundamental para a segurança final do sistema, pois o método mais simples de ataque a uma cifra é conhecido como "força bruta"e consiste em simplesmente testar todas as possíveis chaves existentes no universo de chaves possíveis. Dessa forma um sistema criptográfico com chaves de 128 bits, por exemplo, possui um conjunto de 2^{128} possíveis chaves. O esquema geral de criptografia simétrica é apresentado na Fig. 2.1

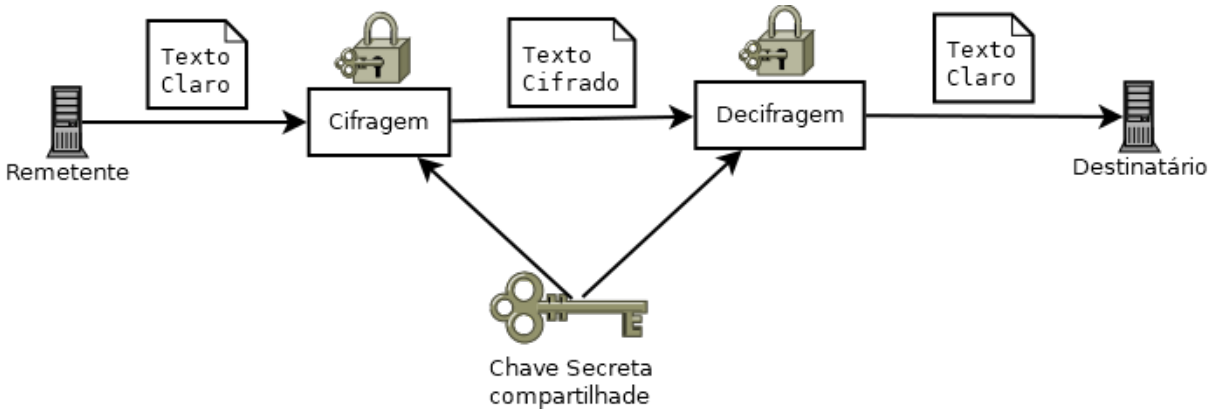

FIG. 2.1: esquema de criptografia simétrica

Um dos sistemas criptográficos de chave simétrica amplamente utilizado foi o DES (*Data Encryption Standard*) , desenvolvido no início da década de 1970 pela IBM e baseado numa estrutura de Feistel. Essa estrutura propiciava os elementos desejados em um bom sistema criptográfico: Confusão e Difusão. O esquema dessa estrutura está apresentado na Fig. 2.2 (F é a função que inclui permutação e substituição, gerando a confusão e difusão;IP e FP são permutações inversas entre si).

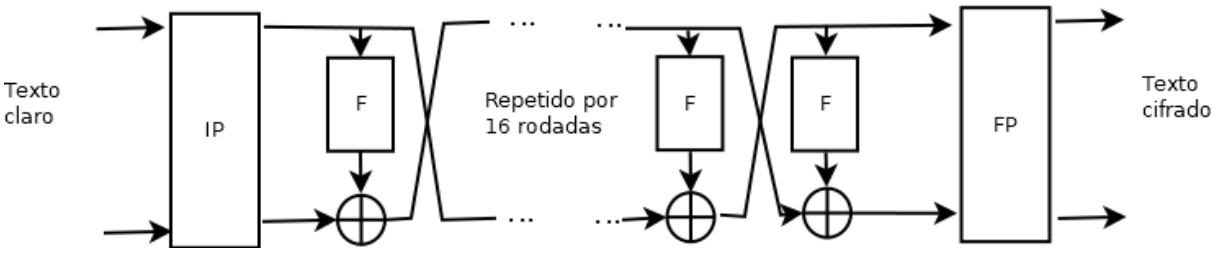

FIG. 2.2: esquema da estrutura Feistel

Durante mais de duas décadas o DES foi amplamente utilizado como padrão criptográfico, mas a sua chave foi considerada de pequeno tamanho (56 bits) para os padrões computacionais do final da década de 1990. Uma tentativa de diminuir esse problema foi a criação do triplo DES (3DES) sistema no qual a chave consiste de chaves DES concatenadas, e o processo de cifragem consiste em cifrar com uma parte da chave (k1), decifrar com outra parte (k2) e por fim cifrar com a terceira parte da chave (k3). As opções de chave são:

a) Opção de chave 1: k1, k2 e k3 distintas.

b) Opção de chave 2: k1=k3 e k2 distinta. nesse modo a chave possui 128 bits.

c) Opção de chave 2: k1=k2=k3. Nessa opção o 3DES funciona como um DES simples, embora com maior tempo de computação.

Embora o uso das opções 1 e 2 aumentem a segurança contra ataques de força bruta, o custo computacional para realizar três vezes o processo DES de cifragem/decifragem para cada bloco a ser cifrado manteve a necessidade de um novo algoritmo para ser o padrão. Nesta dissertação será elencado o 3DES com opção de chave 2 como um dos algoritmos de estudo. O esquema utilizado está apresentado na Fig. 2.3.

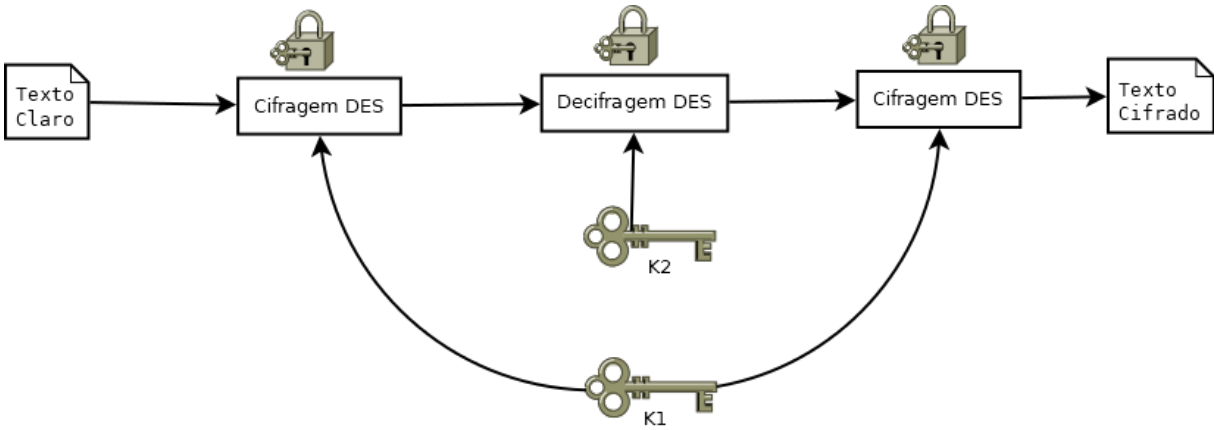

FIG. 2.3: esquema 3DES com opção de chave 2

Durante o período de 1997 a 2000, o NIST (*National Institute of Standards and Technology*) realizou um concurso para criar o novo padrão criptográfico que seria conhecido como AES (*Advanced Encryption Standard*) e que serviria para substituir o DES. O vencedor do concurso foi a cifra Rijndael, que atualmente é referenciada simplesmente como AES. Seu esquema de cifragem com chave de 128 bits está apresentado na Fig. 2.4. Além disso, demais algoritmos de cifragem foram finalistas nesse concurso: Serpent, Twofish, RC6 e MARS. Esse conjunto de algoritmos é conhecido como "finalistas do AES"e, dentre esses, foram elencados para estudo nesta dissertação: AES, Serpent, Twofish e RC6. De maneira geral todos esses algoritmos e o 3DES utilizarão sua versão com chave de 128 bits e o modo de operação ECB (Electronic CodeBook).

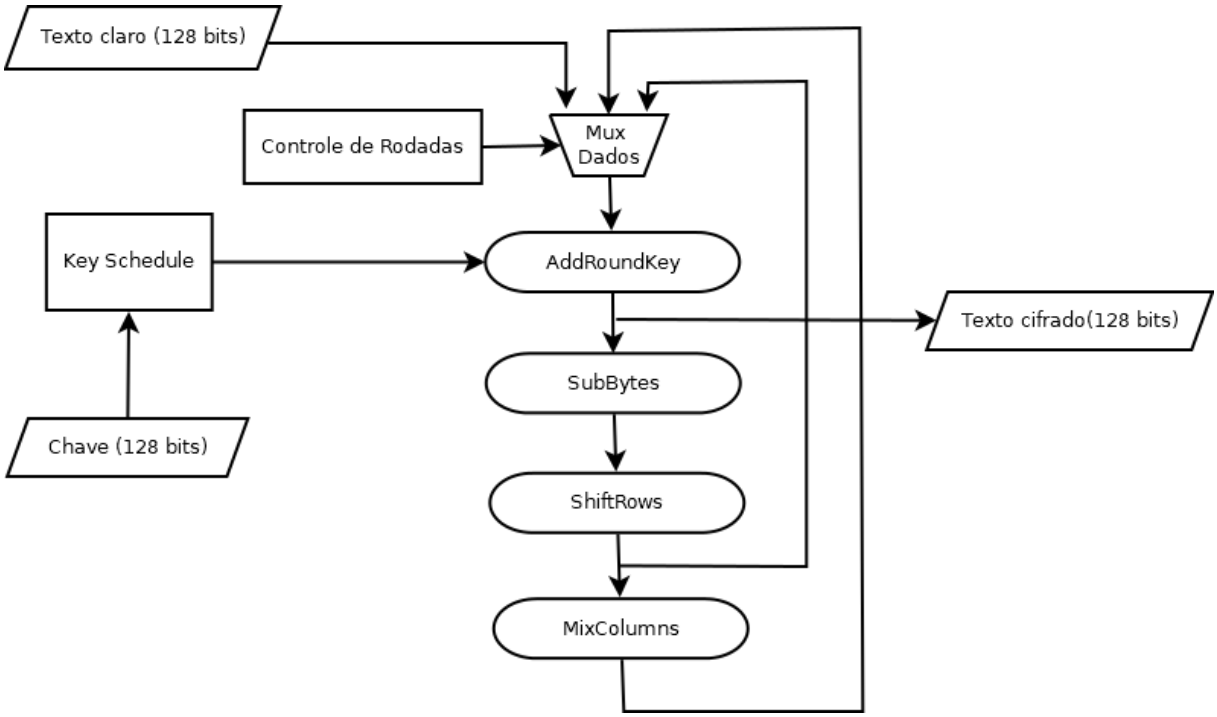

FIG. 2.4: esquema de cifragem AES (128 bit)

2.1.1 MODO DE OPERAÇÃO ECB

Os sistemas criptográficos apresentados operam dividindo as informações a serem criptografadas em blocos de informação de tamanho fixo. Os algoritmos avaliados neste trabalho operam com blocos de tamanho de 64 bits. Os algoritmos podem lidar com esses blocos de diversas maneiras, conhecidas como modos de operação. O modo mais simples é chamado de ECB, onde cada bloco é cifrado através do uso da chave de maneira independente. Dessa forma é possível realizar de forma paralelizada a cifragem desses blocos. O esquema de cifragem nesse modo de operação é apresentado na Fig. 2.5.

Embora simples, esse modo de operação tem algumas peculiaridades. Quando o tamanho do texto a ser criptografado tem tamanho não múltiplo do tamanho do bloco de operação se faz necessário o uso de "*padding*"(ou completamento) do texto para se obter um último bloco válido. Além disso a presença de redundância nos textos em claro se propaga ao texto cifrado, e isto permite o surgimento de similaridades entre textos cifrados por um mesmo par {algoritmo,chave}. Por esse motivo os algoritmos estudados nesta dissertação estarão de forma geral sendo executados em modo de operação ECB.

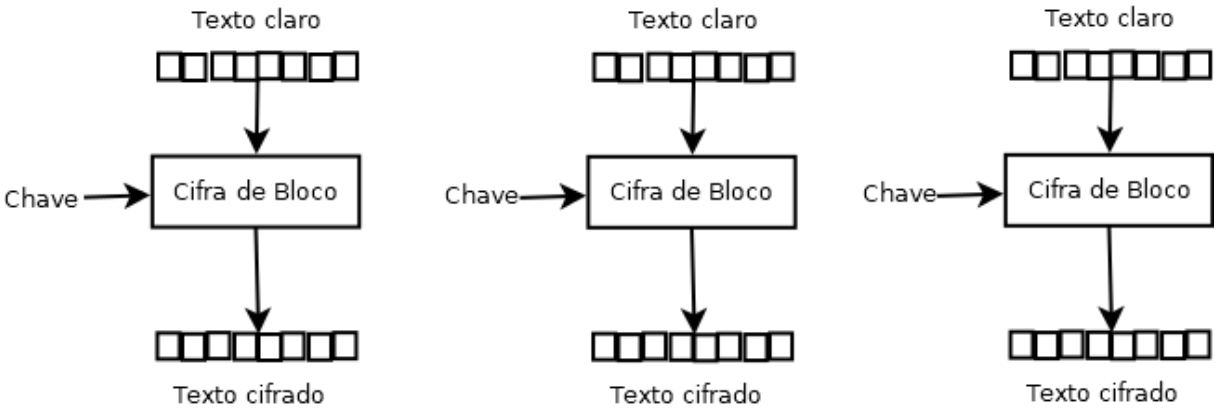

FIG. 2.5: cifragem em modo de operação ECB

Embora o modo de operação ECB ofereça vulnerabilidades e propagação de informações de redundância ao texto cifrado, a sua simplicidade de operação e possibilidade de paralelização fazem com que seja amplamente utilizada, e por isto, seu estudo se justifica.

2.1.2 DEMAIS MODOS DE OPERAÇÃO

O modo de operação *Cipher Block Chaining* (CBC) adiciona o bloco anterior cifrado adicionado via XOR ao texto em claro do bloco atual antes do processo de cifragem de cada bloco. Para o primeiro bloco, o XOR é feito com um vetor de inicialização (IV) que deverá ser transmitido junto com o texto cifrado para ser possível a deciframento. O esquema de cifragem está apresentado na Fig.2.6.

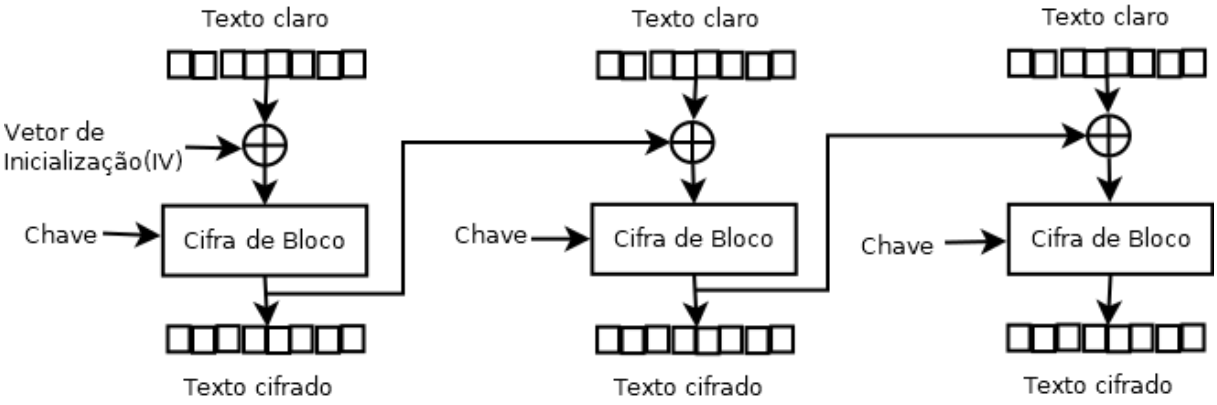

FIG. 2.6: Cifragem em modo de operação CBC

Já o modo de operação *Counter* (CTR) utiliza um valor aleatório de uso único (*Nonce*) ou um IV junto com um contador incremental como entrada do algoritmo de cifragem, que é posteriormente somado ao texto em claro (com um XOR por exemplo). Dessa forma cada bloco terá uma entrada com pequenas diferenças, introduzidas pela contagem do contador, que evitarão a repetição de blocos cifrados, mesmo com bloco em claro idênticos. Esse modo permite a paralelização da cifragem ou deciframento, pois é possível fazer a conta a priori desde que se saiba a posição do bloco a ser cifrado. O esquema de cifragem é apresentado na Fig.2.7.

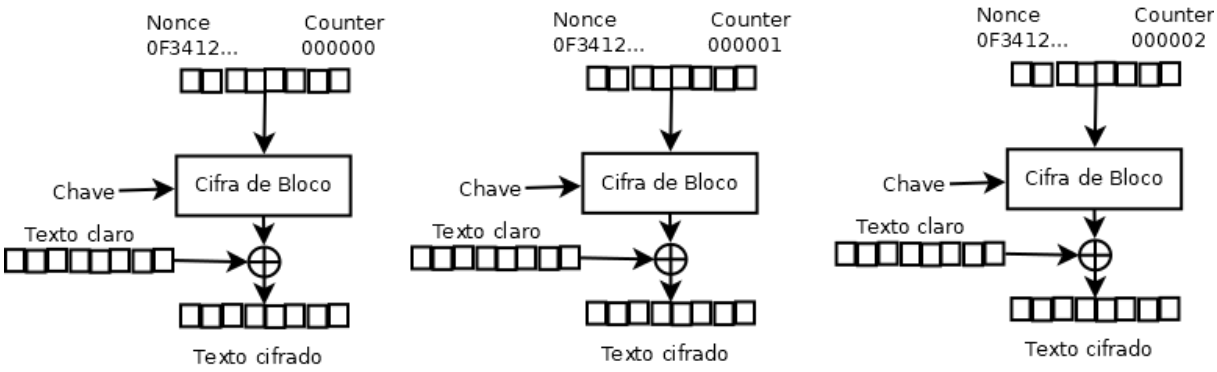

FIG. 2.7: Cifragem em modo de operação CTR

Esses modos de operação ocultam as redundâncias do texto em claro nos criptogramas gerados, aumentando a entropia das cifras resultantes. Por esse motivo os estudos desta dissertação focarão no modo ECB. No capítulo 5 se faz uma análise do uso dos métodos estudados no caso de cifras em modo de operação CBC.

2.2 TÉCNICAS DE RECUPERAÇÃO DE INFORMAÇÃO

Os sistemas de Recuperação de Informação (RI) podem ser classificados conforme algumas propriedades. Baseados nos estudos apresentados em (SOUZA, 2007) este trabalho utilizará um sistema baseado no Modelo de espaço de vetores. Nesse modelo cada documento é representado por um vetor n-dimensional , onde n é o número de palavras distintas que aparecem no texto. Além disso, quando lidar com documentos cifrados, o conceito de palavra se torna uma sequência específica de bits (como exemplo: palavras de 64 bits em representação hexadecimal FF43 , 037F).

