UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

PROTÓTIPO DE SOFTWARE PARA OCULTAR TEXTO CRIPTOGRAFADO EM IMAGENS DIGITAIS.

FÁBIO LUIS TAVARES JASCONE

BLUMENAU
2003

2003/2-12

FÁBIO LUIS TAVARES JASCONE

PROTÓTIPO DE SOFTWARE PARA OCULTAR TEXTO CRIPTOGRAFADO EM IMAGENS DIGITAIS.

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciência da Computação — Bacharelado.

Prof. Francisco Adell Péricas - Orientador

BLUMENAU
2003

2003/2-12

PROTÓTIPO DE SOFTWARE PARA OCULTAR TEXTO CRIPTOGRAFADO EM IMAGENS DIGITAIS.

Por

FÁBIO LUIS TAVARES JASCONE

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Francisco Adell Péricas, MSc. – Orientador, FURB
Membro:	Prof. Paulo César Rodacki Gomes, FURB
Membro:	Prof. Maurício Capobianco Lopes, FURB

Blumenau, 12 de novembro de 2003

Dedico este trabalho aos meus pais pelo incentivo, carinho e amor que nunca me faltaram, e também pela paciência em esperar o primeiro filho adquirir grau superior.

Não faça da vida um drama, pule os obstáculos, quebre as barreiras, escale penhascos se for preciso, mas não deixe de ser feliz e seguir em frente.

Fábio Luis Tavares Jascone

AGRADECIMENTOS

A Deus, por me dar algo tão precioso, a vida.

Aos meus pais, Carlos e Selma, por me amarem, por estarem sempre ao meu lado em todos os momentos, pelos conselhos, apoio e incentivo à conclusão deste curso.

À minha namorada, Ana Luisa, pelo amor e compreensão em todos os momentos, principalmente na minha ausência no decorrer deste trabalho.

Ao meu orientador, Francisco Adell Péricas, pela sua paciência, dedicação e sabedoria.

Aos meus grandes amigos, aqueles que sabem que são considerados como tal, que junto comigo hoje fazem parte da turma Amigos do Barney, e que mesmo sem saberem me fazem crescer e ser ainda mais feliz.

RESUMO

Este trabalho apresenta a especificação e implementação de um protótipo para ocultar mensagens criptografadas em imagens digitais, utilizando algoritmo de criptografia em conjunto com a esteganografia. Para isto, utilizou-se o algoritmo de Rijndael, no qual se baseia o novo padrão *Advanced Encryption Standard* (AES), principal algoritmo simétrico de encriptação de dados, em conjunto com o método *Last Significant Bit* (LSB) de esteganografia, para ocultar a informação criptografada em uma imagem.

Palavras chaves: Esteganografia; Criptografia.

ABSTRACT

This work presents the specification and implementation of an archetype to occult encrypted messages in digital images, using cryptography algorithm together with steganography. For this, the algorithm of Rijndael was used, in which if it bases the new Advanced Encryption Standard (AES), main symmetrical algorithm of encryptation of data, together with method Last Significant Bit (LSB) of steganography, to occult the encrypted information in an image.

Key-Words: Steganography; Cryptography.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Com o crescente uso das redes de computadores e da internet por organizações para conduzir seus negócios, surgiu a necessidade de se utilizar melhores mecanismos para garantir a segurança das transações de informações confidenciais. A questão segurança é bastante enfatizada, principalmente quando se imagina a possibilidade de se ter suas informações expostas a intrusos, que surgem com meios cada vez mais sofisticados para violar a privacidade e a segurança das comunicações. Devido a estas preocupações, a proteção da informação tem se tornado um dos interesses primários dos administradores de sistemas (MACÊDO e TRINTA, 1998).

Segundo Lucchensi (1986), à medida que redes de computadores tornam-se mais disseminadas, a necessidade de se tornar os dados seguros e autênticos também aumenta. Mensagens e dados precisam ser protegidos de modo que somente pessoas ou processos autorizados consigam utilizá-los.

Em informática, a segurança consiste na certeza de que as informações de uso restrito, não devem ser acessadas por pessoas não autorizadas. Para garantir isto, é necessário que estas informações sejam criptografadas. Segundo Stang e Moon (1994), qualquer bom sistema de segurança de dados deve incluir a criptografia.

Criptografia é a arte ou a ciência de escrever em cifra ou em código (LUCCHENSI, 1986). Em outras palavras, são os princípios, meios e métodos de transformar uma informação originalmente expressa com clareza, em uma forma codificada, ou seja, permite que apenas o destinatário autorizado decifre e compreenda a informação original. É um mecanismo de segurança que permite a implementação de diversos serviços (autenticação, não-repúdio, integridade, confidencialidade, controle de acesso). Para criptografar um texto, utiliza-se uma chave, da qual depende grande parte da segurança do processo.