Por exemplo, se o documento 1 contiver as palavras Planeta, Satélite, Estrela enquanto o documento 2 contiver apenas Estrela e o documento 3 as palavras Planeta, Satélite, a representação desses documentos no modelo vetorial em uso seria:

a) Documento 1: (1,1,1)

b) Documento 2: (0,0,1)

c) Documento 3: (1,1,0)

Para realizar o agrupamento das cifras deve-se calcular a similaridade entre dois documentos cifrados. Para esse propósito se faz necessário o uso de uma medida de similaridade entre dois documentos. A escolhida foi a similaridade do cosseno entre os dois vetores, novamente baseando-se nos bons resultados obtidos e apresentados por (SOUZA, 2007). Essa medida é calculada pela divisão do produto interno entre dois vetores pelo produto dos módulos desses vetores. Quando a similaridade cosseno é nula, não houve repetição de termos entre os dois documentos. Quanto mais próxima de 1 é a medida, maior repetição de termos ocorreu entre os dois documentos. A fórmula para cálculo de similaridade cosseno é apresentada em 2.1.

$$cos(A, B) = \frac{A \bullet B}{\parallel A \parallel \parallel B \parallel} \tag{2.1}$$

Continuando o exemplo anterior, o cálculo da similaridade através da distância cosseno teria como resultado:

a) Similaridade entre documentos 1 e 2: 0,333

b) Similaridade entre documentos 1 e 3: 0,667

c) Similaridade entre documentos 2 e 3: 0,0

Munido das ferramentas para calcular a similaridade entre dois documentos é possível então calcular a matriz de similaridade entre os documentos. Essa matriz é simétrica, e portanto faz-se necessário calcular apenas a matriz triangular superior ou inferior. Além disso a diagonal principal é trivialmente calculada com similaridade 1. Um exemplo dessa matriz é apresentado na Tabela 2.1. O esquema de cálculo de similaridade está apresentado na Fig.2.8.

Documentos	1	2	3	4
1	1.000	0.190	0	0.450
2	0.190	1.000	0.663	0
3	0	0.663	1.000	0
4	0.450	0	0	1.000

TAB. 2.1: Exemplo de Matriz de Similaridade

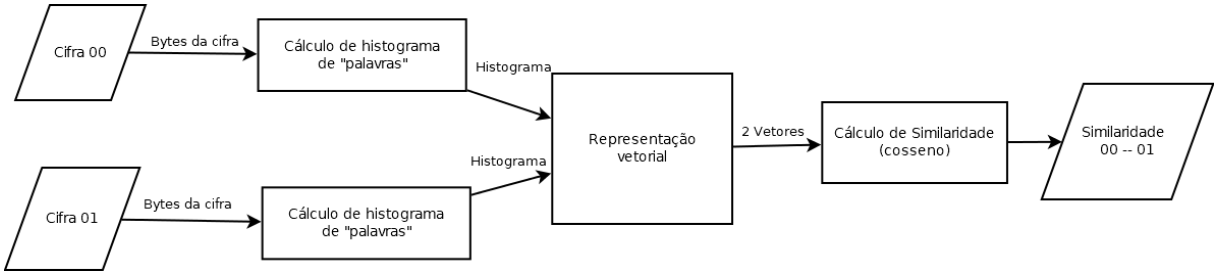

FIG. 2.8: cálculo de similaridade para cada par de documentos

2.3 TRANSFORMADAS WAVELET

Uma transformada é um operador linear que altera o domínio da função sobre a qual é executada. Sinais digitais podem ser interpretados como funções de amplitude no domínio do tempo. Quando a transformada é definida por uma integral, é chamada transformada integral. Esse é o caso da transformada *Wavelet*. A equação geral é apresentada em 2.2, com Tf e f representando funções de saída e entrada respectivamente, e a função K de duas variáveis chamada de *Kernel*, é o núcleo da transformada integral e determina o comportamento da mesma..

$$(Tf)(u) = \int_{t_1}^{t_2} K(t,u)\, f(t)\, dt \tag{2.2}$$

A *wavelet* é uma forma de onda de curta duração cujo valor médio de amplitude é zero. A utilização de uma *wavelet* como núcleo em uma transformada integral do sinal é

chamada de transformada *Wavelet*. Para utilizar essa transformada, uma *wavelet* original (chamada *wavelet* mãe) é usada e alterada através de dois parâmetros:

a) escala (a) : que permite alterar a largura da *Wavelet* mãe

b) deslocamento (b): que permite ajustar a *wavelet* mãe no eixo do domínio.

A fórmula final da transformada *wavelet* contínua é apresentada em 2.3 (Ψ representa a função *wavelet* mãe).

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} \overline{\Psi(\frac{t-b}{a})}\, x(t)\, dt \tag{2.3}$$

2.3.1 FUNÇÕES WAVELET MÃE

Algumas funções distintas podem ser usadas como *wavelet* mãe, todas devem apresentar suporte compacto (serem de curta duração) e amplitude média nula. Na Fig. 2.9 estão exemplos de *wavelets* amplamente utilizadas.

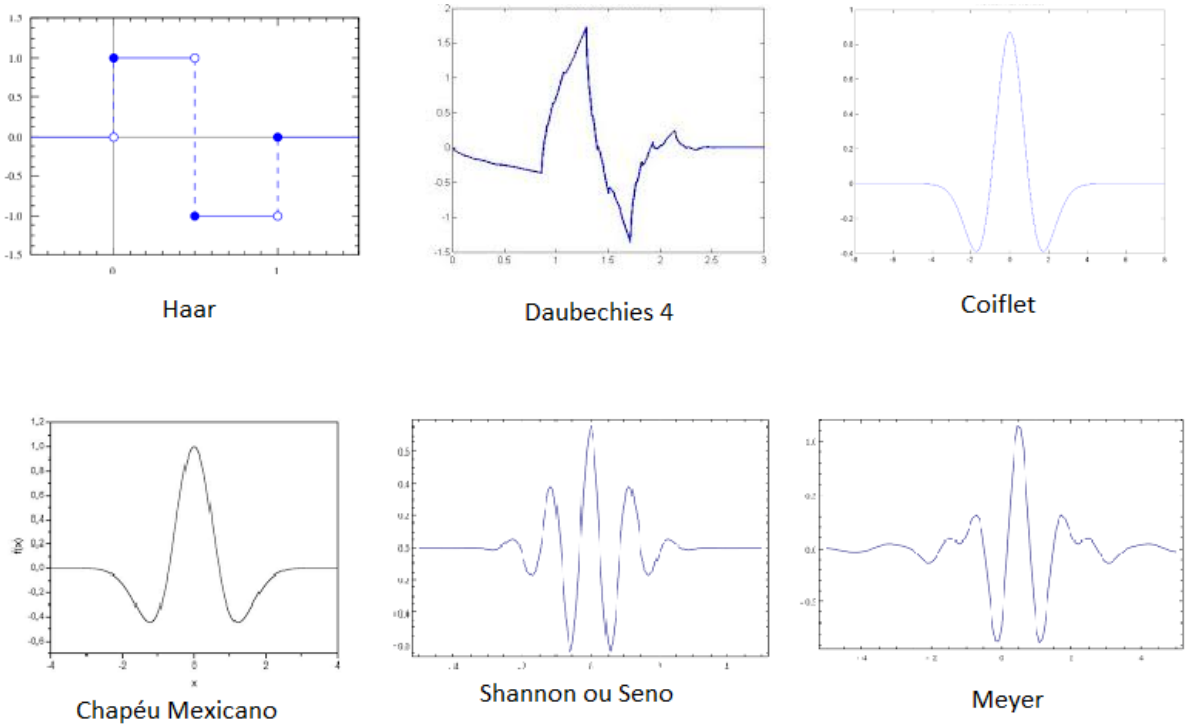

FIG. 2.9: exemplo de *wavelets* mãe

Dentre as funções apresentadas destacam-se a *wavelet* Haar, que é a mais simples delas, de fácil implementação e uso, constitui base ortogonal e serve para detecção de bordas (diferenças abruptas) em sinais. Também se destaca a *wavelet* Daubechies 4, que

também constitui base ortogonal e é robusta para identificar sinais com características mistas no tempo e frequência. O uso de *wavelets* que formam bases ortogonais permite a reconstrução perfeita do sinal através da transformada inversa.

Essas duas *wavelets* serão as mais utilizadas como núcleos de transformada neste trabalho pela sua simplicidade e por permitirem a configuração de bases ortogonais.

2.3.2 ANÁLISE MULTIRRESOLUÇÃO

A decomposição de um sinal pode ser feita utilizando-se filtros passa-baixa (que excluem as frequências mais altas)e filtros passa-alta (que excluem as frequências mais baixas). Na transformada *wavelet* discreta essa divisão feita pelo uso de dois filtros cria dois componentes que representam o sinal. A parte obtida pelo filtro passa-baixa é chamada de Aproximação, e guarda a informação mais genérica do sinal, já a parte obtida pelo filtro passa-alta é chamado de Detalhes, e representa, como o nome indica, os detalhes mais finos do sinal. O esquema dessa decomposição *Discrete Wavelet Transform* (DWT) está apresentada na Fig. 2.10 (S é o sinal original, S' o sinal reconstruído, H são filtros passa alta e L são os filtros passa baixa, D1 e A1 representam o primeiro nível de detalhes e aproximações, respectivamente).

A utilização desse tipo de transformada permite a eliminação de ruídos, compressão e reconstrução de sinais.

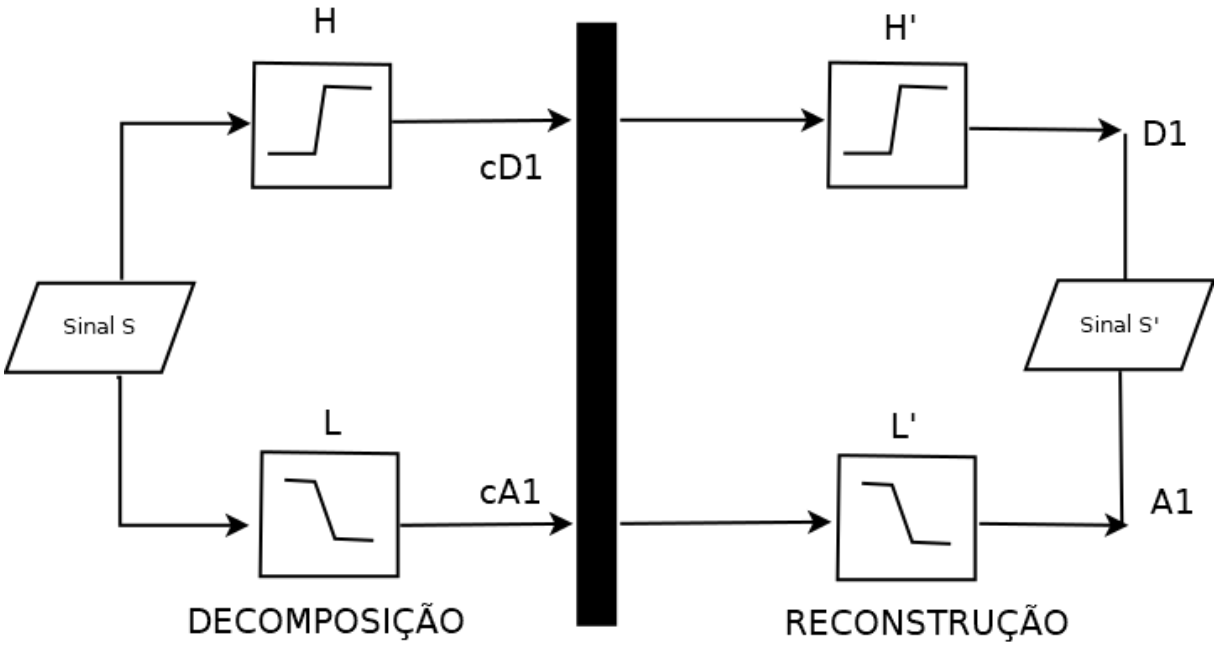

FIG. 2.10: esquema *Discrete Wavelet Transform* (DWT)

Esse processo pode ser repetido por diversas vezes, fazendo a decomposição das apro-

ximações gerando diversos níveis de resolução. Os detalhes gerados a cada nível são mantidos, enquanto as aproximações são passadas adiante para o próximo nível de decomposição. A esse processo é dado o nome de Análise Multirresolução. O uso de um banco de filtros permite a implementação desse modelo. O nível de detalhe representa por quantas vezes foi aplicada a decomposição para obter tais detalhes. O esquema para dois níveis está apresentado na 2.11.

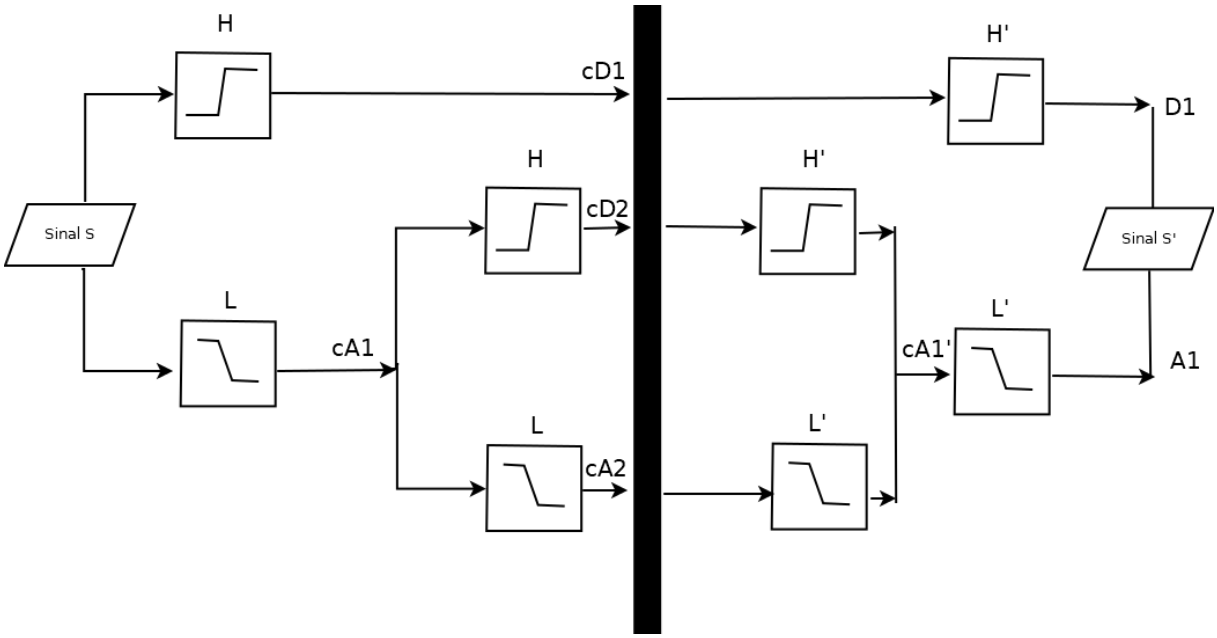

FIG. 2.11: esquema de Análise Multirresolução (MRA)

2.4 BASES VETORIAIS PARA USO COM AS TRANSFORMADAS *WAVELET*

Nesta seção são apresentados os diferentes tipos de base vetorial que serão utilizados neste trabalho (em experimentos apresentados no capítulo 5).

2.4.1 BASE TRIVIAL

A base trivial é apenas a representação das palavras na ordem crescente,(0000, 0001, 0010... 1111) e é a base utilizada nos experimentos que não envolvem o uso de *wavelets* de forma geral. O emprego dessa base nos experimentos com *wavelets* permite uma comparação direta com os experimentos feitos sem o emprego da transformada.

2.4.2 BASE PREFERENCIAL

O modelo preferencial ordena as palavras em ordem não crescente de ocorrências de cada palavra para um dos documentos e mantém essa ordem para o segundo documento (a ser

comparado). Por exemplo, as palavras 00, 01, 02 e 03 com ocorrências de 50, 66 , 35 e 16 respectivamente no documento 1 seria ordenada na forma 01, 00, 02, 03.

2.4.3 BASE WAVELETS

Nesta base vetorial as palavras são reorganizadas de acordo com quais são mais relevantes dentro de cada documento (cifra) e menos prevalentes entre os documentos (cifras) distintos. Esses valores são representados respectivamente pela *Term Frequency* (TF - frequência dos termos) e *Inverse Document Frequency* (IDF - frequência inversa de documentos). TF representa o quão relevante cada palavra é em um documento específico, e é calculada segundo a equação 2.4. (TF_{ij} é a frequência relativa do termo i no documento j e f_{ij} é a frequência absoluta do termo i no documento j).

$$TF_{\mathrm{ij}} = \frac{f_{\mathrm{ij}}}{max(f_{\mathrm{ij}})} \tag{2.4}$$

IDF representa a importância global da palavra. Quanto menos ela aparecer em múltiplos documentos, mais importante ela é. Assim o IDF é calculado segundo a equação 2.5 (N_{i} é o numero de documentos que contem o termo i, e N é o número total de documentos).

$$IDF_{\mathrm{i}} = \log_{10}(\frac{N}{Ni}) \tag{2.5}$$

Tendo calculado ambos o IDF e TF, é possível calcular o peso w_{ij} para o termo i no documento j. Isso é feito multiplicando os termos TF e IDF, conforme mostrado na equação 2.6.

$$w_{\mathrm{ij}} = IDF_{\mathrm{i}} \times TF_{\mathrm{ij}} \tag{2.6}$$

A última etapa necessária para reordenar as palavras é a Matriz Termo-Termo M_{tt} que representa a "força de ligação"entre termos. O seu cálculo é realizado pela multiplicação da matriz Termo-Documento M_{td} e a sua transposta. Este cálculo é mostrado na equação 2.7. ($\bar{w}_{\mathrm{ij}}$ é o peso normalizado para o termo i no documento j).

$$\mathbf{M}_{\mathrm{tt}} = \bar{\mathbf{M}}_{\mathrm{td}}\bar{\mathbf{M}}^{\mathrm{t}}{}_{\mathrm{td}} = \begin{bmatrix} \bar{w}_{11} & \cdots & \bar{w}_{1\mathrm{N}} \\ \vdots & \ddots & \vdots \\ \bar{w}_{\mathrm{t}1} & \cdots & \bar{w}_{\mathrm{tN}} \end{bmatrix} \begin{bmatrix} \bar{w}_{11} & \cdots & \bar{w}_{\mathrm{t}1} \\ \vdots & \ddots & \vdots \\ \bar{w}_{1\mathrm{N}} & \cdots & \bar{w}_{\mathrm{Nt}} \end{bmatrix} \tag{2.7}$$

O reordenamento dos termos é então realizado escolhendo-se primeiro um dos termos com o menor IDF, seguido do termo que possui maior valor de peso na linha atual da

Matriz Termo-Termo. Empates são resolvidos pelo menor valor de IDF entre os termos empatados, e cada termo deve aparecer apenas uma vez. Após todos os termos aparecerem exatamente uma vez, o reordenamento está completo e a base vetorial escolhida. Essa nova base é chamada de base *wavelet* e cada cifra recebida deve ser reordenada para essa base antes da aplicação das transformadas *wavelet*. Um exemplo de Matriz Termo-Termo é apresentada em 2.8. Essa matriz foi calculada a partir da coleção de documentos: Doc1(B,C,C,D,D), Doc2(B,B,C,C), Doc3(D,C,D,C), Doc4(D,A).

$$\bar{\mathbf{M}}_{\mathrm{tt}} = \begin{bmatrix} 1.00 & 0.00 & 0.00 & 0.58 \\ 0.00 & 1.00 & 0.77 & 0.26 \\ 0.00 & 0.77 & 1.00 & 0.67 \\ 0.58 & 0.26 & 0.67 & 1.00 \end{bmatrix} \tag{2.8}$$

Após a conversão para a base *wavelet*, a transformada *wavelet* é aplicada a todos os documentos da base de dados, a matriz de similaridade é calculada e a fase de treinamento é encerrada. No exemplo dado, a ordem dos termos escolhidos seria C,B,D,A. Pois C tem o menor IDF, o termo mais próximo em $\bar{\mathbf{M}}_{\mathrm{tt}}$ é B, que possui D como próximo termo, seguido de A.

2.5 CLASSIFICADOR K-NN

O algoritmo do classificador k-NN (*K Nearest Neighbor*) consiste em representar cada objeto presente na base de treinamento como um ponto no espaço definido pelos atributos, e quando se desejar classificar um objeto desconhecido, calcula-se a distância do ponto que representa este objeto (x) para cada exemplo da base de treinamento (Usualmente utilizando-se a distância euclidiana). Os k pontos mais próximos de x então participam de uma votação para determinar a classe estimada do objeto desconhecido . Esta votação, em caso de classificação, usualmente utiliza-se de um sistema de maioria simples.

Então k é um parâmetro de entrada do algoritmo e, para evitar empates, utiliza-se normalmente um valor ímpar. Um dos métodos para escolha desse parâmetro é o teste de diferentes valores de k na fase de validação, escolhendo-se o que apresenta melhor resposta ao problema estudado.

2.6 TRABALHOS RELACIONADOS

O problema de agrupamento e possível classificação de criptogramas foi estudado em trabalhos anteriores. Inicialmente no trabalho de (CARVALHO, 2006), através do uso

de técnicas de Recuperação de Informação, construiu um algoritmo capaz de aglutinar os criptogramas de acordo com as similaridades, partindo das maiores para as menores. Através do método de ligação simples, obteve-se um dendrograma através do qual é possível, determinando-se um ponto de corte, realizar o agrupamento dos criptogramas gerados por diferentes algoritmos utilizando modo de operação ECB.

Iterando sobre o trabalho anterior, (SOUZA, 2007) introduziu o uso de redes neurais auto-ajustáveis para realizar o agrupamento dos criptogramas gerados em modo ECB. Através do uso da técnica de mapa de Kohonen , a rede neural faz a separação em grupos. A novidade apresentada foi a possibilidade de fazer a tarefa de classificação, ou seja, dado um novo criptograma desconhecido, determinar a qual grupo daqueles treinados ele pertenceria. Todavia a tarefa de classificação permaneceu apenas uma possibilidade de trabalho futuro.

Continuando o trabalho de agrupamento e classificação de criptogramas, (OLIVEIRA, 2011) desenvolveu uma técnica aplicando algoritmos genéticos. Após a fase de treinamento do algoritmo genético, gerava um conjunto de criptogramas representados em um "dicionário"de histogramas. Ao receber um novo criptograma de algoritmo desconhecido, separava-o em caracteres do tamanho desejado (64 bits) e o histograma desses caracteres era comparado com os histogramas do dicionário, resultando na classificação desse criptograma. Esse método obteve bons resultados para cifras geradas em modo ECB mas com uso de apenas uma chave. Além disso acabou com a necessidade de conhecimento prévio do número de grupos existentes.

O uso de histogramas para classificação de algoritmos criptográficos foi analisado por (NAGIREDDY, 2008), obtendo bons resultados para os executados em modo ECB, mas não conseguindo resultados expressivos em modo *Cypher Block Chaining* (CBC). Além disso, neste trabalho foram testadas diversas técnicas para detecção de padrões em algoritmos.

Um outro caminho para o algoritmo de agrupamento e classificação foi pesquisado por (TORRES, 2011). A representação dos criptogramas da coleção como vértices de um grafo com as arestas representando a semelhança entre esses vértices através do seu peso permitiu a criação de um novo algoritmo que busca os subgrafos conexos. Dessa forma ocorre a separação dos grupos pelo par (criptograma,chave), gerando um processo mais eficiente. Além disso adicionou uma nova métrica de avaliação. Similar aos demais trabalhos citados, e os resultados obtidos foram significativos para o modo ECB.

Mais recentemente (TAN et al., 2016) buscou identificar os algoritmos e método de cifragem que geraram determinada cifra, utilizando Support Vector Machines (SVM) e

conseguindo resultados relevantes quando as chaves usadas na base de treino e teste eram as mesmas. Em trabalho recentemente publicado, (DE MELLO; XEXÉO, 2016) conseguem utilizar distintas técnicas de Aprendizado de Máquina e processamento massivo para realizar a classificação de cifras, incluindo os modos ECB e CBC, utilizando processo de divisão do bloco em "palavras"de tamanhos menores e acumulando os resultados. Entretanto a grande necessidade de processamento e memória ainda é uma restrição que pode se buscar superar, embora os resultados sejam bastante promissores na classificação de cifras.

Quanto ao uso de processamento de sinais digitais e *wavelets* no contexto de Recuperação de Informação pode-se citar (SILVA, 2007) que demonstrou a possibilidade do uso dessa técnica na representação dos textos e documentos para facilitar a busca e recuperação de informação. Continuando esse trabalho, (FERREIRA, 2011) realizou a comparação do uso de *wavelets* distintas no problema de busca e recuperação de informação. Esses dois trabalhos indicaram a possibilidade da aplicação de *wavelets* para o problema de detecção de padrões em criptogramas.

Em um campo não relacionado, mas que demonstra a força do uso de *Wavelets* como auxílio para resolver os problemas de agrupamento e classificação encontra-se (WANG et al., 2015) que apresenta melhorias no estado da arte na classificação de frutas através de análise de imagem com o uso de transformadas *Wavelet* sobre os componentes dessas imagens.

3 EXPERIMENTOS COM RECUPERAÇÃO DE INFORMAÇÃO

Este capítulo trata dos experimentos executados visando a analisar os usos de técnicas de Recuperação de Informação na resolução dos problemas de agrupamento e classificação de criptogramas. A base de dados empregada em todos os experimentos é descrita na seção seguinte. Posteriormente é descrita a teoria que embasa os experimentos e também é descrito o ambiente de execução dos mesmos, em termos de *hardware* e *software*. Os experimentos visaram a medir a influência do uso de diferentes tamanhos de palavras e também d uso de múltiplas chaves. A principal ideia apresentada é a divisão do criptograma em palavras de tamanho menor do que o bloco de cifragem, e o principal resultado por ela gerado foi a percepção de diferentes comportamentos das cifras geradas por 3DES ou DES.

3.1 DESCRIÇÃO DA BASE DE DADOS

A base de dados foi criada a partir da base Reuters-21578 composta de 22 arquivos de documentos contendo textos de notícias em inglês. Cada um dos documentos tem tamanho próximo a 1,4 MB, gerando uma base de textos em claro de 30MB. Esta escolha se baseou na facilidade de obtenção, tamanho da base de dados e amplo uso em pesquisa científica.

Foram então gerados conjuntos de 5 chaves distintas, um conjunto para cada algoritmo de cifragem elencado nos experimentos (AES, Serpent, Twofish, RC6,3DES e DES). Assim foram nomeadas as chaves AES1 ate AES5, Serpent1 ate Serpent5 e assim por diante. Todas essas chaves foram geradas aleatoriamente, independentemente e possuem 128 bits exceto as chaves do DES que possuem 64 bits.

Posteriormente, cada algoritmo de cifragem citado anteriormente foi executado sobre cada arquivo de texto em claro, em modo ECB, uma vez com cada chave distinta pertencente ao próprio algoritmo, gerando 660 arquivos cifrados Para fins de testes com as mesmas chaves, as 5 chaves do AES (AES1 ate AES5) foram elencadas como chaves globais e os demais algoritmos de cifragem (exceto DES) foram executados com estas chaves sobre todos os arquivos, gerando mais 440 arquivos cifrados. O processo de criação da Base de Dados é apresentado na Fig. 3.1.

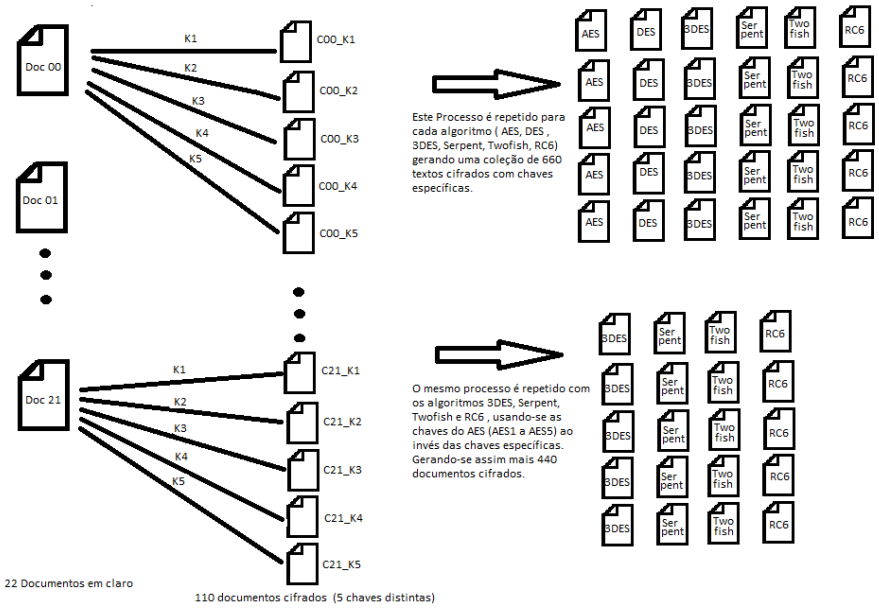

FIG. 3.1: Criação da Base de Dados

3.2 FUNDAMENTOS TEÓRICOS PARA OS EXPERIMENTOS

Os experimentos com técnicas de RI apresentados neste capítulo baseiam-se no cálculo e comparação das Matrizes de Similaridade entre cifras. Conforme apresentado na figura 2.8, o processo de cálculo de cada similaridade entre duas cifras segue as seguintes etapas:

a) Leitura dos *bytes* de ambas as cifras.

b) Cálculo do histograma de "palavras"para ambas as cifras; (o tamanho de palavra considerado é utilizado neste momento)

c) Representação de ambos os documentos no es;paço vetorial considerado (cada palavra existente em pelo menos uma das cifras gera uma dimensão);

d) Cálculo de similaridade entre os vetores usando-se a medida de similaridade desejada (no caso específico a similaridade de cosseno).

Após repetir-se este processo para cada par de cifras, obtém-se a Matriz de Similaridades desejada. O resultado de similaridade nula representa a ortogonalidade entre os vetores (não há repetição de palavras entre as cifras) e a partir de então, quanto mais próximo da unidade está a similaridade, mais próximos os vetores estão entre si (o número de aparições de cada palavra específica em ambas as cifras estão mais próximos).

3.2.1 DESCRIÇÃO DO AMBIENTE DE EXPERIMENTO

Os experimentos foram executados em um ambiente de *hardware* e *software* com as seguintes especificações:

a) Processador: Intel core i7 2600 3,4 GHz

b) Memória RAM : 16 GB;

c) Discos: 1 SSD 60GB (S.O.) e 1 SSD 220 GB (Dados e programas)

d) Sistema Operacional: Windows 10 Pro v. 10.0.15063

e) Java: JRE 1.8.0

Os programas foram escritos em linguagem Java utilizando-se a IDE Eclipse Neon. Cada experimento foi executado separadamente e de forma ininterrupta.

3.3 DESCRIÇÃO DO PRIMEIRO EXPERIMENTO - PALAVRAS DE 64BITS

O primeiro experimento foi feito através da implementação de um sistema de RI dos textos cifrados que calcula a matriz de similaridade entre os diferentes arquivos, utilizando como medida de similaridade a similaridade de cosseno entre os arquivos. Essa escolha se baseou nos estudos realizados por (SILVA, 2007) que demonstram os bons resultados obtidos utilizando-se essa medida. Os arquivos foram lidos considerando palavras de 64 bits e então foi gerado um dicionário contendo essas palavras para cada documento. Cada palavra então era considerada uma dimensão independente no cálculo vetorial da distância cosseno. O esquema empregado para o cálculo está apresentado na Fig. 3.2.

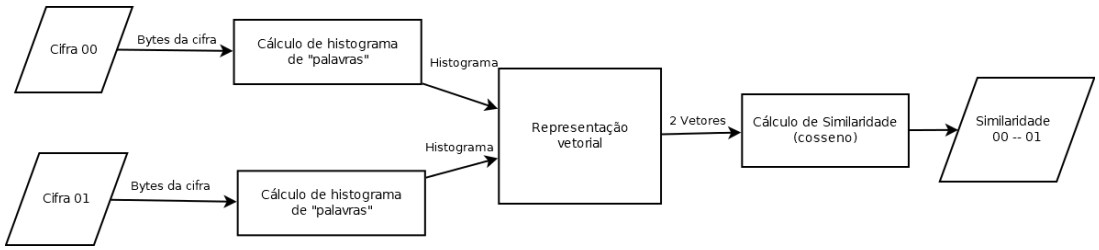

FIG. 3.2: Cálculo de Similaridade cosseno

3.3.1 PRIMEIRA ETAPA - CHAVE ÚNICA

Na primeira etapa um subconjunto dos documentos (devido ao tamanho da base de dados cifrados, apenas os arquivos 00 ate 03 cifrados com cada algoritmo foram elencados) foi

comparado usando apenas as cifras geradas com a mesma chave (AES1). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.1.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	AES1	3DES00	AES1
AES01	AES1	Twofish01	AES1	3DES01	AES1
AES02	AES1	Twofish02	AES1	3DES02	AES1
AES03	AES1	Twofish03	AES1	3DES03	AES1
Serpent00	AES1	RC6_00	AES1		
Serpent01	AES1	RC6_01	AES1		
Serpent02	AES1	RC6_02	AES1		
Serpent03	AES1	RC6_03	AES1		

TAB. 3.1: chaves empregadas em cada documento no primeiro experimento, etapa 1

A matriz de similaridade (uma matriz triangular com a diagonal principal igual a 1) foi gerada e o resultado obtido foi o seguinte:

Todos os elementos da matriz que representam documentos cifrados com o mesmo par (algoritmo, chave) apresentam alta similaridade entre seus elementos (todos acima de 0,85). Os elementos da matriz que representam documentos cifrados por algoritmos distintos (embora com a mesma chave) apresentaram similaridade nula. Esse resultado demonstra que os algoritmos distintos estão gerando palavras de 64 bits distintas mesmo com a mesma chave. Dessa forma foi possível realizar o agrupamento por algoritmo, quando usada apenas uma chave. Um extrato da matriz de similaridade é apresentado na tabela 3.2, e a tabela completa encontra-se no Apêndice 8.2.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,895	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0,897	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0,897	0	0
Twofish01						1	0	0
3DES00							1	0,948
3DES01								1

TAB. 3.2: Matriz de similaridade (extrato): Experimento 1, etapa 1 (palavra de 64 bits)

3.3.2 SEGUNDA ETAPA - CHAVES DISTINTAS POR ALGORITMO

Na segunda etapa o mesmo subconjunto anterior de documentos (arquivos 00 até 03) foram comparados, mas agora para cada algoritmo foi utilizado sua primeira chave própria (AES1 para AES, Serpent1 para Serpent e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.3.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES1	Twofish01	Twofish1	3DES01	3DES1
AES02	AES1	Twofish02	Twofish1	3DES02	3DES1
AES03	AES1	Twofish03	Twofish1	3DES03	3DES1
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent1	RC6_01	RC6_1		
Serpent02	Serpent1	RC6_02	RC6_1		
Serpent03	Serpent1	RC6_03	RC6_1		

TAB. 3.3: chaves empregadas em cada documento no primeiro experimento, etapa 2

O mesmo cálculo de matriz de similaridade foi executado e o resultado foi similar: Os grupos de documentos gerados pelo mesmo algoritmo apresentaram alta similaridade (acima de 0,85) e idênticas às obtidas na etapa anterior. Já os elementos da matriz que representam comparações entre documentos cifrados por algoritmos distintos apresentaram similaridade nula.

Novamente pode-se realizar o agrupamento por algoritmo, desde que os documentos cifrados por cada algoritmo utilizassem a mesma chave. A repetição dos resultados de similaridade mesmo com a alteração da chave para todos os algoritmos implica que a similaridade decorre da presença de redundância, do uso do modo ECB e da escolha de palavra de tamanho 64 bits. Um extrato da matriz de similaridade é apresentado na tabela 3.4, e a tabela completa encontra-se no Apêndice 8.3.

3.3.3 TERCEIRA ETAPA - CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na terceira etapa, o mesmo subconjunto anterior de documentos (arquivos 00 até 03) foram comparados, mas agora para cada algoritmo foi utilizada uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03, e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,895	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0,897	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0,897	0	0
Twofish01						1	0	0
3DES00							1	0,948
3DES01								1

TAB. 3.4: Matriz de similaridade (extrato): Experimento 1, etapa 2 (palavra de 64 bits)

chaves relativas estão apresentadas na Tabela 3.5.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 3.5: chaves empregadas em cada documento no primeiro experimento, etapa 3

O cálculo da matriz de similaridade foi feito e o resultado foi distinto dos anteriores. Cada documento apresentou apenas similaridade 1 consigo mesmo, mas similaridade nula com todos os outros documentos comparados. Dessa forma o único agrupamento obtido foi de 20 grupos distintos, cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado que o agrupamento depende da chave, pois é realizado através do par (algoritmo, chave) e não apenas de cada algoritmo, considerando-se que as chaves testadas são independentes. Um extrato da matriz de similaridade é apresentado na tabela 3.6, e a tabela completa encontra-se no Apêndice 8.4.

O tempo de execução para cada uma das etapas foi de aproximadamente 6 minutos para gerar o dicionário a partir de cada documento, e de 12 minutos por comparação entre dois documentos. O tempo total foi de 40 horas.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0	0	0
Twofish01						1	0	0
3DES00							1	0
3DES01								1

TAB. 3.6: Matriz de similaridade (extrato): Experimento 1, etapa 3 (palavra de 64 bits)

3.4 DESCRIÇÃO DO SEGUNDO EXPERIMENTO - PALAVRAS DE TAMANHOS DISTINTOS

O segundo experimento foi realizado visando a estudar a influência dos tamanhos de palavras considerados na análise de RI. Inicialmente foi escolhido um tamanho de palavra de 32 bits (divisor do tamanho do bloco de 64). Então novamente cada palavra era considerada uma dimensão independente no cálculo vetorial da similaridade cosseno.

3.4.1 PRIMEIRA ETAPA - 32 BITS, CHAVE ÚNICA

Na primeira etapa um subconjunto dos documentos (devido ao tamanho da base de dados cifrados, apenas os arquivos 00 ate 03 cifrados com cada algoritmo foram elencados) foi comparado usando apenas as cifras geradas com a mesma chave (AES1). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.7.

A matriz de similaridade foi gerada e o resultado obtido foi o seguinte:

Todos os elementos da matriz que representam documentos cifrados com o mesmo algoritmo apresentam alta similaridade entre seus elementos (todos acima de 0,85) . Os elementos da matriz que representam documentos cifrados por algoritmos distintos finalistas do AES (embora com a mesma chave) apresentaram similaridade baixa (resultados entre $2x10^{-6}$ a $5x10^{-6}$). Esse resultado demonstra que os algoritmos distintos estão gerando palavras de 32 bits distintas mesmo com a mesma chave. A existência de similaridade não nula entre documentos gerados com algoritmos distintos é explicada pela expectativa estatística de documentos do tamanho analisado gerarem palavras repetidas de forma

Documentos	Chaves	Documentos	Chaves	Documentos	Chaves
AES00	AES1	Twofish00	AES1	3DES00	AES1
AES01	AES1	Twofish01	AES1	3DES01	AES1
AES02	AES1	Twofish02	AES1	3DES02	AES1
AES03	AES1	Twofish03	AES1	3DES03	AES1
Serpent00	AES1	RC6_00	AES1		
Serpent01	AES1	RC6_01	AES1		
Serpent02	AES1	RC6_02	AES1		
Serpent03	AES1	RC6_03	AES1		

TAB. 3.7: chaves empregadas em cada documento no segundo experimento, etapa 1 (palavra de 32 bits)

independente. Dessa forma foi possível realizar o agrupamento por algoritmo, quando usada apenas uma chave.