A criptografia pode ser classificada em duas categorias básicas, de acordo com o tipo de chave utilizada: sistema de chave simétrica (ou chave secreta), onde a chave usada para criptografar os dados é a mesma usada para decriptografar, e que tem como principal método de encriptação o *Data Encryption Standard* (DES) e sistema de chave assimétrica (ou chave pública), onde as chaves de criptografia e decriptografia são diferentes, e que tem como principal método de encriptação o Rivest, Shamir, Adleman (RSA).

Junto com a criptografia, pode-se usar a esteganografia, que significa “esconder escrita”, ou seja, camuflar uma informação dentro de outra. Enquanto a criptografia esconde o significado da mensagem, a esteganografia esconde a existência dessa mensagem.

A esteganografia é bastante utilizada por quem necessita comunicar-se e trocar informações sigilosas, mas sem levantar suspeitas. Um exemplo, segundo Wilner (2001), seriam os terroristas, que utilizam a esteganografia para obter informações suficientes para a elaboração de um atentado e para passar as ordens de execução do mesmo. Hoje em dia há vários canais de comunicação disponíveis, através do telefone, fax, e-mail, ICQ, cartas, telegramas, entre outros. Porém, o que é praticamente imune a qualquer tipo de vigilância eletrônica é a esteganografia.

Um caso típico de esteganografia é codificar uma mensagem como mudanças sutis nos *pixels* de uma foto digital, ou ruídos imperceptíveis em um arquivo de áudio. Para um observador comum, é só uma foto ou um arquivo de som. Mas para quem envia e quem recebe, existe uma mensagem (WILNER, 2001).

Em imagens, a informação pode ser escondida codificando cada bit dessa informação ao longo da imagem, roubando um bit de cada *pixel*, ou simplesmente colocando a informação em áreas não utilizadas da imagem, que não chamam atenção. O método clássico é chamado de inserção do último bit significativo (LSB – *Last Significant Bit*).

De um modo geral, pode-se dizer que a criptografia embaralha uma informação tornando-a incompreensível, enquanto a esteganografia preocupa-se em esconder a informação, sendo essa criptografada ou não, dentro de outra aparentemente inofensiva.

Assim, como sugestão de extensão do trabalho de conclusão de curso de Zanella (2002), neste trabalho será utilizado um algoritmo simétrico de criptografia, em conjunto com técnicas de esteganografia, para desenvolver um protótipo de software para criptografar mensagens, através de uma chave secreta definida pelo usuário e inserir essa informação criptografada em imagens do tipo *bitmap*; e o processo inverso, extrair a informação da imagem e decriptografá-la.

1.1 OBJETIVOS

O objetivo do trabalho proposto é desenvolver um protótipo de software que utiliza métodos de criptografia em conjunto com a esteganografia, para esconder texto criptografado em imagens digitais, e vice e versa.

Como objetivos específicos destacam-se:

a) construção de uma aplicação onde o usuário redigirá uma mensagem que será escondida em uma imagem, ou irá extrair uma mensagem de uma imagem;
b) uso de algoritmo de criptografia de chave secreta;
c) aplicação do método LSB de esteganografia em imagens;

1.2 ESTRUTURA

O presente trabalho está subdividido em capítulos que serão explicitados a seguir.

O primeiro capítulo apresenta a contextualização e justificativa para o desenvolvimento da proposta do trabalho.

O segundo capítulo aborda segurança da informação, detalhando alguns conceitos importantes sobre: ataques, mecanismos de segurança, políticas de segurança, vulnerabilidades, etc.

O terceiro capítulo explica criptografia, desde a encriptação convencional ao funcionamento de alguns dos principais algoritmos de criptografia, de chave secreta e pública.

O quarto capítulo detalha o item esteganografia, desde a sua história até a aplicação na atualidade. Aborda também o método de inserção do último bit significante (LSB), o funcionamento da esteganografia com imagens digitais e faz um breve comparativo com a criptografia. Além disso, é explicado o funcionamento da compressão de Huffman, método eficiente utilizado para comprimir texto.

O quinto capítulo trata sobre o desenvolvimento do trabalho, mostrando os diagramas de classe, casos de uso e diagramas de seqüência. Este capítulo também explica a implementação do protótipo e os resultados obtidos.

O sexto capítulo apresenta as considerações finais, abrangendo as conclusões do desenvolvimento deste trabalho, as dificuldades encontradas e as sugestões para próximos trabalhos.

2 SEGURANÇA DA INFORMAÇÃO

A segurança em informática consiste na certeza de que as informações de uso restrito não devem ser acessadas, copiadas ou codificadas por pessoas não autorizadas. Para a garantia disto, é necessário que as informações sejam cifradas.