Uma diferença expressiva foi encontrada nas cifras geradas pelo algoritmo 3DES. A similaridade entre as cifras dos finalistas do AES e as cifras geradas pelo 3DES foram aproximadamente 2 vezes menor do que entre as cifras dos finalistas do AES (resultados entre $0{,}5\text{x}10^{-6}$ a $2\text{x}10^{-6}$). Dessa forma fica possível identificar cifras geradas pelo 3DES quando comparadas com os demais finalistas do AES. Isso indica a possibilidade de criar um classificador binário do tipo "3DES/não 3DES".

Um extrato da matriz de similaridade é apresentado na tabela 3.8, e a tabela completa encontra-se no Apêndice 8.5.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,8947	$3{,}96\text{x}10^{-6}$	$3{,}18\text{x}10^{-6}$	$2{,}97\text{x}10^{-6}$	$3{,}67\text{x}10^{-6}$	$2{,}19\text{x}10^{-6}$	$1{,}37\text{x}10^{-5}$
AES01		1	$2{,}68\text{x}10^{-6}$	$3{,}37\text{x}10^{-6}$	$3{,}90\text{x}10^{-6}$	$3{,}13\text{x}10^{-6}$	$2{,}38\text{x}10^{-5}$	$2{,}96\text{x}10^{-5}$
Serpent00			1	0,8973	$3{,}22\text{x}10^{-6}$	$4{,}40\text{x}10^{-6}$	$2{,}19\text{x}10^{-6}$	$2{,}18\text{x}10^{-6}$
Serpent01				1	$3{,}67\text{x}10^{-6}$	$2{,}66\text{x}10^{-6}$	$2{,}89\text{x}10^{-6}$	$1{,}79\text{x}10^{-6}$
Twofish00					1	0,898	$3{,}22\text{x}10^{-6}$	$2{,}91\text{x}10^{-6}$
Twofish01						1	$1{,}44\text{x}10^{-6}$	$2{,}30\text{x}10^{-6}$
3DES00							1	0,948
3DES01								1

TAB. 3.8: Matriz de similaridade (extrato): Experimento 2, etapa 1 (palavra de 32 bits)

O tempo de execução para esta etapa foi de aproximadamente 23 minutos para gerar o dicionário a partir de cada documento, e de 46 minutos por cada comparação entre dois documentos. O tempo total de execução foi de 155 horas.

3.4.2 SEGUNDA ETAPA - 16 BITS, CHAVE ÚNICA

Na segunda etapa o mesmo subconjunto anterior de documentos (arquivos 00 ate 03) foi comparado, mas agora o tamanho de cada palavra considerada era de 16 bits (um quarto do tamanho do bloco). Novamente a chave escolhida foi AES1 e usada em todas as cifras. A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.7.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	AES1	3DES00	AES1
AES01	AES1	Twofish01	AES1	3DES01	AES1
AES02	AES1	Twofish02	AES1	3DES02	AES1
AES03	AES1	Twofish03	AES1	3DES03	AES1
Serpent00	AES1	RC6_00	AES1		
Serpent01	AES1	RC6_01	AES1		
Serpent02	AES1	RC6_02	AES1		
Serpent03	AES1	RC6_03	AES1		

TAB. 3.9: chaves empregadas em cada documento no segundo experimento, etapa 2 (palavra de 16 bits)

O mesmo cálculo de matriz de similaridade foi executado e o resultado foi similar: Os grupos de documentos gerados pelo mesmo algoritmo apresentaram alta similaridade (acima de 0,93), as similaridades foram ainda superiores para as cifras geradas pelo 3DES (acima de 0,95). Já os elementos da matriz que representam comparações entre documentos cifrados por algoritmos distintos, mas finalistas do AES apresentaram similaridade baixa (entre 0,40 e 0,46). De maneira análoga à etapa anterior a similaridade entre documentos cifrados pelos finalistas do AES e os documentos cifrados pelo 3DES ficou ainda mais abaixo (entre 0,25 a 0,32).

Novamente pode-se realizar o agrupamento por algoritmo, desde que os documentos cifrados por cada algoritmo utilizem a mesma chave. Novamente a diferença nos valores obtidos em relação a cifras geradas pelo 3DES sugere a possibilidade de classificação para esse sistema.

Um extrato da matriz de similaridade é apresentado na tabela 3.10, e a tabela completa encontra-se no Apêndice 8.6.

O tempo de execução nessa etapa foi reduzida, próximo a 4 minutos para gerar o dicionário a partir de cada documento, e de 8 minutos por comparação entre dois documentos. O tempo total de execução foi de 26,67 horas.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,9418	0,4594	0,4430	0,4536	0,4373	0,3167	0,3012
AES01		1	0,4556	0,4390	0,4485	0,4318	0,3140	0,2985
Serpent00			1	0,9408	0,4559	0,4383	0,3210	0,3066
Serpent01				1	0,4394	0,4222	0,3093	0,2952
Twofish00					1	0,9414	0,3183	0,3025
Twofish01						1	0,3049	0,2906
3DES00							1	0,9589
3DES01								1

TAB. 3.10: Matriz de similaridade (extrato): Experimento 2, etapa 2 (palavra de 16 bits)

3.4.3 TERCEIRA ETAPA - 16 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na terceira etapa o mesmo subconjunto anterior de documentos (arquivos 00 até 03) foi comparado, mas agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03 , e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.11.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 3.11: chaves empregadas em cada documento no primeiro experimento, etapa 3 (palavra de 16 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi análogo ao anterior. Cada documento apresentou apenas similaridade 1 consigo mesmo, e similaridade baixa com os outros documentos comparados (entre 0,40 e 0,50 para finalistas do AES e entre 0,25 a 0,32 quando comparado entre cifras finalistas do AES e 3DES). Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento.

Através dessa etapa ficou demonstrado novamente que há uma similaridade menor quando a comparação envolve cifras geradas pelo 3DES. O tempo de execução foi similar à etapa anterior.

Um extrato da matriz de similaridade é apresentado na tabela 3.12, e a tabela completa encontra-se no Apêndice 8.7.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,4362	0,4560	0,4355	0,4526	0,4378	0,3157	0,3033
AES01		1	0,44360	0,4192	0,4346	0,4176	0,3089	0,2969
Serpent00			1	0,4338	0,4574	0,4421	0,3203	0,3054
Serpent01				1	0,4351	0,4234	0,3051	0,2917
Twofish00					1	0,4387	0,3195	0,3038
Twofish01						1	0,3096	0,2872
3DES00							1	0,2141
3DES01								1

TAB. 3.12: Matriz de similaridade (extrato): Experimento 2, etapa 3 (palavra de 16 bits)

3.4.4 QUARTA ETAPA - 8 BITS, CHAVE DISTINTA POR ALGORITMO

Na quarta etapa o teste foi similar à etapa dois, mas agora o tamanho de cada palavra considerada era de 8 bits (um oitavo do tamanho do bloco). Para cada algoritmo foi utilizado sua primeira chave própria (AES1 para AES, Serpent1 para Serpent e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.13.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES1	Twofish01	Twofish1	3DES01	3DES1
AES02	AES1	Twofish02	Twofish1	3DES02	3DES1
AES03	AES1	Twofish03	Twofish1	3DES03	3DES1
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent1	RC6_01	RC6_1		
Serpent02	Serpent1	RC6_02	RC6_1		
Serpent03	Serpent1	RC6_03	RC6_1		

TAB. 3.13: chaves empregadas em cada documento no segundo experimento, etapa 4 (palavra de 8 bits)

O mesmo cálculo de matriz de similaridade foi executado e o resultado foi similar: Os grupos de documentos gerados pelo mesmo algoritmo apresentaram alta similaridade

(acima de 0,999). Já os elementos da matriz que representam comparações entre documentos cifrados por algoritmos distintos, mas finalistas do AES apresentaram similaridade mais baixa, todavia a diferença fica aparente apenas na terceira casa decimal (entre 0,997 e 0,999). De maneira análoga à etapa anterior a similaridade entre documentos cifrados pelos finalistas do AES e os documentos cifrados pelo 3DES ficou ainda mais abaixo (entre 0,994 a 0,996).

Novamente pode-se realizar o agrupamento por algoritmo, desde que os documentos cifrados por cada algoritmo utilizem a mesma chave. Mais uma vez a diferença nos valores obtidos em relação a cifras geradas pelo 3DES sugere a possibilidade de classificação para esse sistema.

Um extrato da matriz de similaridade é apresentado na tabela 3.14, e a tabela completa encontra-se no Apêndice 8.8.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,99974	0,99748	0,99723	0,99784	0,99753	0,99579	0,99533
AES01		1	0,99751	0,99728	0,99775	0,99744	0,99594	0,99554
Serpent00			1	0,99975	0,99800	0,99777	0,99596	0,99577
Serpent01				1	0,99784	0,99765	0,99600	0,99578
Twofish00					1	0,99971	0,99590	0,99549
Twofish01						1	0,99554	0,99509
3DES00							1	0,99964
3DES01								1

TAB. 3.14: Matriz de similaridade (extrato): Experimento 2, etapa 4 (palavra de 8 bits)

O tempo de execução nessa etapa foi bem reduzido, próximo a 225 milissegundos para gerar o dicionário a partir de cada documento, e de 450 milissegundos por comparação entre dois documentos. O tempo total de execução foi de 90 segundos. Devido à essa redução significativa de tempo de execução, o teste foi repetido utilizando-se a base de dados completa, e os resultados foram similares aos obtidos com a base reduzida, corroborando a validade do uso da base reduzida.

3.4.5 QUINTA ETAPA - 8 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na quinta etapa o mesmo subconjunto anterior de documentos (arquivos 00 até 03) foi comparado, mas agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento

AES02, AES4 para o documento AES03, e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 3.15.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 3.15: chaves empregadas em cada documento no primeiro experimento, etapa 5 (palavra de 8 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi análogo ao anterior. Cada documento apresentou apenas similaridade 1 consigo mesmo, e similaridade mais baixa com os outros documentos comparados, embora as diferenças ficassem aparentes na terceira casa decimal (entre 0,997 e 0,999 para finalistas do AES e entre 0,994 a 0,996 quando comparado entre cifras finalistas do AES e 3DES). Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado novamente que há uma similaridade menor quando a comparação envolve cifras geradas pelo 3DES. O tempo de execução foi similar ao da etapa anterior.

Um extrato da matriz de similaridade é apresentado na tabela 3.16, e a tabela completa encontra-se no Apêndice 8.9.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,99733	0,99748	0,99759	0,99784	0,99787	0,99579	0,99468
AES01		1	0,99736	0,99717	0,99776	0,99712	0,99565	0,99445
Serpent00			1	0,99738	0,99800	0,99731	0,99596	0,99435
Serpent01				1	0,99778	0,99747	0,99614	0,99470
Twofish00					1	0,99764	0,99590	0,99501
Twofish01						1	0,99597	0,99487
3DES00							1	0,99297
3DES01								1

TAB. 3.16: Matriz de similaridade (extrato): Experimento 2, etapa 5 (palavra de 8 bits)

3.4.6 TESTE SUPLEMENTAR - NÚMERO DE PALAVRAS DISTINTAS GERADAS POR ALGORITMO

Foi feito o seguinte como teste suplementar para conseguir descobrir os possíveis motivos dessa diferença gerada pelo 3DES.

Considerando-se uma palavra de 16 bits, o universo de palavras possíveis é de 2^{16}(65536) palavras distintas.

Ao executar a criação do dicionário para o documento 00 para os algoritmos distintos, obtiveram-se os seguintes resultados:

a) AES: 65495 palavras distintas geradas;

b) Serpent:65508 palavras distintas geradas;

c) Twofish:65489 palavras distintas geradas;

d) RC6: 65510 palavras distintas geradas;

e) 3DES: 65267 palavras distintas geradas.

Ao realizar a comparação com o documento 01 para cada algoritmo distinto, obtiveram-se os seguintes resultados acumulados:

a) AES: 65536 palavras distintas geradas.

b) Serpent:65536 palavras distintas geradas.

c) Twofish:65536 palavras distintas geradas.

d) RC6: 65536 palavras distintas geradas.

e) 3DES: 65530 palavras distintas geradas.

As diferenças para esse tamanho de documento são pequenas, mas é possível notar que o 3DES gera um menor número de palavras que os finalistas do AES (269 palavras ausentes das possíveis no primeiro documento para o 3DES, contra 41, 28, 47, 26 palavras ausentes das possíveis no primeiro documento pros finalistas AES)

4 CLASSIFICADOR BINÁRIO 3DES/DES

Este capítulo trata dos experimentos executados visando a criar um classificador binário para 3DES/DES , principal contribuição deste trabalho. A base de dados empregada em todos os experimentos foi descrita na seção 3.1 . A principal ideia apresentada é a divisão do criptograma em palavras de tamanho menor do que o bloco de cifragem, seguido do cálculo da similaridade com os documentos pré-processados da base de treino e posterior classificação entre as classes 3DES/DES ou finalista AES (não 3DES/DES).

4.1 A DIVISÃO DO BLOCO

Ao considerar-se as palavras com tamanhos divisores do bloco de 64 bits (32, 16 e 8 bits) notou-se que as cifras geradas por conjuntos distintos de (algoritmo, chave) apresentavam similaridade diferentes de zero, embora ainda muito inferiores à similaridade entre pares gerados pelo mesmo par (algoritmo, chave). A tabela 4.1 exibe os valores médios de similaridade entre os pares similares ou distintos para cada divisor. Além disso foi possível perceber que a similaridade entre documentos cifrados por um dos finalistas do concurso AES e documentos cifrados pelo 3DES ou DES apresentava valores menores, o que possibilitou a criação do classificador apresentado neste trabalho.

palavra	**(alg.,chave) iguais**	**(alg.,chave) distintos**	**"finalista AES"x3DES**
64bits	0,88	0	0
32bits	0,91	$3{,}5\text{x}10^{-6}$	$1{,}25\text{x}10^{-6}$
16bits	0,95	0,43	0,28
8bits	0,9997	0,9975	0,9950

TAB. 4.1: valor médio de similaridade

A partir da percepção de comportamento distinto das cifras geradas pelo 3DES ou DES, criou-se o classificador binário entre as classes "3DES/DES"e "finalistas do AES"(não 3DES/DES). Essa classificação é possível utilizando-se a palavra com 16 bits ou 8 bits, todavia devido a maior celeridade na execução, optou-se por prosseguir com as palavras de 8 bits. O estudo do resultado de aplicar palavras de 16 bits foi deixado para trabalhos posteriores.

4.2 CLASSIFICADOR BINÁRIO 3DES/DES

O classificador consiste em duas etapas. Na fase de treino são calculadas a matriz de similaridade entre todos os documentos considerando-se um tamanho de palavra escolhido. O modelo escolhido recebe essa matriz com as classes identificadas ("3DES/DES"ou "finalista AES"(não 3DES/DES))e realiza o treinamento. Na fase de teste o classificador recebe uma cifra sem a classe identificada e deve calcular a similaridade com os documentos da base, e aplica esses dados ao modelo treinado obtendo a provável classe. No caso da presença de classe verdadeira disponível é então verificada a correta classificação ou não dessa cifra de teste.

Em um primeiro momento o modelo de classificador testado foi o K-NN, considerando-se o tamanho de palavra de 8 bits e a base de teste foi constituída de 24 documentos apenas dentre os presentes na base de dados (4 AES,4 Serpent,4 Twofish, 4 RC6, 4 3DES e 4 DES).

4.2.1 RESULTADOS DOS TESTES

Inicialmente o modelo K-NN foi testado utilizando-se o modelo split 70-30 , sendo 70% da base usada para treinamento e o restante usado na fase de teste. O resultado foi uma acurácia de 100%, com precisões e abrangências iguais a 1, conforme demonstra a tabela 4.2.

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	5	0	1
Predito 3DES	0	2	1
Abrangência	1	1	
Acurácia	100%		

TAB. 4.2: resultado classificador split 70/30

O segundo teste desse modelo K-NN foi realizado utilizando-se a validação cruzada 10-*fold* , sendo 90% da base usada para treinamento e o restante usado na fase de teste em cada rodada. O teste é repetido por dez rodadas, usando uma parcela de 10% distinta a cada rodada (ao final das 10 rodadas cada elemento foi usado para treino 9 vezes e para teste 1 vez). O resultado foi uma acurácia de 100%, com precisões e abrangências iguais a 1, conforme demonstra a tabela 4.3.

Por fim o teste prosseguiu-se gerando uma base de testes constituída de documentos cifrados usando-se chaves pseudoaleatórias geradas pelo computador. Foram criados 78 criptogramas gerados por chaves pseudoaleatórias aplicadas sobre os documentos 7, 14,

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	16	0	1
Predito 3DES	0	8	1
Abrangência	1	1	
Acurácia	100%		

TAB. 4.3: resultado classificador validação cruzada 10-*fold*

13, 21, 10 e 3 (cada um foi sujeito a 2 chaves distintas para cada um dos 6 métodos criptográficos - AES, Serpent, Twofish, RC6, 3DES e DES) da base de dados Reuters-21578 e por fim o documento 3 foi dividido pela metade e submetido a mais uma chave pseudoaleatória para testar-se a influência da redução de tamanho. (dessa forma criando-se uma base de testes de 78 documentos cifrados). Os resultados encontram-se resumidos na tabela 4.4.

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	52	9	0,8524
Predito 3DES	0	17	1
Abrangência	1	0,6538	
Acurácia	88,46%		

TAB. 4.4: resultado classificador 5-NN utilizando base de testes com chaves pseudoaleatórias

4.3 CLASSIFICAÇÃO DE CRIPTOGRAMAS DE TAMANHOS MENORES

Posteriormente, na tentativa de se obter um classificador mais preciso e capaz de classificar textos cifrados de tamanhos menores, prosseguiu-se na criação de um modelo de classificação próprio , levando-se em conta a diferença de similaridade entre o texto a ser classificado com relação a documentos finalistas do AES e com relação a documentos cifrados pelo 3DES ou DES.

O processo incia-se com a separação do texto cifrado em blocos de 8 bits, e a contagem de cada ocorrência das distintas "palavras"de 8 bits, gerando um histograma. Esse histograma é então comparado com o histograma similar pré-calculado para cada documento da base de testes. Para poder classificar documentos de tamanhos menores, esse histograma é normalizado multiplicando-se pela relação de tamanhos entre o documento da base de dados e o documento a ser classificado. Procede-se então o cálculo de similaridade (distância cosseno) entre os dois documentos. Um caso especial pode ocorrer, onde a similaridade é alta demais o que implicaria em descoberta não só do algoritmo

mas também de qual chave foi utilizada, pois apenas o mesmo par (algoritmo,chave) geraria similaridade tão alta com o documento recebido para classificação. O esquema do classificador está apresentado na Fig. 4.1.

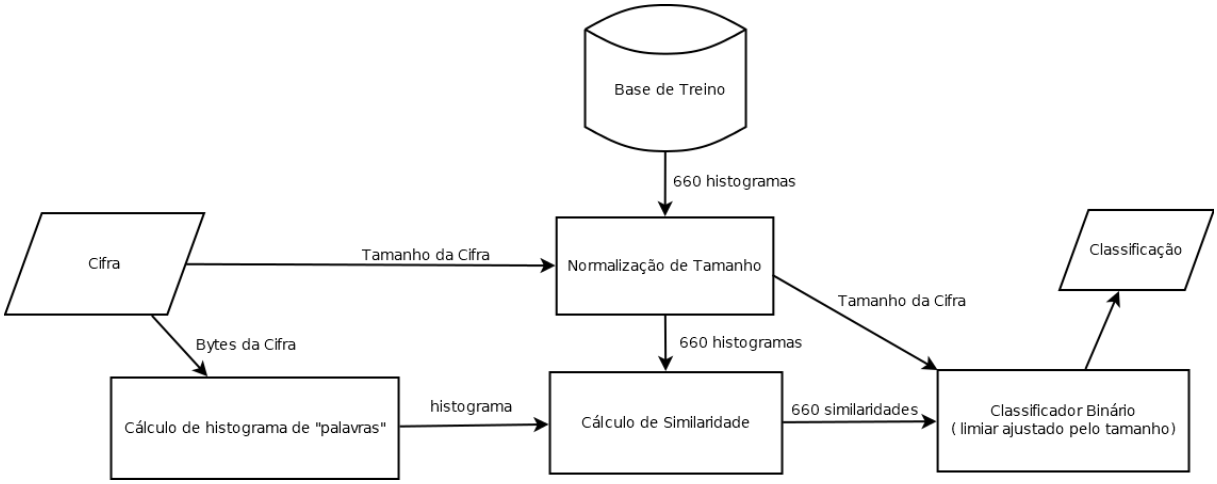

FIG. 4.1: Esquema do Classificador Binário

Munido da similaridade do documento a ser classificado com cada um dos documentos da base, procede-se o cálculo da média de similaridade da cifra desconhecida tanto com os finalistas do AES, quanto com os documentos da base gerados pelo 3DES e DES. Essas médias são usadas para escolher entre as classes 3DES/DES ou não, usando um limiar de classificação. Após múltiplas iterações, ficou evidente que o limiar que separa ambas as classes reduz-se conforme o tamanho da cifra a ser classificada também diminui. Esse efeito é mostrado na Fig. 4.2 e para melhor visualização, em escala logarítmica na Fig. 4.3. Para tentar compensar essa redução, o limiar é ajustado através de interpolação entre os dois valores mais próximos (alto e baixo, como mostrado na Equação 4.1 (L representa Limiar, e T tamanho da cifra).

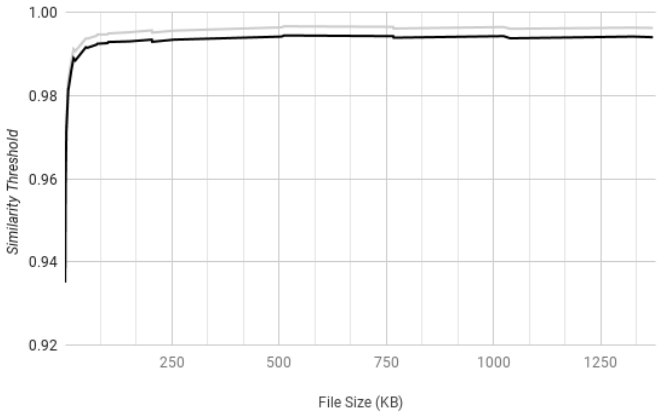

FIG. 4.2: Gráfico de Ajuste de Limiar. A linha clara mostra o limiar comparando com finalistas do AES. Linha escura mostra o limiar comparando com 3DES e DES.

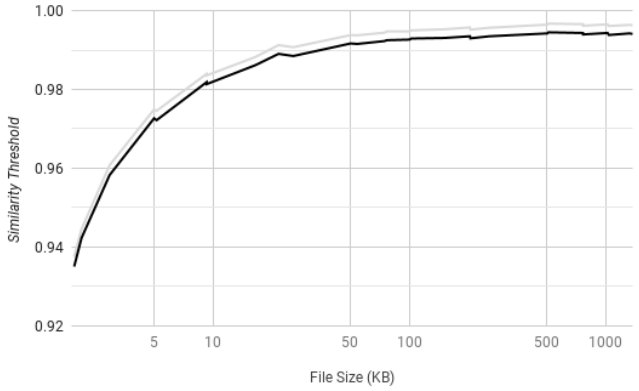

FIG. 4.3: Gráfico de Ajuste de Limiar(em escala logarítmica). A linha clara mostra o limiar comparando com finalistas do AES. Linha escura mostra o limiar comparando com 3DES e DES.

$$L_{\text{adj}} = \frac{(L_{\text{alto}} \times |T - T_{\text{baixo}}|) + (L_{\text{baixo}} \times |T - T_{\text{alto}}|)}{|T_{\text{alto}} - T_{\text{baixo}}|} \tag{4.1}$$

Como um exemplo, para uma cifra de 50000 *Bytes*, o limiar calculado para classificação comparando com finalistas do AES seria 0.9937951841 (interpolado entre os valores de 25701 *Bytes* e 50284 *Bytes*). De maneira similar, o limiar para classificação comparando com cifras 3DES/DES seria de 0.9916713248. Em caso de as duas médias gerarem classificações distintas, a escolhida será a que apresentar maior distância do limiar.

4.3.1 RESULTADOS DOS TESTES

O teste do classificador foi conduzido da seguinte forma:

a) Um dos 22 documentos em Inglês usados para a criação da base de treino foi escolhido e truncado para o tamanho desejado.

b) O texto em claro resultante foi cifrado usando os 6 algoritmos (AES, DES, 3DES, Serpent, Twofish, RC6) cada um com uma chave diferente, geradas de forma pseudo aleatória em tempo de execução.

c) Essas 6 cifras então eram submetidas ao classificador.

d) Os itens a,b e c foram repetidos, com novas chaves sendo geradas a cada repetição, até que o número desejado de tentativas era obtido para cada tamanho (6 repetições gerando 36 classificações).

e) O tamanho era então reduzido e o processo repetido (itens a,b,c e d) para cada tamanho. Os tamanhos escolhidos foram 1.3MB , 1 MB, 500KB, 200 KB, 100KB, 50 KB, 25 KB, 10 KB (aproximadamente 1000 palavras de texto).

Obteve-se acurácia total conforme a redução de tamanho até chegar a documentos cifrados com tamanho em 200 KB . O resultado para esse tamanho ainda foi de acurácia de 100%, com precisões e abrangências iguais a 1, conforme demonstra a tabela 4.5. Todavia, conforme o tamanho de texto reduziu-se a acurácia também diminuiu como mostram as tabelas 4.6 a 4.9, embora a acurácia em 100KB ainda seja de 97,22%.

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	24	0	1
Predito 3DES	0	12	1
Abrangência	1	1	
Acurácia	100%		

TAB. 4.5: resultado classificador próprio com textos em 200KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito Não 3DES	24	1	0,96
Predito 3DES	0	11	1
Abrangência	1	0,9167	
Acurácia	97,22%		

TAB. 4.6: resultado classificador próprio com textos em 100KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito Não 3DES	22	1	0,9565
Predito 3DES	2	11	0,8461
Abrangência	0,9167	0,9167	
Acurácia	91,67%		

TAB. 4.7: resultado classificador próprio com textos em 50KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito Não 3DES	22	5	0,8148
Predito 3DES	2	7	0,7777
Abrangência	0,9167	0,5833	
Acurácia	80,56%		

TAB. 4.8: resultado classificador próprio com textos em 25KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito Não 3DES	14	0	1
Predito 3DES	10	12	0,5454
Abrangência	0,5833	1	
Acurácia	72,22%		

TAB. 4.9: resultado classificador próprio com textos em 10KB

5 EXPERIMENTOS COM *WAVELETS*

Este capítulo trata dos experimentos executados visando a analisar os usos de *Wavelets* associados a técnicas de Recuperação de Informação na resolução dos problemas de agrupamento e classificação de criptogramas. A base de dados empregada em todos os experimentos foi descrita na seção 3.1 . A principal ideia apresentada é a utilização de duas *wavelets* distintas juntamente com o emprego de bases vetoriais diferentes, e o principal resultado foi a redução de espaço de armazenamento, com resultados similares aos anteriores.

5.1 MODELOS DE BASES VETORIAIS

O modo de considerar a ordem das palavras é relevante para o cálculo da transformada wavelet. A diferença entre os modelos foi apresentada na seção 2.4, entretanto aqui é apresentado um sumário.

O modelo trivial considera as palavras em ordem crescente (00000000 , 00000001 , ... 11111111). O modelo preferencial ordena as palavras em ordem não crescente de ocorrências de cada palavra para um dos documentos e mantém essa ordem para o segundo documento (a ser comparado). Por fim o modelo *wavelet* calcula os índices de relevância das palavras no conjunto total de documentos, levando em conta de forma direta a relevância intra documento e de forma inversa a relevância inter documentos, e calculando uma base única para todos os documentos através da Matriz Termo-Termo.

5.2 DESCRIÇÃO DO TERCEIRO EXPERIMENTO - MODELO TRIVIAL

O terceiro experimento foi feito visando a estudar a utilização das *wavelets* no processo já implementado de RI. Após a divisão dos textos cifrados em palavras procede-se a aplicação da transformada *wavelet*. Então novamente era realizado o cálculo vetorial da similaridade cosseno entre as cifras. O esquema empregado para o cálculo está apresentado na Fig. 5.1

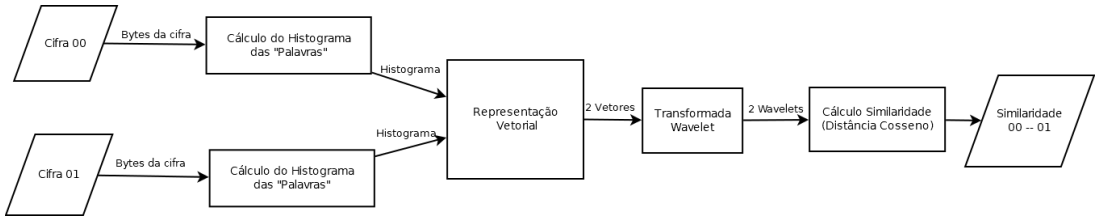

FIG. 5.1: Cálculo de Similaridade cosseno utilizando *Wavelets*

Na primeira etapa os textos foram divididos em blocos de 64 bits, o modelo trivial foi utilizado e duas *wavelets* foram testadas: Haar e Daubechies4.

5.2.1 PRIMEIRA ETAPA - 64 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Nesta etapa o subconjunto de documentos (arquivos 00 até 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03, e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.1.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.1: chaves empregadas em cada documento no terceiro experimento, etapa 1 (64 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, mas similaridade nula com todos os outros documentos comparados. Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado que o agrupamento depende da chave, pois é realizado através do par (Algoritmo, chave) e não apenas de cada algoritmo, considerando-se que as chaves testadas são independentes.

Um extrato da matriz de similaridade é apresentado na tabela 5.2, e a tabela completa encontra-se no Apêndice 8.11.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0	0	0
Twofish01						1	0	0
3DES00							1	0
3DES01								1

TAB. 5.2: Matriz de similaridade (extrato): Experimento 3, etapa 1 (palavra de 64 bits)

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 102684 segundos (aproximadamente 28,52 horas).

5.2.2 SEGUNDA ETAPA - 16 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na segunda etapa os textos foram divididos em blocos de 16 bits , o modelo trivial foi utilizado e duas *wavelets* foram testadas : Haar e Daubechies4.

Nesta etapa o subconjunto anterior de documentos (arquivos 00 ate 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03 , e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.3.

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, e similaridade baixa com os outros documentos comparados (entre 0,40 e 0,50 para finalistas do AES e entre 0,20 a 0,30 quando comparado entre cifras finalistas do AES e 3DES). Dessa forma o único agrupamento obtido foi de 20 grupos distintos, cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado novamente que há uma similaridade menor quando a comparação envolve cifras geradas pelo 3DES.

Documentos	Chaves	Documentos	Chaves	Documentos	Chaves
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.3: chaves empregadas em cada documento no terceiro experimento, etapa 2 (16 bits)

Um extrato da matriz de similaridade é apresentado na tabela 5.4, e a tabela completa encontra-se no Apêndice 8.12.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,4366	0,4598	0,4376	0,4540	0,4340	0,3169	0,3083
AES01		1	0,4561	0,4343	0,4489	0,4293	0,3142	0,3056
Serpent00			1	0,4409	0,4563	0, 4419	0,3213	0,3057
Serpent01				1	0,4398	0,4267	0,3096	0,2959
Twofish00					1	0,4357	0,3186	0,3120
Twofish01						1	0,3051	0,3006
3DES00							1	0,2153
3DES01								1

TAB. 5.4: Matriz de similaridade (extrato): Experimento 3, etapa 2 (palavra de 16 bits)

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 99386 segundos (aproximadamente 27,6 horas).

Ao final do terceiro experimento pôde-se notar que ambas as *wavelets* apresentaram resultados semelhantes entre si, e o tempo de execução foi um pouco superior ao apresentado sem uso das *wavelets*. A única vantagem obtida foi a redução no espaço de armazenamento.

5.3 DESCRIÇÃO DO QUARTO EXPERIMENTO - MODELO PREFERENCIAL

O quarto experimento foi feito visando a estudar a utilização de outra base vetorial de *wavelets*. Após a divisão dos textos cifrados em palavras procede-se a aplicação da transformada *wavelet*. Então novamente cada palavra era considerada uma dimensão indepen-

dente no cálculo vetorial da similaridade cosseno.

Na primeira etapa os textos foram divididos em blocos de 64 bits , o modelo preferencial foi utilizado e duas *wavelets* foram testadas : Haar e Daubechies4.

5.3.1 PRIMEIRA ETAPA - 64 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Nesta etapa o subconjunto anterior de documentos (arquivos 00 até 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03 , e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.5.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.5: chaves empregadas em cada documento no quarto experimento, etapa 1 (64 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, mas similaridade nula com todos os outros documentos comparados. Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado que o agrupamento depende da chave, pois é realizado através do par (Algoritmo, chave) e não apenas de cada algoritmo, considerando-se que as chaves testadas são independentes.

Um extrato da matriz de similaridade é apresentado na tabela 5.6, e a tabela completa encontra-se no Apêndice 8.13.

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 103578 segundos (aproximadamente 28,71 horas).

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0	0	0
Twofish01						1	0	0
3DES00							1	0
3DES01								1

TAB. 5.6: Matriz de similaridade (extrato): Experimento 4, etapa 1 (palavra de 64 bits)

5.3.2 SEGUNDA ETAPA - 16 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na segunda etapa os textos foram divididos em blocos de 16 bits, o modelo trivial foi utilizado e duas *wavelets* foram testadas: Haar e Daubechies4.

Nesta etapa o subconjunto anterior de documentos (arquivos 00 até 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03 , e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.7.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.7: chaves empregadas em cada documento no quarto experimento 4, etapa 2 (16 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem

uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, e similaridade baixa com os outros documentos comparados (entre 0,40 e 0,50 para finalistas do AES e entre 0,20 a 0,30 quando comparado entre cifras finalistas do AES e 3DES). Dessa maneira o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado novamente que há uma similaridade menor quando a comparação envolve cifras geradas pelo 3DES.

Um extrato da matriz de similaridade é apresentado na tabela 5.8, e a tabela completa encontra-se no Apêndice 8.14.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,4366	0,4564	0,4359	0,4530	0,4382	0,3159	0,3036
AES01		1	0,4364	0,4197	0,4350	0,4180	0,3091	0,2971
Serpent00			1	0,4342	0,4578	0,4425	0,3205	0,3056
Serpent01				1	0,4355	0,4238	0,2909	0,2919
Twofish00					1	0,4391	0,3197	0,3040
Twofish01						1	0,3098	0,2874
3DES00							1	0,2142
3DES01								1

TAB. 5.8: Matriz de similaridade (extrato): Experimento 4, etapa 2 (palavra de 16 bits)

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 99772 segundos (aproximadamente 27,75 horas).

Ao final do quarto experimento pôde-se notar que ambas as *wavelets* apresentaram resultados semelhantes entre si, e o tempo de execução foi um pouco superior ao apresentado sem uso das *wavelets*. A única vantagem obtida novamente foi a redução no espaço de armazenamento. O uso da base preferencial não trouxe benefícios evidentes em relação ao terceiro experimento.

5.4 DESCRIÇÃO DO QUINTO EXPERIMENTO - MODELO *WAVELET*

O quinto experimento foi feito visando a estudar a utilização de outra base vetorial de *wavelets*. Após a divisão dos textos cifrados em palavras procede-se a aplicação da transformada *wavelet*. Então novamente cada palavra era considerada uma dimensão independente no cálculo vetorial da similaridade cosseno.

Na primeira etapa os textos foram divididos em blocos de 64 bits, o modelo *wavelet* foi utilizado e duas *wavelets* foram testadas: Haar e Daubechies4.

5.4.1 PRIMEIRA ETAPA - 64 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Nesta etapa o subconjunto anterior de documentos (arquivos 00 até 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03, e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.9.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.9: chaves empregadas em cada documento no quinto experimento, etapa 1 (64 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, mas similaridade nula com todos os outros documentos comparados. Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado que o agrupamento depende da chave, pois é realizado através do par (Algoritmo, chave) e não apenas de cada algoritmo, considerando-se que as chaves testadas são independentes.

Um extrato da matriz de similaridade é apresentado na tabela 5.10, e a tabela completa encontra-se no Apêndice 8.15.

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 110235 segundos (aproximadamente 30,6 horas).

5.4.2 SEGUNDA ETAPA - 16 BITS, CHAVES DISTINTAS POR DOCUMENTO E ALGORITMO

Na segunda etapa os textos foram divididos em blocos de 16 bits , o modelo trivial foi utilizado e duas *wavelets* foram testadas : Haar e Daubechies4.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0	0	0	0	0	0	0
AES01		1	0	0	0	0	0	0
Serpent00			1	0	0	0	0	0
Serpent01				1	0	0	0	0
Twofish00					1	0	0	0
Twofish01						1	0	0
3DES00							1	0
3DES01								1

TAB. 5.10: Matriz de similaridade (extrato): Experimento 5, etapa 1 (palavra de 64 bits)

Nesta etapa o subconjunto anterior de documentos (arquivos 00 ate 03) foram comparados, e agora para cada algoritmo foi utilizado uma chave própria distinta (AES1 para o documento AES00, AES2 para o documento AES01, AES3 para o documento AES02, AES4 para o documento AES03 , e de forma análoga para os outros algoritmos: Serpent1 para Serpent00 e assim por diante). A relação de documentos e suas chaves relativas estão apresentadas na Tabela 5.11.

Documentos	**Chaves**	**Documentos**	**Chaves**	**Documentos**	**Chaves**
AES00	AES1	Twofish00	Twofish1	3DES00	3DES1
AES01	AES2	Twofish01	Twofish2	3DES01	3DES2
AES02	AES3	Twofish02	Twofish3	3DES02	3DES3
AES03	AES4	Twofish03	Twofish4	3DES03	3DES4
Serpent00	Serpent1	RC6_00	RC6_1		
Serpent01	Serpent2	RC6_01	RC6_2		
Serpent02	Serpent3	RC6_02	RC6_3		
Serpent03	Serpent4	RC6_03	RC6_4		

TAB. 5.11: chaves empregadas em cada documento no quinto experimento, etapa 2 (16 bits)

O cálculo da matriz de similaridade foi feito e o resultado foi similar ao obtido sem uso de *wavelets*. Cada documento apresentou apenas similaridade 1 consigo mesmo, e similaridade baixa com os outros documentos comparados (entre 0,40 e 0,50 para finalistas do AES e entre 0,20 a 0,30 quando comparado entre cifras finalistas do AES e 3DES). Dessa forma o único agrupamento obtido foi de 20 grupos distintos cada um contendo apenas um elemento. Através dessa etapa ficou demonstrado novamente que há uma

similaridade menor quando a comparação envolve cifras geradas pelo 3DES.