A questão segurança é bastante enfatizada, principalmente, quando se imagina a possibilidade de se ter as informações expostas a intrusos, que surgem com meios cada vez mais sofisticados para violar a privacidade e a segurança das comunicações.

Nos primórdios da informática, quando as corporações e universidades tinham um único centro de computação, era fácil conseguir segurança. Tudo o que a organização tinha que fazer era colocar um guarda à porta da sala do computador. O guarda garantiria que ninguém removesse quaisquer fitas, discos ou cartões da sala, a menos que estivesse explicitamente autorizado a fazê-lo (TANENBAUM, 1994).

Segundo Stang e Moon (1993), o objetivo principal da segurança de informações é controlar o acesso a informação. Somente pessoas devidamente autorizadas devem estar habilitadas a apreciar, criar, apagar, ou modificar informações.

Uma das maneiras de se evitar o acesso indevido a informações confidenciais é através da codificação ou cifragem da informação, conhecida como criptografia, fazendo com que apenas as pessoas às quais estas informações são destinadas, consigam compreendê-las. A criptografia fornece técnicas para codificar e decodificar dados, tais que os mesmos possam ser armazenados, transmitidos e recuperados sem sua alteração ou exposição. Em outras palavras, técnicas de criptografia podem ser usadas como um meio efetivo de proteção de informações suscetíveis a ataques, estejam elas armazenadas em um computador ou sendo transmitidas pela rede. Seu principal objetivo é prover uma comunicação segura, garantindo serviços básicos de autenticação, privacidade e integridade dos dados.

2.1 ATAQUES À SEGURANÇA

Os ataques à segurança são qualquer ação que comprometa a segurança de informação. Eles podem ser classificados como ataques passivos, nos quais o intruso apenas lê as informações que trafegam pela rede e ataques ativos, nos quais além de ler as informações, um intruso pode modificá-las e forjá-las.

Stallings (1995) cita quatro categorias para esses ataques (figura 1):

a) interceptação (passivo): uma entidade não autorizada ganha acessos aos componentes ativos, permitindo a captura de informações sigilosas como *username* e *password* (senhas);
b) interrupção (ativo): um componente ativo do sistema é destruído, ou torna-se indisponível ou inutilizável;
c) fabricação (ativo): entidade não autorizada insere objetos falsificados no sistema;
d) modificação (ativo): uma entidade não autorizada não apenas obtém acesso, mas falsifica componentes ativos.

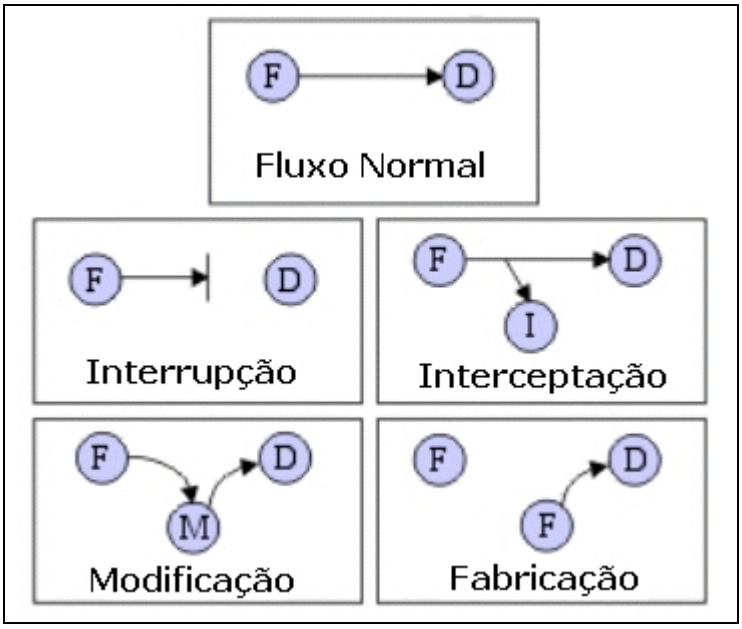


FIGURA 1 – Ameaças da Segurança

Nos ataques ativos, o intruso utiliza, de algum modo, as facilidades da rede. Isso pode significar o uso direto de forma não autorizada; pode também significar a interceptação física da rede, instalando tanto um "grampo" para escuta, ou um nó completo não autorizado. Com um "grampo" ativo simulando um nó, o intruso não só recebe todo o tráfego, como pode também introduzir algo, aumentando ainda mais a ameaça. O intruso pode até mesmo impedir a utilização da rede por outras pessoas, congestionando-a com o tráfego. Se os protocolos utilizados forem do tipo onde o nó não autorizado está em posição de capturar e liberar tráfego, há o risco potencial de endereçar e modificar pacotes de dados.

2.2 MECANISMOS DE SEGURANÇA

Os mecanismos de segurança são técnicas, procedimentos e algoritmos que quando usados adequadamente possibilitam a implementação dos serviços de segurança, pois são mecanismos projetados para detectar, prever ou descobrir um ataque de segurança.

2.2.1 CRIPTOGRAFIA

Em meios de comunicação onde não é possível impedir que o fluxo de pacote de dados seja interceptado, podendo as informações serem lidas ou até modificadas, é necessária a criptografia. Nesse mecanismo, utiliza-se um método que modifique o texto original da mensagem transmitida, gerando um texto criptografado na origem, através de um processo de codificação definido por um método de criptografia. O pacote é então transmitido e, ao chegar no destino, ocorre o processo inverso, isto é, o método de criptografia é aplicado agora para decodificar a mensagem, transformando-a na mensagem original. Esse assunto é detalhado no capítulo 3.

2.2.2 ASSINATURA DIGITAL

Assinatura digital é um conjunto de procedimentos matemáticos realizados com a utilização de técnicas de criptografia assimétrica que permite, de forma única e exclusiva, a comprovação da autoria de um determinado conjunto de dados.

Um determinado usuário *A* codifica uma mensagem utilizando sua chave secreta e a envia para o destinatário. Somente a chave pública de *A* permitirá a decodificação da mensagem, portanto é a prova de que *A* enviou a mensagem. A mensagem assim pode ser decodificada por qualquer um que tenha a chave pública de *A*. Para garantir o sigilo deve-se criptografar duas vezes a mensagem: a primeira utilizando a própria chave secreta (para fazer a assinatura digital) e a seguir utilizando a chave pública do destinatário, para que somente este possa ler a mensagem (SANTOS, EMER e AVER, 1996).

As propriedades da assinatura digital são:

a) a assinatura é autêntica: quando um usuário usa a chave pública de A para decifrar uma mensagem, ele confirma que foi A e somente A quem enviou a mensagem;
b) a assinatura não pode ser forjada: somente A conhece sua chave secreta;
c) o documento assinado não pode ser alterado: se houver qualquer alteração no texto

criptografado este não poderá ser restaurado com o uso da chave pública de A;

d) a assinatura não é reutilizável: a assinatura é uma função do documento e não pode ser transferida para outro documento;
e) a assinatura não pode ser repudiada: o usuário B não precisa de nenhuma ajuda de A para reconhecer sua assinatura e A não pode negar ter assinado o documento.

2.2.3 INTEGRIDADE DOS DADOS

Os mecanismos de controle de integridade de dados atuam em dois níveis: controle da integridade de pacotes isolados e controle da integridade de uma conexão, isto é, dos pacotes e da seqüência de transmissão.

Em relação ao primeiro nível, tem-se que técnicas de detecção de modificações, que são normalmente associadas com a detecção de erros em bits, pacotes ou erros de seqüência introduzidos por enlaces e redes de comunicação, são usadas para garantir a integridade dos dados trafegados em uma rede. Contudo, se os cabeçalhos dos pacotes de dados não forem devidamente protegidos contra possíveis modificações, pode-se contornar a verificação, desde que sejam conhecidas essas técnicas. Portanto, para garantir a integridade é necessário manter confidenciais e íntegras as informações de controle que são usadas na detecção de modificações.

Já para controlar modificações na seqüência de pacotes transmitidos em uma conexão, são necessárias técnicas que garantam a integridade desses pacotes, de forma a garantir que as informações de controle não sejam corrompidas, em conjunto com informações de controle de seqüência. Esses cuidados, apesar de não evitarem a modificação da cadeia de pacotes, garantem a detecção e notificação dos ataques (SOARES et al, 1995).

2.2.4 CONTROLE DE ACESSO

Esse mecanismo de segurança é utilizado para garantir que o acesso a um recurso de rede qualquer seja limitado a usuários devidamente autorizados pelo administrador do sistema.

Como técnicas utilizadas, tem-se a utilização de listas ou matrizes de controles de acesso, que associam recursos a usuários autorizados, ou senhas e *tokens* (método de acesso a rede que utiliza uma espécie de ficha eletrônica circulante a fim de impedir que mais de um

nó transmita simultaneamente) associadas aos recursos, cuja posse determina os direitos de acesso do usuário que as possui.

Como exemplo da utilização de *tokens* para controlar o acesso aos recursos de uma rede, pode-se considerar o método de controle de congestionamento de tráfego conhecido como *controle isorrítmico*. Nesse método, existem permissões, que são os *tokens*, que circulam pela rede. Sempre que um *host* deseja transmitir um novo pacote pela rede, ele primeiramente deve capturar uma dessas permissões e destruí-la, sendo que essa permissão destruída é regenerada pelo *host* que recebe o pacote no destino. Contudo, esse método apresenta um problema: a distribuição das permissões depende das aplicações na rede e o próprio tráfego aleatório desses tokens causa um tráfego extra na rede, diminuindo assim o seu desempenho. Ainda, tem-se que a perda de uma permissão devido a uma falha qualquer na rede deve ser recuperada, de forma a evitar que a sua capacidade de transporte seja reduzida (SOARES et al, 1995).