Um extrato da matriz de similaridade é apresentado na tabela 5.12, e a tabela completa encontra-se no Apêndice 8.16.

	AES		Serpent		Twofish		3DES	
	00	01	00	01	00	01	00	01
AES00	1	0,4371	0,4569	0,4364	0,4536	0,4387	0,3162	0,3038
AES01		1	0,4369	0,4201	0,4354	0,4184	0,3094	0,2974
Serpent00			1	0,4347	0,4583	0,4430	0,3208	0,3059
Serpent01				1	0,4360	0,4242	0,3055	0,2922
Twofish00					1	0,4396	0,3200	0,3042
Twofish01						1	0,3100	0,2876
3DES00							1	0,2143
3DES01								1

TAB. 5.12: Matriz de similaridade (extrato): Experimento 5, etapa 2 (palavra de 16 bits)

Os resultados foram similares tanto para a *wavelet* Haar quanto para a Daubechies4. O tempo de execução total foi de 101551 segundos (aproximadamente 28,2 horas).

Ao final do quinto experimento pôde-se notar que ambas as *wavelets* apresentaram resultados semelhantes entre si, e o tempo de execução foi um pouco superior ao apresentado sem uso das *wavelets*. A única vantagem obtida novamente foi a redução no espaço de armazenamento. O uso da base *wavelet* não trouxe benefícios evidentes em relação ao terceiro experimento.

5.5 EXPERIMENTO COMBINANDO CLASSIFICADOR BINÁRIO COM *WAVELETS*

Com o intuito de verificar a influência do emprego da transformada wavelet sobre o uso do classificador binário descrito no capítulo 4, um novo experimento foi realizado, aplicando-se a transformada Haar na fase do cálculo de similaridade entre a cifra de teste e a base de treino.

Neste experimento, de forma similar aos testes do classificador, 6 documentos da base de textos em inglês (14,7,3,6,17 e 9) foram truncados para o tamanho desejado, depois cifrados com os 6 algoritmos (AES,Serpent,Twofish,RC6, 3DES e DES) usando senhas diferentes geradas pseudoaleatoriamente em tempo de execução. Para cada (documento, tamanho, algoritmo) uma senha diferente foi gerada. Os documentos foram testados sem truncamento (cerca de 1,2MB de tamanho) e truncados para 500KB, 100 KB , 50 KB e 10KB. Os resultados estão apresentados nas tabelas de 5.13 até 5.17.

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	24	0	1
Predito 3DES	0	12	1
Abrangência	1	1	
Acurácia	100%		

TAB. 5.13: resultado classificador em textos sem truncamento (cerca 1.2MB)

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	24	2	0,923
Predito 3DES	0	10	1
Abrangência	1	0,833	
Acurácia	94,44%		

TAB. 5.14: resultado classificador em textos com 500 KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	24	3	0,888
Predito 3DES	0	9	1
Abrangência	1	0,75	
Acurácia	91,66%		

TAB. 5.15: resultado classificador em textos com 100 KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	20	3	0,87
Predito 3DES	4	9	0,692
Abrangência	0,833	0,75	
Acurácia	80,55%		

TAB. 5.16: resultado classificador em textos com 50 KB

	Real Não 3DES	**Real 3DES**	**Precisão**
Predito não 3DES	15	0	1
Predito 3DES	9	12	0,75
Abrangência	0,625	1	
Acurácia	75%		

TAB. 5.17: resultado classificador em textos com 50 KB

Quando comparado ao teste do classificador sem uso de transformada *wavelet*, a capacidade de classificação (representados pela acurácia, precisão e abrangência) manteve-se similar para textos acima de 200KB de tamanho. Para textos de tamanho inferior, a capacidade de classificação foi reduzida. Dessa forma, não ficou evidente nenhum ganho no uso combinado da transformada *wavelet* com o classificador binário.

6 CONSIDERAÇÕES FINAIS

6.1 CONCLUSÕES

Foi testada a aplicação de transformadas *wavelets* em conjunto a técnicas de RI para agrupamento e classificação de criptogramas, tendo sido os experimentos realizados com duas *wavelets* mãe distintas : Haar e Daubechies 4, e utilizando três modelos de base vetorial. O uso das transformadas não trouxe maior facilidade do que o uso isolado de técnicas de RI para resolver os problemas de agrupamento e classificação. A única vantagem discernível foi na redução do espaço em disco para armazenamento, entretanto devido à necessidade de aplicação da transformada, o tempo de execução foi um pouco superior ao da aplicação de técnicas de RI em situação similar de tamanho de palavra considerada e quantidade de documentos.

Através do uso de técnicas de RI sobre os criptogramas (ainda sem aplicação de transformada) foi possível iterar sobre trabalhos anteriores. Por meio da escolha de tamanhos de palavras divisoras do bloco de 64 bits conseguiu-se gerar um classificador binário para identificar os criptogramas gerados pelos algoritmos 3DES/DES como principal resultado deste trabalho.

Analisando-se o tamanho, considerado pequeno, de chave do DES (64bits , sendo apenas 56 bits efetivos) bem como o baixo uso de DES como solução, pode-se utilizar esse classificador para descobrir cifras geradas pelo 3DES, sendo possível eliminar a possibilidade do DES por força bruta se estritamente necessário.

Os testes comprovaram a viabilidade do classificador K-NN com o uso de uma base de treino de 132 documentos. Foi possível obter classificação correta para documentos incluindo uma base de testes com cifras geradas por chaves pseudoaleatórias e com acurácia de 88,46%.

Com a criação de um classificador específico buscou-se realizar a classificação de documentos cifrados de tamanho cada vez menor, tendo sido obtido classificações 100% exatas para textos em inglês cifrados contendo no mínimo 200KB. Com a redução para 100KB na cifra de teste, a acurácia ainda era excelente em 97,22%. A acurácia para tamanhos menores de cifra foi menor, entretanto ainda maior que a da escolha aleatória. A queda

em acurácia coincide com o alto decréscimo no limiar de similaridades, anteriormente mostrado.

Uma situação possível é a detecção não só do algoritmo, como da chave de cifragem. No caso da chave usada na cifra testada coincidir com alguma das chaves da base de treino, a similaridade apresentada seria muito mais alta do que o esperado, devido à igualdade do par (algoritmo, chave). Caso isto ocorra, o classificador não só descobre qual algoritmo foi utilizado mas também qual chave foi utilizada, efetivamente quebrando a cifra.

Consolidando o classificador binário com o uso de *wavelets*, foi testada uma combinação de ambos, adicionando-se a transformada *wavelet* Haar antes do cálculo de similaridade na arquitetura do classificador. Como resultado, esse manteve sua capacidade de classificação (representada pelas acurácia, precisão e abrangência) em textos de tamanho superior a 200KB, e teve resultado um pouco pior para documentos de tamanho inferior. Não ficou evidente nenhuma vantagem na combinação da transformada com o classificador binário.

6.2 LISTA DE CONTRIBUIÇÕES

As contribuições esperadas para este trabalho foram executadas conforme descrito abaixo:

a) **Análise quantitativa do uso de técnicas de RI utilizando divisores do bloco de cifragem como palavra para resolução problema de agrupamento e classificação de criptogramas, principalmente com uso de múltiplas chaves:** Foi realizada a análise quantitativa, e através das técnicas de RI foi possível realizar o agrupamento quando usada apenas uma chave, e o agrupamento e classificação binários para cifras geradas pelo 3DES/DES mesmo com o uso de múltiplas chaves, incluindo aquelas não presentes na base de treino.

b) **Análise quantitativa do uso de transformadas *wavelet* para resolução do problema de agrupamento e classificação de criptogramas, associadas ao algoritmo de cifragem e modo de operação:** Foi realizada a análise quantitativa, todavia este método não facilitou o processo de agrupamento e classificação de cifras em relação ao uso isolado de técnicas de RI, tendo apresentado ganho apenas em relação à quantidade de dados armazenados contraposto a pequeno aumento no tempo de processamento.

c) **Comparação do uso de diferentes *wavelets* para a resolução do problema de agrupamento e classificação de criptogramas:** A análise supracitada levou

em conta dois tipos de *wavelet* mãe distintos: Haar e Daubechies4, demonstrando resultados similares para ambas.

d) **Determinação da capacidade de uso de *wavelets* para detecção de padrões gerados pelo método de cifragem utilizando a interpretação direta da cifra como um sinal:** O uso de Detecção direta de padrões não foi analisado durante este trabalho.

e) **Criação de uma base de dados de cifras geradas com múltiplos algoritmos aplicados sobre textos em claro de uma base de textos de fácil acesso e amplo emprego:** Foi criada uma base de dados de cifras geradas com múltiplos algoritmos aplicados em modo ECB sobre textos em claro de uma base de textos de fácil acesso e amplo emprego contendo 22 documentos em inglês . A base criada contém cifras geradas com 5 chaves distintas para cada algoritmo e 6 algoritmos de cifragem (AES, Serpent, Twofish, RC6, 3DES e DES).

6.3 TRABALHOS FUTUROS

Considerando os resultados obtidos neste trabalho, evidencia-se a possibilidade de buscar em trabalhos futuros o seguinte:

a) Classificadores binários para outros algoritmos ou até um possível classificador multi-classe. A realização de um classificador binário para um dos algoritmos estudado levanta o questionamento sobre a possibilidade de produzir-se classificadores binários para outros algoritmos e uma combinação destes para gerar um classificador multi-classe.

b) Classificação de textos oriundos de outras linguagens e alfabetos, ou até cifras geradas sobre documentos não textuais, como imagens, áudio ou vídeo. A base de treino utilizada neste trabalho foi toda gerada por documentos de texto em inglês, e a redundância presente no texto gerador da cifra é relevante no cálculo da similaridade. Testes para definir o quão relevante é o tipo de documento que gera a base de treino, e se o uso de uma base de treino adequada ao tipo de documento que gera a cifra desconhecida é fundamental para o agrupamento e classificação de criptogramas.

c) Determinação da capacidade de uso de *wavelets* para detecção de padrões gerados pelo método de cifragem utilizando a interpretação direta da cifra como um sinal.

Devido à restrições de prazo, este tipo de técnica não foi testada neste trabalho, e portanto foi deixada como possibilidade de estudos futuros.

d) Teste de outras medidas de distância junto com aplicação de *wavelets*. Durante o estudo da aplicação de *wavelets* combinada com técnicas de RI, levantou-se a possibilidade da medida utilizada (similaridade cosseno) trazer pouca informação, e portanto, o estudo de outras medidas de similaridade ou distância entre as transformadas *wavelet* pode trazer mais informações em continuação a este trabalho.

7 REFERÊNCIAS BIBLIOGRÁFICAS

CARVALHO, C. A. B. **O uso de técnicas de recuperação de informações em criptoanálise**. 2006. 79 f. Dissertação (Mestrado em Sistemas e Computação) – Instituto Militar de Engenharia, Rio de Janeiro, 2006.

CHOW, S.; EISEN, P. A.; JOHNSON, H. ; OORSCHOT, P. C. V. White-box cryptography and an aes implementation. In: ANNUAL INTERNATIONAL WORKSHOP ON SELECTED AREAS IN CRYPTOGRAPHY, 9., SAC '02, 2003., 2003. **Electronic proceedings**... London, UK, UK: Springer-Verlag, 2003, p. 250–270. Disponível em: <http://dl.acm.org/citation.cfm?id=646558.694920>. Acesso em: 1 nov. de 2017.

DE MELLO, F.; XEXÉO, J. Cryptographic algorithm identification using machine learning and massive processing. **IEEE Latin America Transactions**, v. 14, p. 24585–4590, 2016.

FERREIRA, F. R. S. **Avaliação da qualidade do uso de wavelets para recuperação, classificação e agrupamento da informação textual**. 2011. 117 f. Dissertação (Mestrado em Sistemas e Computação) – Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2011.

KERCKHOFFS, A. La cryptographie militaire. **Journal des sciences militaires**, v. IX, p. 5–83,161–191, 1883.

NAGIREDDY, S. **A pattern recognition approach to block cipher identification**. 2008. 85 f. Dissertação (Master of Science) – Indian Institute of Technology Madras, Madras, 2008.

OLIVEIRA, C.; XEXÉO, J. A. ; CARVALHO, C. A. B. Clustering and categorization applied to cryptanalysis. **Cryptologia**, v. 30, n. 3, p. 266–280, 2006.

OLIVEIRA, G. A. **A aplicação de algoritmos genéticos no reconhecimento de padrões criptográficos**. 2011. 94 f. Dissertação (Mestrado em Sistemas e Computação) – Instituto Militar de Engenharia, Rio de Janeiro, 2011.

SHANNON, C. E. A mathematical theory of communication. **The Bell System Technical Journal**, v. 27, p. 379–423,623–656, 1948.

SHANNON, C. E. Communication theory of secrecy systems. **Bell Labs Technical Journal**, v. 28, p. 656–715, 1949.

SILVA, R. L. S. **Modelo de sinais para busca e recuperação de informação textual**. 2007. 127 f. Dissertação (Mestrado em Sistemas e Computação) – Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2007.

SOUZA, W. A. R. **Identificação de padrões em criptogramas usando técnicas de classificação de textos**. 2007. 252 f. Dissertação (Mestrado em Sistemas e Computação) – Instituto Militar de Engenharia, Rio de Janeiro, 2007.

SOUZA, W. A. R.; DE CARVALHO, L. A. V. ; XEXÉO, J. A. M. Identification of n block ciphers.. **IEEE LATIN AMERICA TRANSACTIONS**, v. 9, n. 2, p. 184–191, 2011.

TAN, C.; LI, Y. ; YAO, S. A novel identification approach to encryption mode of block cipher.. **Advances in Intelligent Systems Research**, v. 136, p. 586–591, 2016.

TORRES, R. H. **Desenvolvimento e análise de funções criptográficas para otimização dos padrões de dispersão em criptogramas**. 2011. 117 f. Dissertação (Mestrado em Sistemas e Computação) – Instituto Militar de Engenharia, Rio de Janeiro, 2011.

TORRES, R. H.; OLIVEIRA, G. A. Identification of keys and cryptographic algorithms using genetic algorithm and graph theory. **IEEE LATIN AMERICA TRANSACTIONS**, v. 9, n. 2, p. 178–183, 2011.

WANG, S.; ZHANG, Y.; JI, G.; YANG, J.; WU, J. ; WEI, L. Fruit classification by wavelet-entropy and feedforward neural network trained by fitness-scaled chaotic abc and biogeography-based optimization. **Entropy**, v. 17, p. 5711–5728, 2015.

8 APÊNDICES

APÊNDICE 1: MATRIZES DE SIMILARIDADE DE RI

8.2 PALAVRA DE 64 BITS, CHAVE ÚNICA

		AES	K_AES1			Serpent	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	1	0,8946300145	0,9010329619	0,8871599759	0	0	0	0
K_AES1	01		1	0,9049386678	0,8893815486	0	0	0	0
	02			1	0,8920828822	0	0	0	0
	03				1	0	0	0	0
Serpent	00					1	0,8973816968	0,9010329619	0,887159975856458
K_AES1	01						1	0,9077220572	0,892117087788709
	02							1	0,8920828822456
	03								1
Twofish	00								
K_AES1	01								
	02								
	03								
RC6	00								
K_AES1	01								
	02								
	03								
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.1: Matriz de Similaridade - palavra de 64 bits, chave única

		Twofish	K_AES1			RC6	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	0	0	0	0	0	0	0	0
K_AES1	01	0	0	0	0	0	0	0	0
	02	0	0	0	0	0	0	0	0
	03	0	0	0	0	0	0	0	0
Serpent	00	0	0	0	0	0	0	0	0
K_AES1	01	0	0	0	0	0	0	0	0
	02	0	0	0	0	0	0	0	0
	03	0	0	0	0	0	0	0	0
Twofish	00	1	0,8973816968	0,9010329619	0,8871599759	0	0	0	0
K_AES1	01		1	0,9077220572	0,8921170878	0	0	0	0
	02			1	0,8920828822	0	0	0	0
	03				1	0	0	0	0
RC6	00					1	0,8973816968	0,9010329619	0,8871599759
K_AES1	01						1	0,9077220572	0,8921170878
	02							1	0,8920828822
	03								1
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.2: Matriz de Similaridade - palavra de 64 bits, chave única (continuada)

		3DES	K_AES1		
	Doc	00	01	02	03
AES	00	0	0	0	0
K_AES1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
Serpent	00	0	0	0	0
K_AES1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
Twofish	00	0	0	0	0
K_AES1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
RC6	00	0	0	0	0
K_AES1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
3DES	00	1	0,9478995182	0,946556217	0,940835115
K_AES1	01		1	0,9533666246	0,9429018393
	02			1	0,9443847277
	03				1

TAB. 8.3: Matriz de Similaridade - palavra de 64 bits, chave única (continuada)

		AES	K1			Serpent	K1		
	Doc	00	01	02	03	00	01	02	03
AES	00	1	0,8946300145	0,9010329619	0,8871599759	0	0	0	0
K1	01		1	0,9049386678	0,8893815486	0	0	0	0
	02			1	0,8920828822	0	0	0	0
	03				1	0	0	0	0
Serpent	00					1	0,8973816968	0,9010329619	0,887159975856458
K1	01						1	0,9077220572	0,892117087788709
	02							1	0,8920828822456
	03								1
Twofish	00								
K1	01								
	02								
	03								
RC6	00								
K1	01								
	02								
	03								
3DES	00								
K1	01								
	02								
	03								

TAB. 8.4: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo

		Twofish	K1			RC6	K1		
	Doc	00	01	02	03	00	01	02	03
AES	00	0	0	0	0	0	0	0	0
K1	01	0	0	0	0	0	0	0	0
	02	0	0	0	0	0	0	0	0
	03	0	0	0	0	0	0	0	0
Serpent	00	0	0	0	0	0	0	0	0
K1	01	0	0	0	0	0	0	0	0
	02	0	0	0	0	0	0	0	0
	03	0	0	0	0	0	0	0	0
Twofish	00	1	0,8973816968	0,9010329619	0,8871599759	0	0	0	0
K1	01		1	0,9077220572	0,8921170878	0	0	0	0
	02			1	0,8920828822	0	0	0	0
	03				1	0	0	0	0
RC6	00					1	0,8973816968	0,9010329619	0,8871599759
K1	01						1	0,9077220572	0,8921170878
	02							1	0,8920828822
	03								1
3DES	00								
K1	01								
	02								
	03								

TAB. 8.5: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo (continuada)

		3DES	K1		
	Doc	00	01	02	03
AES	00	0	0	0	0
K1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
Serpent	00	0	0	0	0
K1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
Twofish	00	0	0	0	0
K1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
RC6	00	0	0	0	0
K1	01	0	0	0	0
	02	0	0	0	0
	03	0	0	0	0
3DES	00	1	0,9478995182	0,946556217	0,940835115
K1	01		1	0,9533666246	0,9429018393
	02			1	0,9443847277
	03				1

TAB. 8.6: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo (continuada)