2.2.5 CONTROLE DE ROTEAMENTO

Esse mecanismo garante a transmissão de informação através de rotas fisicamente seguras, cujos canais de comunicação forneçam os níveis apropriados de proteção. Essa garantia se deve ao controle do roteamento de pacotes de dados. Através desse controle, rotas preferenciais (ou obrigatórias) para a transferência de dados são especificadas pelo administrador do sistema.

2.2.6 FIREWALL

Serviço utilizado para aumentar a segurança de redes, principalmente redes locais ligadas à Internet. Trata de uma espécie de barreira de proteção constituída de um conjunto de hardware e software. Pode-se dizer também que *firewall* é um conjunto de políticas de segurança que tem como objetivo tornar uma segurança eficiente.

Firewall é um sistema ou um grupo de sistemas que garante uma política de controle de acesso entre duas redes (normalmente a Internet e uma rede local). Em princípio *firewalls* podem ser vistos como um par de mecanismos: um que existe para bloquear o tráfego e outro que existe para permitir o tráfego. Alguns *firewalls* dão maior ênfase ao bloqueio de tráfego, enquanto outros enfatizam a permissão do tráfego. O importante é configurar o *firewall* de

acordo com a política de segurança da organização que o utiliza, estabelecendo o tipo de acesso que deve ser permitido ou negado (KANISHIMA et al, 2000).

2.3 POLÍTICA DE SEGURANÇA

Uma política de segurança é definida como sendo um conjunto de leis, regras e práticas que definem como uma empresa, ou instituição, gerencia e protege seus recursos e transmitem os seus dados. Um sistema de comunicação de dados pode ser considerado seguro quando garante o cumprimento dessa política, que deve incluir regras detalhadas definindo como as informações e recursos oferecidos pela rede devem ser manipulados.

Uma política de segurança é implementada baseando-se na aplicação de regras que controlem o acesso aos dados e recursos que são trafegados através da rede, isto é, define-se o que é e o que não é permitido em termos de segurança, durante a operação de um dado aplicativo ou recurso da rede, através da definição do nível de acesso autorizado para os usuários que se utilizam do sistema de comunicação de dados. Com base na natureza da autorização que é dada ao usuário, pode-se dividir em dois os tipos de política de segurança existentes: uma baseada em regras, onde os dados e recursos da rede são marcados com rótulos de segurança apropriados que definem o nível de autorização do usuário que os está controlando e uma outra baseada em identidade. Nesse último tipo, o administrador da rede pode especificar explicitamente os tipos de acesso que os usuários da rede podem ter às informações e recursos que estão sob seu controle (SOARES et al, 1995).

2.4 VULNERABILIDADE

Nenhum sistema de segurança é impenetrável. Em qualquer sistema de segurança é necessário saber se a informação que está sendo protegida é mais valiosa para um eventual agressor do que o custo que este teria para burlar o sistema de proteção. Isto ajudaria a se proteger de ataques de baixo custo sem a preocupação com meios mais sofisticados e caros de espionagem.

2.5 SEGURANÇA NA INTERNET

Segundo Kanishima et al (2000), a Internet foi projetada visando fornecer conectividade entre computadores para uma comunidade restrita de usuários que confiavam

mutuamente entre si. Ela não foi projetada para um ambiente comercial, para tráfego de informações valiosas ou sensíveis, ou para resistir a ataques mal-intencionados. Durante a década de 80, antes da popularização da Internet, os computadores foram alvos de ataques individuais e isolados. A solução adotada foi relativamente simples: incentivar os usuários a escolherem boas senhas, prevenir o compartilhamento indiscriminado de contas e arquivos e eliminar as falhas de segurança de programas como *sendmail*, *finger* e *login* à medida que eles iam sendo descobertos.

A partir da década de 90, entretanto, os ataques se tornaram sofisticados e organizados: senhas e outras informações importantes são capturadas, computadores são invadidos, sessões são desviadas, dados são comprometidos pela inserção de informação espúria, etc.

Estes ataques são diretamente relacionados ao protocolo IP que não foi projetado para o ambiente atual da Internet. Assumiu-se que esta tarefa seria realizada por protocolos de maior nível de abstração.

O protocolo IP está em permanente evolução, futuras versões provavelmente fornecerão a segurança e a confiabilidade requeridas. Esta característica, entretanto, também tem suas desvantagens, uma vez que o IP está sendo usado em ambientes para os quais não foi originalmente projetado (KANISHIMA et al, 2000).

Atualmente, a única maneira segura de proteger os pacotes contra espionagem na internet é utilizando criptografia através de alguns métodos de cifragem:

a) em nível de enlace: onde os pacotes são automaticamente cifrados quando enviados por um canal inseguro;
b) entre origem e destino: onde os pacotes são cifrados pela máquina origem e decifrados somente na máquina de destino;
c) em nível de aplicativo: onde a cifragem é realizada pelo próprio programa aplicativo.

Infelizmente, a cifragem dos dados não resolve todos os problemas. A falta de um mecanismo efetivo de autenticação permite que atacantes possam alterar ou falsificar a origem de diversas conexões. Isto é particularmente efetivo para correio eletrônico, serviços de notícias e *WWW*.

3 CRIPTOGRAFIA

Criptografia é a arte ou a ciência de escrever em cifra ou em código, ou seja, é um conjunto de técnicas que permite tornar incompreensível uma mensagem originalmente escrita com clareza, de forma a permitir normalmente que apenas o destinatário a decifre e compreenda. Quase sempre o deciframento requer o conhecimento de uma chave, uma informação secreta disponível ao destinatário (LUCCHESI, 1986).

Segundo Santos, Emer e Aver (1996), a criptografia já esteve presente no sistema de escrita hieroglífica dos egípcios. Desde então vem sendo muito utilizada, principalmente para fins militares e diplomáticos. No âmbito da computação a criptografia é importante para que se possa garantir a segurança em todo o ambiente computacional que necessite de sigilo em relação às informações que manipula. Pode ser usada para se codificar dados e mensagens antes que esses sejam enviados por vias de comunicação, para que mesmo que sejam interceptados, dificilmente possam ser decodificados.

A criptografia computacional é usada para garantir:

a) sigilo: somente os usuários autorizados têm acesso à informação;
b) integridade: garantia de que a informação original não foi alterada;
c) autenticação de usuário: garante que a pessoa com quem se está comunicando é realmente quem diz ser;
d) autenticação de remetente: garante que a mensagem recebida foi de fato enviada pelo remetente;
e) autenticação do destinatário: garante que a mensagem enviada foi recebida pelo destinatário;
f) autenticação de atualidade: garante que a mensagem é atual.

Segundo Soares et al (1995), a criptografia surgiu da necessidade de se enviar informações sensíveis através de meios de comunicação não confiáveis, ou seja, em meios onde não é possível garantir que um intruso não irá interceptar o fluxo de dados para leitura ou para modificá-lo. A forma de contornar esse problema é utilizar um método que modifique o texto original da mensagem a ser transmitida, gerando texto criptografado na origem, através de um processo de codificação definido por um método de criptografia. O texto criptografado é então transmitido e, no destino, o processo inverso ocorre, isto é, o método de

criptografia é aplicado agora para decodificar o texto criptografado transformando-o no texto original.

A criptografia de dados é feita através de algoritmos que realizam o ciframento e o deciframtento dos dados. Esses algoritmos utilizam, na maioria das vezes, técnicas baseadas em fórmulas matemáticas com diferentes combinações. O algoritmo pode ser amplamente conhecido, pois a segurança do sistema consiste na chave utilizada para cifrar e decifrar os dados, conhecida apenas pelas partes envolvidas no processo. Segundo Oliveira (2000), a criptografia é uma das melhores formas de garantir a confidencialidade das informações que trafegam pela internet.

Os algoritmos criptográficos podem ser classificados em dois tipos: os de chave secreta e os de chave pública. Os algoritmos de chave secreta, também chamados de algoritmos simétricos, caracterizam-se por utilizar a mesma chave tanto para a cifragem como para a decifragem dos dados. Os algoritmos de chave pública, também chamados de algoritmos assimétricos, utilizam-se de duas chaves: uma secreta só conhecida por pessoas autorizadas e outra pública que pode ser divulgada.

Sistemas criptográficos são geralmente classificados quanto a sua funcionalidade e eficiência ao longo de três dimensões:

a) o tipo de operações usadas para codificar o texto: os algoritmos de encriptação são baseados em dois princípios gerais: substituição, em que cada elemento no texto original é mapeado em outro elemento; e transposição, em que elementos no texto claro são rearranjados. O requisito fundamental para o bom funcionamento e eficiência do sistema é que a mensagem não seja perdida, que todas as operações sejam reversíveis;
b) o número de chaves usado: se ambos, transmissor e receptor, usam a mesma chave, o sistema é denominado simétrico, de chave secreta ou encriptação convencional. Se cada um, transmissor e receptor, usa uma chave diferente, o sistema é denominado assimétrico, de duas chaves ou enctriptação de chave pública;
c) o modo em que o texto original é processado: um bloco cifrado processa a entrada em um bloco de elementos no momento, produzindo um bloco de saída para cada bloco de entrada; enquanto um nível cifrado processa os elementos de entrada continuamente, produzindo saída todo o tempo.