8.4 PALAVRA DE 64 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0	0	0	0	0	0	0
AES K2	01		1	0	0	0	0	0	0
AES K3	02			1	0	0	0	0	0
AES K4	03				1	0	0	0	0
Serpent K1	00					1	0	0	0
Serpent K2	01						1	0	0
Serpent K3	02							1	0
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.7: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0	0	0	0	0	0	0	0
AES K2	01	0	0	0	0	0	0	0	0
AES K3	02	0	0	0	0	0	0	0	0
AES K4	03	0	0	0	0	0	0	0	0
Serpent K1	00	0	0	0	0	0	0	0	0
Serpent K2	01	0	0	0	0	0	0	0	0
Serpent K3	02	0	0	0	0	0	0	0	0
Serpent K4	03	0	0	0	0	0	0	0	0
Twofish K1	00	1	0	0	0	0	0	0	0
Twofish K2	01		1	0	0	0	0	0	0
Twofish K3	02			1	0	0	0	0	0
Twofish K4	03				1	0	0	0	0
RC6 K1	00					1	0	0	0
RC6 K2	01						1	0	0
RC6 K3	02							1	0
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.8: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0	0	0	0
AES K2	01	0	0	0	0
AES K3	02	0	0	0	0
AES K4	03	0	0	0	0
Serpent K1	00	0	0	0	0
Serpent K2	01	0	0	0	0
Serpent K3	02	0	0	0	0
Serpent K4	03	0	0	0	0
Twofish K1	00	0	0	0	0
Twofish K2	01	0	0	0	0
Twofish K3	02	0	0	0	0
Twofish K4	03	0	0	0	0
RC6 K1	00	0	0	0	0
RC6 K2	01	0	0	0	0
RC6 K3	02	0	0	0	0
RC6 K4	03	0	0	0	0
3DES K1	00	1	0	0	0
3DES K2	01		1	0	0
3DES K3	02			1	0
3DES K4	03				1

TAB. 8.9: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento (continuada)

		AES	K_AES1			Serpent	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	1	0,89463	0,90104	0,887165	3,96E-006	3,18E-006	3,92E-006	3,41E-006
K_AES1	01		1	0,90493905	0,88938277	2,68E-006	3,37E-006	2,90E-006	4,09E-006
	02			1	0,89208	1,34E-005	9,45E-006	8,75E-006	9,91E-006
	03				1	3,17E-006	2,65E-006	3,38E-006	2,88E-006
Serpent	00					1	0,89738	0,901033	0,8871611
K_AES1	01						1	0,907722	0,892116
	02							1	0,892083
	03								1
Twofish	00								
K_AES1	01								
	02								
	03								
RC6	00								
K_AES1	01								
	02								
	03								
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.10: Matriz de Similaridade - palavra de 32 bits, chave única

		Twofish	K_AES1			RC6	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	2,97E-006	3,67E-006	2,45E-006	3,17E-006	4,95E-006	5,14E-006	4,17E-006	3,66E-006
K_AES1	01	3,90E-006	3,13E-006	4,11E-006	4,09E-006	3,90E-006	3,62E-006	3,87E-006	3,36E-006
	02	4,66E-006	2,91E-006	3,40E-006	1,21E-006	3,19E-006	1,94E-006	1,94E-006	2,42E-006
	03	4,88E-006	3,86E-006	3,14E-006	4,57E-006	2,93E-006	5,30E-006	3,87E-006	4,56E-006
Serpent	00	3,22E-006	4,40E-006	1,96E-006	4,39E-006	1,06E-005	3,18E-006	1,08E-005	1,24E-005
K_AES1	01	3,67E-006	2,66E-006	3,39E-006	2,65E-006	1,13E-005	3,87E-006	8,97E-006	6,03E-006
	02	2,21E-006	1,94E-006	1,22E-006	2,90E-006	1,18E-005	3,39E-006	5,59E-006	3,63E-006
	03	6,83E-006	5,30E-006	5,08E-006	5,53E-006	1,02E-005	2,17E-006	7,01E-006	4,81E-006
Twofish	00	1	0,897381	0,9010318	0,88716	2,10E-005	2,20E-006	3,68E-006	4,15E-006
K_AES1	01		1	0,90771944	0,8921142	2,40E-005	3,38E-006	1,94E-006	2,65E-006
	02			1	0,89208	2,31E-005	1,94E-006	4,62E-006	1,69E-006
	03				1	2,22E-005	4,34E-006	3,38E-006	3,85E-006
RC6	00					1	0,8973838	0,9010335	0,8871631
K_AES1	01						1	0,90772266	0,8921197
	02							1	0,892089
	03								1
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.11: Matriz de Similaridade - palavra de 32 bits, chave única (continuada)

		3DES	K_AES1		
	Doc	00	01	02	03
AES	00	2,19E-006	1,37E-005	2,78E-006	2,61E-006
K_AES1	01	2,38E-005	2,96E-005	2,12E-005	2,63E-005
	02	1,16E-006	8,37E-006	8,70E-007	1,01E-006
	03	1,30E-006	2,21E-005	2,59E-006	2,14E-006
Serpent	00	2,19E-006	2,18E-006	1,61E-006	7,24E-007
K_AES1	01	2,89E-006	1,79E-006	1,16E-006	3,01E-006
	02	1,01E-006	8,66E-007	5,80E-007	1,15E-006
	03	7,20E-007	1,29E-006	1,30E-006	1,43E-006
Twofish	00	3,22E-006	2,91E-006	1,32E-006	2,17E-006
K_AES1	01	1,44E-006	2,30E-006	1,01E-006	2,86E-006
	02	2,61E-006	1,30E-006	8,70E-007	8,62E-007
	03	8,64E-007	1,29E-006	2,02E-006	9,99E-007
RC6	00	1,90E-006	1,02E-006	1,32E-006	1,74E-006
K_AES1	01	1,73E-006	2,45E-006	2,89E-006	1,58E-006
	02	1,74E-006	3,18E-006	1,01E-006	2,73E-006
	03	1,30E-006	7,18E-007	1,01E-006	1,14E-006
3DES	00	1	0,947899727	0,946556512	0,94083549
K_AES1	01		1	0,95336701	0,942901
	02			1	0,944385
	03				1

TAB. 8.12: Matriz de Similaridade - palavra de 32 bits, chave única (continuada)

		AES	K_AES1			Serpent	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	1	0,9417766271	0,942975954	0,9369128072	0,4593349741	0,4429801488	0,4359036288	0,4515388512
K_AES1	01		1	0,9449479418	0,9376594329	0,4556176646	0,4389657241	0,4320265022	0,4471348117
	02			1	0,9370084052	0,4356782919	0,4204144119	0,4138068826	0,4285756045
	03				1	0,4505956471	0,4345480641	0,4282570956	0,4432493187
Serpent	00					1	0,9408361323	0,941979921	0,9370511959
K_AES1	01						1	0,9453690973	0,9379556284
	02							1	0,9369228233
	03								1
Twofish	00								
K_AES1	01								
	02								
	03								
RC6	00								
K_AES1	01								
	02								
	03								
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.13: Matriz de Similaridade - palavra de 16 bits, chave única

		Twofish	K_AES1			RC6	K_AES1		
	Doc	00	01	02	03	00	01	02	03
AES	00	0,4535532604	0,4372282705	0,4298278886	0,4457849656	0,4515540237	0,4370885818	0,4286509889	0,4433879217
K_AES1	01	0,4485184064	0,4317738428	0,424988946	0,4403819081	0,4479450143	0,4339790864	0,4248462556	0,4400940845
	02	0,4298382059	0,4137400699	0,4072782597	0,4224911619	0,4282527104	0,4154476079	0,4070352939	0,4211258843
	03	0,4453451133	0,4288541784	0,422140691	0,4378962266	0,443041671	0,4298434225	0,4208950478	0,4361068311
Serpent	00	0,4558901578	0,4383515472	0,4296094163	0,4465414676	0,4537883222	0,437126262	0,4304335487	0,4435163478
K_AES1	01	0,4394049195	0,4222621762	0,4133399393	0,4297730295	0,4383006451	0,4230217299	0,4158020829	0,4287312303
	02	0,4309286776	0,4143196773	0,4059144777	0,4218804441	0,4299040648	0,4148501752	0,4083868465	0,4202492824
	03	0,4477927338	0,4301528672	0,4217045012	0,4385614211	0,4450453031	0,4295439829	0,4220029209	0,4350128564
Twofish	00	1	0,941371687	0,9420310538	0,9362948386	0,4521764358	0,4348777528	0,428724847	0,4435953724
K_AES1	01		1	0,9457898283	0,9377733128	0,436051186	0,4190775926	0,4134637861	0,4274925338
	02			1	0,9366060405	0,4277119194	0,4116115202	0,4060128698	0,419622691
	03				1	0,4430443485	0,4261845835	0,4201215118	0,4347615257
RC6	00					1	0,9418262004	0,9431607547	0,9376321424
K_AES1	01						1	0,946157639	0,938811945
	02							1	0,9378278942
	03								1
3DES	00								
K_AES1	01								
	02								
	03								

TAB. 8.14: Matriz de Similaridade - palavra de 16 bits, chave única (continuada)

		3DES	K_AES1		
	Doc	00	01	02	03
AES	00	0,316638169	0,3011742741	0,2959750523	0,3104766884
K_AES1	01	0,3139430212	0,2985185122	0,2929078746	0,307809546
	02	0,3005822588	0,2858372472	0,2809660284	0,2950006256
	03	0,3109750018	0,2954137437	0,2901660084	0,3048551754
Serpent	00	0,3210574343	0,3065900424	0,3002060606	0,313586302
K_AES1	01	0,3093406806	0,2952018996	0,2890057577	0,3022402048
	02	0,3063929107	0,2926216602	0,2864000887	0,2993963338
	03	0,3156584642	0,3018784627	0,2949956769	0,3081949185
Twofish	00	0,3183387304	0,3024728054	0,2963011917	0,3095854853
K_AES1	01	0,3048634888	0,2906589891	0,2841096498	0,2969116676
	02	0,3011407749	0,2865650869	0,2801458257	0,2929816726
	03	0,311827713	0,2967057291	0,2902009161	0,303127402
RC6	00	0,3173226709	0,3038550918	0,2982728932	0,3089404197
K_AES1	01	0,3051133023	0,2921563633	0,2868744506	0,2973982353
	02	0,300196506	0,2873187574	0,2821506871	0,2925683818
	03	0,3115043929	0,2980794469	0,2924665764	0,3032482239
3DES	00	1	0,9588900551	0,9568907398	0,9536315269
K_AES1	01		1	0,9623083724	0,9546573654
	02			1	0,9554975962
	03				1

TAB. 8.15: Matriz de Similaridade - palavra de 16 bits, chave única (continuada)

8.7 PALAVRA DE 16 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0,4362	0,4276	0,4451	0,4560	0,4355	0,4308	0,4467
AES K2	01		1	0,4189	0,4308	0,4360	0,4192	0,4125	0,4314
AES K3	02			1	0,4227	0,4304	0,4128	0,4045	0,4214
AES K4	03				1	0,4435	0,4283	0,4226	0,4350
Serpent K1	00					1	0,4338	0,4296	0,4442
Serpent K2	01						1	0,4125	0,4267
Serpent K3	02							1	0,4261
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.16: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0,4526	0,4378	0,4305	0,4441	0,4555	0,4348	0,4314	0,4463
AES K2	01	0,4346	0,4176	0,4139	0,4312	0,4319	0,4196	0,4135	0,4270
AES K3	02	0,4282	0,4135	0,4091	0,4218	0,4318	0,4155	0,4047	0,4229
AES K4	03	0,4458	0,4302	0,4235	0,4387	0,4412	0,4306	0,4219	0,4389
Serpent K1	00	0,4574	0,4421	0,4307	0,4463	0,4522	0,4318	0,4280	0,4480
Serpent K2	01	0,4351	0,4234	0,4148	0,4243	0,4343	0,4205	0,4177	0,4236
Serpent K3	02	0,4291	0,4111	0,4078	0,4218	0,4297	0,4107	0,4074	0,4204
Serpent K4	03	0,4446	0,4282	0,4226	0,4389	0,4437	0,4276	0,4241	0,4369
Twofish K1	00	1	0,4387	0,4298	0,4451	0,4507	0,4349	0,4275	0,4497
Twofish K2	01		1	0,4121	0,4294	0,4349	0,4148	0,4123	0,4268
Twofish K3	02			1	0,4215	0,4258	0,4106	0,4067	0,4206
Twofish K4	03				1	0,4428	0,4305	0,4242	0,4344
RC6 K1	00					1	0,4374	0,4288	0,4461
RC6 K2	01						1	0,4132	0,4238
RC6 K3	02							1	0,4179
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.17: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0,3157	0,3033	0,2965	0,3130
AES K2	01	0,3089	0,2969	0,2888	0,3006
AES K3	02	0,3028	0,2865	0,2887	0,2954
AES K4	03	0,3089	0,2943	0,2944	0,3103
Serpent K1	00	0,3203	0,3054	0,2954	0,3089
Serpent K2	01	0,3051	0,2917	0,2872	0,2994
Serpent K3	02	0,3001	0,2846	0,2807	0,2934
Serpent K4	03	0,3122	0,2968	0,2921	0,3073
Twofish K1	00	0,3195	0,3038	0,3014	0,3115
Twofish K2	01	0,3096	0,2872	0,2860	0,3003
Twofish K3	02	0,2982	0,2926	0,2832	0,3006
Twofish K4	03	0,3119	0,2964	0,2923	0,3115
RC6 K1	00	0,3160	0,3026	0,2997	0,3087
RC6 K2	01	0,3047	0,2911	0,2861	0,3024
RC6 K3	02	0,2986	0,2887	0,2778	0,2952
RC6 K4	03	0,3081	0,2981	0,2971	0,3054
3DES K1	00	1	0,2141	0,2067	0,2232
3DES K2	01		1	0,2030	0,2170
3DES K3	02			1	0,2002
3DES K4	03				1

TAB. 8.18: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento (continuada)

		AES	K1			Serpent	K1		
	Doc	00	01	02	03	00	01	02	03
AES	00	1	0,99974703	0,99974915	0,99972679	0,99747953	0,99723341	0,99724251	0,99738994
K1	01		1	0,99974071	0,99971622	0,99751112	0,99728333	0,99724709	0,99742072
	02			1	0,99973525	0,99723696	0,99697703	0,99697084	0,99714207
	03				1	0,99752078	0,99726659	0,99725687	0,99744141
Serpent	00					1	0,99975072	0,99975663	0,99977088
K1	01						1	0,99974279	0,99970899
	02							1	0,99972628
	03								1
Twofish	00								
K1	01								
	02								
	03								
RC6	00								
K1	01								
	02								
	03								
3DES	00								
K1	01								
	02								
	03								

TAB. 8.19: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo

		Twofish	K1			RC6	K1		
	Doc	00	01	02	03	00	01	02	03
AES	00	0,99784026	0,99752756	0,99747428	0,99772859	0,99746669	0,99732610	0,99720531	0,99744022
K1	01	0,99775389	0,99744703	0,99733373	0,99760555	0,99739938	0,99720271	0,99710114	0,99735826
	02	0,99754761	0,99722332	0,99713635	0,99740336	0,99730456	0,99713626	0,99701620	0,99728567
	03	0,99775789	0,99747176	0,99738255	0,99760985	0,99739171	0,99721197	0,99712104	0,99734905
Serpent	00	0,99800082	0,99777246	0,99760662	0,99781437	0,99742064	0,99716812	0,99719732	0,99735195
K1	01	0,99784847	0,99765976	0,99747313	0,99764661	0,99731876	0,99705484	0,99709797	0,99726646
	02	0,99777484	0,99757004	0,99743753	0,99760045	0,99724008	0,99701349	0,99700513	0,99713916
	03	0,99783648	0,99763284	0,99747591	0,99767706	0,99740425	0,99711391	0,99718341	0,99726181
Twofish	00	1	0,999716	0,99974198	0,99972826	0,99786036	0,99768536	0,99770652	0,99790147
K1	01		1	0,99975304	0,99973499	0,99749413	0,99730680	0,99736287	0,99755022
	02			1	0,99971454	0,99744901	0,99732615	0,99731652	0,99751184
	03				1	0,99771522	0,99758422	0,99757581	0,99777081
RC6	00					1	0,99974845	0,99976795	0,99977461
K1	01						1	0,99974689	0,99971298
	02							1	0,99972234
	03								1
3DES	00								
K1	01								
	02								
	03								

TAB. 8.20: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo (continuada)

		3DES	K1		
	Doc	00	01	02	03
AES	00	0,99579541	0,99533926	0,99523118	0,99561116
K1	01	0,99594112	0,99554360	0,99537489	0,99574812
	02	0,99565861	0,99522289	0,99513393	0,99547257
	03	0,99584028	0,99546828	0,99528803	0,99564544
Serpent	00	0,99596927	0,99577253	0,99559236	0,99585075
K1	01	0,99600304	0,99577957	0,99558223	0,99588745
	02	0,99574945	0,99550253	0,99535624	0,99564921
	03	0,99589541	0,99560877	0,99548307	0,99572567
Twofish	00	0,99590050	0,99549460	0,99544174	0,99567826
K1	01	0,99554747	0,99509026	0,99509010	0,99531732
	02	0,99541571	0,99496296	0,99497383	0,99523856
	03	0,99583715	0,99543289	0,99542072	0,99561727
RC6	00	0,99524884	0,99472033	0,99475143	0,99516814
K1	01	0,99527716	0,99474715	0,99476194	0,99522835
	02	0,99520692	0,99464701	0,99470416	0,99510679
	03	0,99542977	0,99488752	0,99493010	0,99534566
3DES	00	1	0,99964804	0,99962927	0,99963831
K1	01		1	0,99962043	0,99954033
	02			1	0,99957692
	03				1

TAB. 8.21: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo (continuada)

8.9 PALAVRA DE 8 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0,99732994	0,99755858	0,99760298	0,99747953	0,99759320	0,99764436	0,99788243
AES K2	01		1	0,99729390	0,99709277	0,99736043	0,99717099	0,99727245	0,99747026
AES K3	02			1	0,99696299	0,99758342	0,99753550	0,99718015	0,99741567
AES K4	03				1	0,99711500	0,99734693	0,99680203	0,99719122
Serpent K1	00					1	0,99738112	0,99722313	0,99768595
Serpent K2	01						1	0,99740209	0,99755262
Serpent K3	02							1	0,99721756
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.22: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo e documento