3.1 ENCRIPTAÇÃO CONVENCIONAL

A encriptação convencional, ou tradicional, é também definida como encriptação simétrica ou de chave secreta.

Nesta técnica, uma mesma chave é utilizada para criptografar e decriptografar uma mensagem que, portanto, deve ser de conhecimento tanto do emissor como do receptor da mesma. Em sistemas criptográficos simétricos, os algoritmos de criptografia e decriptografia são os mesmos, muda-se apenas a forma como são utilizadas as chaves.

A segurança da encriptação convencional depende de alguns fatores. O algoritmo de encriptação precisa ser implementado de maneira que torne impossível decriptografar uma mensagem baseando-se apenas no texto cifrado. Além disso, a segurança da criptografia convencional depende do sigilo da chave, e não do sigilo do algoritmo, tanto é que o funcionamento dos principais algoritmos de criptografia é acessível a todos. Com isso assume-se que é impossível decriptografar uma mensagem baseando-se no texto cifrado mesmo tendo conhecimento do algoritmo de encriptação e decriptação (KONKOL, 1997).

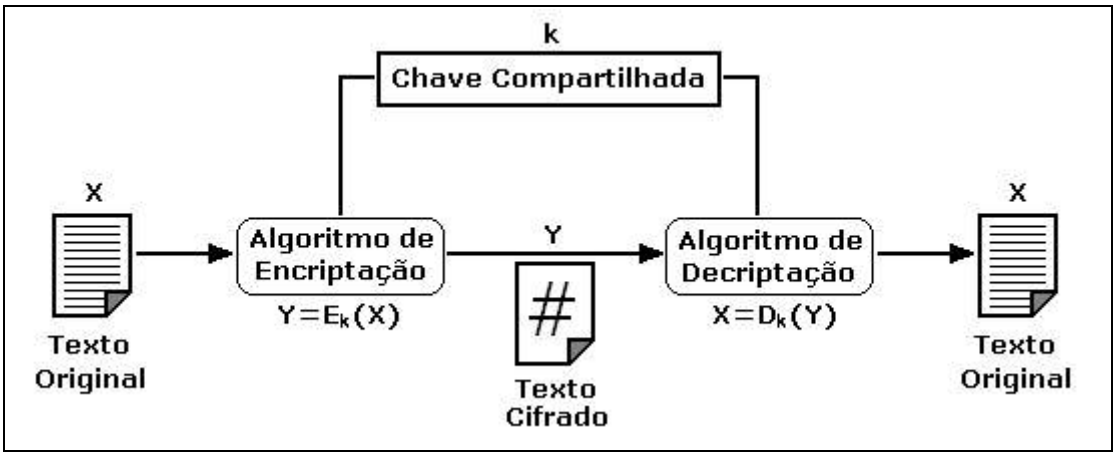


FIGURA 2 – Modelo Simplificado da Encriptação Convencional

Como mostra a figura 2, a mensagem original é encriptada com uma chave *K* e enviada através de algum meio eletrônico. Porém para o destinatário conseguir decifrar esta mensagem, ele deve ter a mesma chave *K* utilizada na encriptação. Com este modelo pode-se garantir a confidencialidade da mensagem, desde que somente transmissor e receptor tenham conhecimento da chave *K*.

Com a mensagem X e a chave de encriptação K como entrada, o algoritmo de encriptação, ou decriptação, forma o texto cifrado $Y = [Y1, Y2, ..., Yn]$. Pode-se escrever isso com $Y = E_k(X)$, onde Y é produzido pelo uso do algoritmo E em função do texto X, com a função específica determinada pelo valor da chave K. O receptor, conhecendo a chave K, é capaz de inverter a transformação através de $X = D_k(Y)$, onde D representa o algoritmo de decriptação.

O texto cifrado não sofre alteração quanto ao seu tamanho. É importante salientar também que o texto cifrado não contém qualquer parte da chave.

3.1.1 CIFRAS DE SUBSTITUIÇÃO

É uma forma de encriptação que troca cada caracter ou grupo de caracteres por outro, de acordo com uma tabela de substituição. Pode-se quebrar esse método analisando-se a frequência de cada caracter no texto cifrado e comparando-se essas frequências com aquelas que normalmente aparecem em um determinado idioma. As vogais têm maior frequência que as consoantes e alguns caracteres possuem frequência baixíssima em relação aos demais (SANTOS, EMER e AVER, 1996).

A substituição é chamada monoalfabética quando não depende da posição, mas apenas da letra original. Quando depende também da posição, é chamada polialfabética.