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0,99784026	0,99787436	0,99793110	0,99775481	0,99746669	0,99766824	0,99769873	0,9976318763
AES K2	01	0,99776609	0,99711742	0,99739966	0,99712828	0,99735421	0,99718000	0,99711343	0,9972058144
AES K3	02	0,99776085	0,99754682	0,99758510	0,99745551	0,99714062	0,99756350	0,99756698	0,9970369451
AES K4	03	0,99751402	0,99733671	0,99723452	0,99723734	0,99733776	0,99712022	0,99693189	0,997174747
Serpent K1	00	0,99800082	0,99731558	0,99755422	0,99775175	0,99742064	0,99769255	0,99772019	0,9976944602
Serpent K2	01	0,99778833	0,99747641	0,99736739	0,99755581	0,99746563	0,99775694	0,99737357	0,9974859627
Serpent K3	02	0,99729032	0,99747073	0,99747870	0,99726489	0,99726402	0,99744642	0,99737429	0,9971470546
Serpent K4	03	0,99792132	0,99782095	0,99760855	0,99750708	0,99743433	0,99766965	0,99743833	0,9976013839
Twofish K1	00	1	0,99763854	0,99769534	0,99775849	0,99786036	0,99778925	0,99778351	0,9975466858
Twofish K2	01		1	0,99772547	0,99766272	0,99738679	0,99766826	0,99756308	0,997528661
Twofish K3	02			1	0,99761999	0,99744587	0,99766499	0,99740045	0,9975208463
Twofish K4	03				1	0,99727443	0,99741527	0,99730762	0,9975455937
RC6 K1	00					1	0,99716297	0,99714022	0,9972977305
RC6 K2	01						1	0,99730005	0,9974814424
RC6 K3	02							1	0,997134711
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.23: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo e documento (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0,99579541	0,99467931	0,99447520	0,99511671
AES K2	01	0,99565219	0,99445512	0,99485158	0,99486096
AES K3	02	0,99564351	0,99473462	0,99443786	0,99486218
AES K4	03	0,99533582	0,99456548	0,99446679	0,99492163
Serpent K1	00	0,99596927	0,99435565	0,99487336	0,99557358
Serpent K2	01	0,99614220	0,99469994	0,99475995	0,99440329
Serpent K3	02	0,99594006	0,99509270	0,99441204	0,99451671
Serpent K4	03	0,99616917	0,99510360	0,99452727	0,99502057
Twofish K1	00	0,99590050	0,99501164	0,99458071	0,99525349
Twofish K2	01	0,99596858	0,99487377	0,99509328	0,99487267
Twofish K3	02	0,99614043	0,99466097	0,99456325	0,99504773
Twofish K4	03	0,99639464	0,99491692	0,99524118	0,99507729
RC6 K1	00	0,99524884	0,99518589	0,99447266	0,99459347
RC6 K2	01	0,99546746	0,99483454	0,99470908	0,99507867
RC6 K3	02	0,99568132	0,99473234	0,99444970	0,99486258
RC6 K4	03	0,99579488	0,99469948	0,99490682	0,99550454
3DES K1	00	1	0,99297205	0,99308340	0,99318182
3DES K2	01		1	0,99134452	0,99206128
3DES K3	02			1	0,99229184
3DES K4	03				1

TAB. 8.24: Matriz de Similaridade - palavra de 8 bits, chave distinta por algoritmo e documento (continuada)

APÊNDICE 10: MATRIZES DE SIMILARIDADE COM WAVELETS

8.11 PALAVRA DE 64 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO TRIVIAL

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0	0	0	0	0	0	0
AES K2	01		1	0	0	0	0	0	0
AES K3	02			1	0	0	0	0	0
AES K4	03				1	0	0	0	0
Serpent K1	00					1	0	0	0
Serpent K2	01						1	0	0
Serpent K3	02							1	0
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.25: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo trivial

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0	0	0	0	0	0	0	0
AES K2	01	0	0	0	0	0	0	0	0
AES K3	02	0	0	0	0	0	0	0	0
AES K4	03	0	0	0	0	0	0	0	0
Serpent K1	00	0	0	0	0	0	0	0	0
Serpent K2	01	0	0	0	0	0	0	0	0
Serpent K3	02	0	0	0	0	0	0	0	0
Serpent K4	03	0	0	0	0	0	0	0	0
Twofish K1	00	1	0	0	0	0	0	0	0
Twofish K2	01		1	0	0	0	0	0	0
Twofish K3	02			1	0	0	0	0	0
Twofish K4	03				1	0	0	0	0
RC6 K1	00					1	0	0	0
RC6 K2	01						1	0	0
RC6 K3	02							1	0
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.26: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo trivial(continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0	0	0	0
AES K2	01	0	0	0	0
AES K3	02	0	0	0	0
AES K4	03	0	0	0	0
Serpent K1	00	0	0	0	0
Serpent K2	01	0	0	0	0
Serpent K3	02	0	0	0	0
Serpent K4	03	0	0	0	0
Twofish K1	00	0	0	0	0
Twofish K2	01	0	0	0	0
Twofish K3	02	0	0	0	0
Twofish K4	03	0	0	0	0
RC6 K1	00	0	0	0	0
RC6 K2	01	0	0	0	0
RC6 K3	02	0	0	0	0
RC6 K4	03	0	0	0	0
3DES K1	00	1	0	0	0
3DES K2	01		1	0	0
3DES K3	02			1	0
3DES K4	03				1

TAB. 8.27: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo trivial (continuada)

8.12 PALAVRA DE 16 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO TRIVIAL

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0.4366	0.4280	0.4455	0.4598	0.4376	0.4296	0.4458
AES K2	01		1	0.4241	0.4405	0.4561	0.4343	0.4238	0.4396
AES K3	02			1	0.4218	0.4361	0.4143	0.4045	0.4208
AES K4	03				1	0.4510	0.4301	0.4206	0.4373
Serpent K1	00					1	0.4409	0.4296	0.4479
Serpent K2	01						1	0.4148	0.4323
Serpent K3	02							1	0.4248
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.28: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo trivial

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0.4540	0.4340	0.4290	0.4473	0.4520	0.4347	0.4300	0.4464
AES K2	01	0.4489	0.4293	0.4252	0.4409	0.4484	0.4316	0.4260	0.4432
AES K3	02	0.4303	0.4104	0.4071	0.4238	0.4287	0.4105	0.4053	0.4238
AES K4	03	0.4458	0.4253	0.4199	0.4389	0.4435	0.4260	0.4210	0.4384
Serpent K1	00	0.4563	0.4419	0.4357	0.4505	0.4542	0.4408	0.4346	0.4476
Serpent K2	01	0.4398	0.4267	0.4202	0.4330	0.4387	0.4247	0.4190	0.4313
Serpent K3	02	0.4314	0.4201	0.4140	0.4265	0.4303	0.4181	0.4123	0.4247
Serpent K4	03	0.4482	0.4349	0.4272	0.4418	0.4455	0.4333	0.4268	0.4382
Twofish K1	00	1	0.4357	0.4308	0.4450	0.4526	0.4407	0.4318	0.4443
Twofish K2	01		1	0.4145	0.4283	0.4365	0.4254	0.4172	0.4283
Twofish K3	02			1	0.4207	0.4281	0.4195	0.4106	0.4215
Twofish K4	03				1	0.4435	0.4321	0.4249	0.4360
RC6 K1	00					1	0.4352	0.4316	0.4406
RC6 K2	01						1	0.4148	0.4246
RC6 K3	02							1	0.4183
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.29: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo trivial (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0.3169	0.3083	0.2985	0.3122
AES K2	01	0.3142	0.3056	0.2943	0.3092
AES K3	02	0.3008	0.2925	0.2819	0.2960
AES K4	03	0.3112	0.3020	0.2901	0.3050
Serpent K1	00	0.3213	0.3057	0.3012	0.3085
Serpent K2	01	0.3096	0.2959	0.2930	0.2974
Serpent K3	02	0.3066	0.2917	0.2887	0.2930
Serpent K4	03	0.3159	0.2987	0.2986	0.3034
Twofish K1	00	0.3186	0.3120	0.2997	0.3123
Twofish K2	01	0.3051	0.3006	0.2883	0.2989
Twofish K3	02	0.3014	0.2957	0.2853	0.2942
Twofish K4	03	0.3121	0.3062	0.2938	0.3062
RC6 K1	00	0.3175	0.2982	0.2934	0.3128
RC6 K2	01	0.3053	0.2897	0.2821	0.3013
RC6 K3	02	0.3004	0.2843	0.2770	0.2964
RC6 K4	03	0.3117	0.2937	0.2885	0.3077
3DES K1	00	1	0.2153	0.2091	0.2207
3DES K2	01		1	0.2001	0.2103
3DES K3	02			1	0.2066
3DES K4	03				1

TAB. 8.30: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo trivial (continuada)

8.13 PALAVRA DE 64 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO PREFERENCIAL

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0	0	0	0	0	0	0
AES K2	01		1	0	0	0	0	0	0
AES K3	02			1	0	0	0	0	0
AES K4	03				1	0	0	0	0
Serpent K1	00					1	0	0	0
Serpent K2	01						1	0	0
Serpent K3	02							1	0
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.31: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo preferencial

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0	0	0	0	0	0	0	0
AES K2	01	0	0	0	0	0	0	0	0
AES K3	02	0	0	0	0	0	0	0	0
AES K4	03	0	0	0	0	0	0	0	0
Serpent K1	00	0	0	0	0	0	0	0	0
Serpent K2	01	0	0	0	0	0	0	0	0
Serpent K3	02	0	0	0	0	0	0	0	0
Serpent K4	03	0	0	0	0	0	0	0	0
Twofish K1	00	1	0	0	0	0	0	0	0
Twofish K2	01		1	0	0	0	0	0	0
Twofish K3	02			1	0	0	0	0	0
Twofish K4	03				1	0	0	0	0
RC6 K1	00					1	0	0	0
RC6 K2	01						1	0	0
RC6 K3	02							1	0
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.32: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo preferencial(continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0	0	0	0
AES K2	01	0	0	0	0
AES K3	02	0	0	0	0
AES K4	03	0	0	0	0
Serpent K1	00	0	0	0	0
Serpent K2	01	0	0	0	0
Serpent K3	02	0	0	0	0
Serpent K4	03	0	0	0	0
Twofish K1	00	0	0	0	0
Twofish K2	01	0	0	0	0
Twofish K3	02	0	0	0	0
Twofish K4	03	0	0	0	0
RC6 K1	00	0	0	0	0
RC6 K2	01	0	0	0	0
RC6 K3	02	0	0	0	0
RC6 K4	03	0	0	0	0
3DES K1	00	1	0	0	0
3DES K2	01		1	0	0
3DES K3	02			1	0
3DES K4	03				1

TAB. 8.33: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo preferencial (continuada)

8.14 PALAVRA DE 16 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO PREFERENCIAL

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0,4366	0,4280	0,4455	0,4564	0,4359	0,4312	0,4470
AES K2	01		1	0,4193	0,4312	0,4364	0,4197	0,4129	0,4318
AES K3	02			1	0,4231	0,4309	0,4132	0,4049	0,4218
AES K4	03				1	0,4439	0,4287	0,4230	0,4354
Serpent K1	00					1	0,4342	0,4299	0,4446
Serpent K2	01						1	0,4128	0,4271
Serpent K3	02							1	0,4265
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.34: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo preferencial

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0,4530	0,4382	0,4308	0,4444	0,4559	0,4352	0,4317	0,4466
AES K2	01	0,4350	0,4180	0,4143	0,4316	0,4323	0,4200	0,4139	0,4274
AES K3	02	0,4286	0,4138	0,4095	0,4222	0,4322	0,4158	0,4050	0,4233
AES K4	03	0,4462	0,4305	0,4239	0,4391	0,4416	0,4309	0,4223	0,4393
Serpent K1	00	0,4578	0,4425	0,4311	0,4467	0,4526	0,4321	0,4284	0,4484
Serpent K2	01	0,4354	0,4238	0,4152	0,4247	0,4347	0,4209	0,4180	0,4239
Serpent K3	02	0,4295	0,4115	0,4082	0,4222	0,4301	0,4111	0,4078	0,4208
Serpent K4	03	0,4450	0,4286	0,4230	0,4393	0,4440	0,4280	0,4245	0,4373
Twofish K1	00	1	0,4391	0,4303	0,4455	0,4511	0,4353	0,4279	0,4501
Twofish K2	01		1	0,4125	0,4298	0,4353	0,4152	0,4127	0,4272
Twofish K3	02			1	0,4219	0,4262	0,4110	0,4071	0,4210
Twofish K4	03				1	0,4432	0,4308	0,4246	0,4348
RC6 K1	00					1	0,4378	0,4293	0,4465
RC6 K2	01						1	0,4136	0,4242
RC6 K3	02							1	0,4183
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.35: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo preferencial (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0,3159	0,3035	0,2967	0,3132
AES K2	01	0,3091	0,2971	0,2890	0,3008
AES K3	02	0,3030	0,2867	0,2889	0,2956
AES K4	03	0,3091	0,2945	0,2946	0,3105
Serpent K1	00	0,3205	0,3056	0,2956	0,3090
Serpent K2	01	0,3053	0,2919	0,2874	0,2996
Serpent K3	02	0,3003	0,2848	0,2809	0,2936
Serpent K4	03	0,3124	0,2970	0,2923	0,3076
Twofish K1	00	0,3197	0,3040	0,3016	0,3117
Twofish K2	01	0,3098	0,2874	0,2861	0,3006
Twofish K3	02	0,2984	0,2928	0,2833	0,3008
Twofish K4	03	0,3121	0,2966	0,2925	0,3117
RC6 K1	00	0,3162	0,3028	0,2999	0,3089
RC6 K2	01	0,3049	0,2913	0,2863	0,3026
RC6 K3	02	0,2988	0,2889	0,2780	0,2954
RC6 K4	03	0,3083	0,2983	0,2973	0,3056
3DES K1	00	1	0,2142	0,2068	0,2233
3DES K2	01		1	0,2031	0,2171
3DES K3	02			1	0,2003
3DES K4	03				1

TAB. 8.36: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo preferencial (continuada)

8.15 PALAVRA DE 64 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO *WAVELET*

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0	0	0	0	0	0	0
AES K2	01		1	0	0	0	0	0	0
AES K3	02			1	0	0	0	0	0
AES K4	03				1	0	0	0	0
Serpent K1	00					1	0	0	0
Serpent K2	01						1	0	0
Serpent K3	02							1	0
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.37: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo *wavelet*

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0	0	0	0	0	0	0	0
AES K2	01	0	0	0	0	0	0	0	0
AES K3	02	0	0	0	0	0	0	0	0
AES K4	03	0	0	0	0	0	0	0	0
Serpent K1	00	0	0	0	0	0	0	0	0
Serpent K2	01	0	0	0	0	0	0	0	0
Serpent K3	02	0	0	0	0	0	0	0	0
Serpent K4	03	0	0	0	0	0	0	0	0
Twofish K1	00	1	0	0	0	0	0	0	0
Twofish K2	01		1	0	0	0	0	0	0
Twofish K3	02			1	0	0	0	0	0
Twofish K4	03				1	0	0	0	0
RC6 K1	00					1	0	0	0
RC6 K2	01						1	0	0
RC6 K3	02							1	0
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.38: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo *wavelet*(continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0	0	0	0
AES K2	01	0	0	0	0
AES K3	02	0	0	0	0
AES K4	03	0	0	0	0
Serpent K1	00	0	0	0	0
Serpent K2	01	0	0	0	0
Serpent K3	02	0	0	0	0
Serpent K4	03	0	0	0	0
Twofish K1	00	0	0	0	0
Twofish K2	01	0	0	0	0
Twofish K3	02	0	0	0	0
Twofish K4	03	0	0	0	0
RC6 K1	00	0	0	0	0
RC6 K2	01	0	0	0	0
RC6 K3	02	0	0	0	0
RC6 K4	03	0	0	0	0
3DES K1	00	1	0	0	0
3DES K2	01		1	0	0
3DES K3	02			1	0
3DES K4	03				1

TAB. 8.39: Matriz de Similaridade - palavra de 64 bits, chave distinta por algoritmo e documento, modelo *wavelet* (continuada)

8.16 PALAVRA DE 16 BITS, CHAVES DISTINTAS POR ALGORITMO E DOCUMENTO, MODELO *WAVELET*

		AES K1	AES K2	AES K3	AES K4	Serpent K1	Serpent K2	Serpent K3	Serpent K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	1	0,4371	0,4285	0,4460	0,4569	0,4364	0,4317	0,4475
AES K2	01		1	0,4198	0,4316	0,4369	0,4202	0,4134	0,4323
AES K3	02			1	0,4236	0,4313	0,4137	0,4053	0,4222
AES K4	03				1	0,4444	0,4291	0,4235	0,4359
Serpent K1	00					1	0,4347	0,4304	0,4451
Serpent K2	01						1	0,4133	0,4276
Serpent K3	02							1	0,4270
Serpent K4	03								1
Twofish K1	00								
Twofish K2	01								
Twofish K3	02								
Twofish K4	03								
RC6 K1	00								
RC6 K2	01								
RC6 K3	02								
RC6 K4	03								
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.40: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo *wavelet*

		Twofish K1	Twofish K2	Twofish K3	Twofish K4	RC6 K1	RC6 K2	RC6 K3	RC6 K4
	Doc	00	01	02	03	00	01	02	03
AES K1	00	0,4535	0,4387	0,4314	0,4449	0,4564	0,4357	0,4322	0,4471
AES K2	01	0,4354	0,4184	0,4147	0,4320	0,4327	0,4205	0,4143	0,4279
AES K3	02	0,4291	0,4143	0,4100	0,4226	0,4327	0,4163	0,4055	0,4237
AES K4	03	0,4467	0,4310	0,4244	0,4396	0,4421	0,4314	0,4228	0,4398
Serpent K1	00	0,4583	0,4430	0,4316	0,4472	0,4531	0,4326	0,4289	0,4489
Serpent K2	01	0,4359	0,4242	0,4157	0,4252	0,4352	0,4214	0,4185	0,4244
Serpent K3	02	0,4300	0,4120	0,4087	0,4227	0,4306	0,4115	0,4083	0,4213
Serpent K4	03	0,4454	0,4291	0,4235	0,4398	0,4445	0,4285	0,4250	0,4377
Twofish K1	00	1	0,4396	0,4307	0,4460	0,4515	0,4356	0,4284	0,4506
Twofish K2	01		1	0,4130	0,4303	0,4358	0,4156	0,4131	0,4277
Twofish K3	02			1	0,4224	0,4267	0,4114	0,4076	0,4214
Twofish K4	03				1	0,4437	0,4313	0,4251	0,4353
RC6 K1	00					1	0,4383	0,4297	0,4470
RC6 K2	01						1	0,4140	0,4247
RC6 K3	02							1	0,4187
RC6 K4	03								1
3DES K1	00								
3DES K2	01								
3DES K3	02								
3DES K4	03								

TAB. 8.41: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo *wavelet* (continuada)

		3DES K1	3DES K2	3DES K3	3DES K4
	Doc	00	01	02	03
AES K1	00	0,3162	0,3038	0,2969	0,3135
AES K2	01	0,3094	0,2974	0,2892	0,3011
AES K3	02	0,3033	0,2869	0,2892	0,2959
AES K4	03	0,3094	0,2948	0,2948	0,3107
Serpent K1	00	0,3207	0,3059	0,2959	0,3093
Serpent K2	01	0,3055	0,2921	0,2877	0,2999
Serpent K3	02	0,3006	0,2851	0,2811	0,2938
Serpent K4	03	0,3126	0,2973	0,2925	0,3078
Twofish K1	00	0,3200	0,3043	0,3018	0,3119
Twofish K2	01	0,3100	0,2876	0,2864	0,3008
Twofish K3	02	0,2986	0,2930	0,2836	0,3010
Twofish K4	03	0,3123	0,2968	0,2927	0,3119
RC6 K1	00	0,3164	0,3030	0,3001	0,3091
RC6 K2	01	0,3052	0,2916	0,2865	0,3029
RC6 K3	02	0,2990	0,2891	0,2782	0,2956
RC6 K4	03	0,3086	0,2985	0,2976	0,3056
3DES K1	00	1	0,2143	0,2069	0,2235
3DES K2	01		1	0,2032	0,2173
3DES K3	02			1	0,2004
3DES K4	03				1

TAB. 8.42: Matriz de Similaridade - palavra de 16 bits, chave distinta por algoritmo e documento, modelo *wavelet* (continuada)