3.1.2 CIFRAS DE TRANSPOSIÇÃO

Transposição é uma forma de encriptação onde troca-se a posição dos caracteres na mensagem. Por exemplo, pode-se reescrever o texto percorrendo-o por colunas, ou então definir o tamanho para um vetor de trocas e também uma ordem em que as trocas serão feitas. Por exemplo, em um vetor de tamanho 6 pode-se trocar o primeiro caracter pelo terceiro, o segundo pelo quinto e o quarto pelo sexto. Se a frequência dos caracteres for a mesma do idioma, tem-se substituição por transposição. Se for diferente, tem-se por substituição. Também é possível combinar substituição e transposição, ou vice e versa (SANTOS, EMER e AVER, 1996).

3.1.3 MÁQUINAS DE CIFRAGEM

As máquinas de cifragem baseiam-se em engrenagens com tamanhos diferentes e que giram a velocidades diferentes, obtendo uma substituição polialfabética com chave de 26n, onde n é o número de engrenagens.

3.2 CRIPTOANÁLISE

Criptoanálise é o processo de tentativas para descobrir *X* ou *K*, ou ambos. A estratégia usada pelo criptoanalista depende da natureza do esquema de encriptação e a avaliação da informação para o criptoanalista.

A maior dificuldade é quando tudo o que é avaliado é o próprio texto cifrado. Em alguns casos, nem sempre o algoritmo de encriptação é conhecido, mas em geral pode-se assumir que o oponente conhece o algoritmo usado para a encriptação. Um possível ataque sobre essas circunstâncias é a aproximação da força bruta das tentativas de todas as chaves possíveis. Se o espaço da chave é muito grande, isto se torna impraticável. Assim, o oponente precisa confiar em uma análise do próprio texto cifrado, geralmente aplicando vários testes estatísticos para isso. Para usar essa aproximação, o oponente precisa pelo menos ter alguma idéia geral do tipo de texto que está criptografado.

3.3 MÉTODOS DE CRIPTOGRAFIA

A segurança da criptografia não está no algoritmo utilizado para encriptar ou decriptar os dados, e sim na chave. Para implementar esta questão, existem dois métodos, um que utiliza o conceito de chave secreta, e outro que utiliza o conceito de chave pública.

3.3.1 CRIPTOGRAFIA COM CHAVE SECRETA

Os métodos de criptografia que utilizam a mesma chave para codificação são classificados como simétricos ou baseados em chave secreta ou chave única. Um exemplo desse método seria substituir as letras de um texto pela n-ésima letra após sua posição no alfabeto, nesse caso o texto criptografado produzido para um mesmo texto normal variará de acordo com o valor de n, que é a chave utilizada nos procedimentos de codificação do método.

O quadro 1 ilustra o exemplo citado anteriormente, onde *X* é o texto no formato original, *Y* é o texto criptografado e *K* representa a chave que foi utilizada, no caso, 2 posições na sequência do alfabeto (26 letras).

```
x = "texto qualquer"
y = "vgavq sxcnsxgt"
k = 2
```

QUADRO 1 – Exemplo Simples de Criptografia com Chave Secreta

Uns dos algoritmos mais conhecidos de chave secreta é o *Data Encryption Standard* (DES), que cifra blocos de 64 bits (8 caracteres) usando uma chave de 56 bits mais 8 bits de paridade.

Um passo de cifragem do DES tem dois objetivos básicos, a difusão e a confusão. A difusão visa eliminar a redundância existente na mensagem original, distribuindo-a pela mensagem cifrada. O propósito da confusão é tornar a relação entre a mensagem e a chave tão complexa quanto possível. O DES pode ser quebrado pelo método da força bruta, tentando-se todas as combinações possíveis de chave.

3.3.2 CRIPTOGRAFIA COM CHAVE PÚBLICA

Esse método de criptografia baseia-se na utilização de chaves distintas, uma para a codificação (*E*) e outra para a decodificação (*D*), escolhidas de forma que a derivação de *D* a partir de *E* seja, em termos práticos, senão impossível, pelo menos muito difícil de ser realizada. Uma vez respeitada essa condição, não há razão para não tornar a chave *E* pública, simplificando bastante a tarefa de gerenciamento das chaves. Os métodos de criptografia que exibem essas características são denominados assimétricos, ou baseados em chave pública.

Uma mensagem cifrada com uma chave pública só pode ser decifrada pela outra chave secreta com a qual está relacionada. O processo é ilustrado na figura 3. A chave usada para cifrar recebe o nome de “chave pública” pois ela deve ser publicada e amplamente divulgada pelo seu possuidor, fazendo com que qualquer pessoa possa lhe enviar mensagens cifradas. Já a chave usada para decifrar as mensagens, deve ser mantida em sigilo. Geralmente, os

usuários deste tipo de criptografia publicam suas chaves públicas em suas home pages, assinaturas dos e-mails, etc.

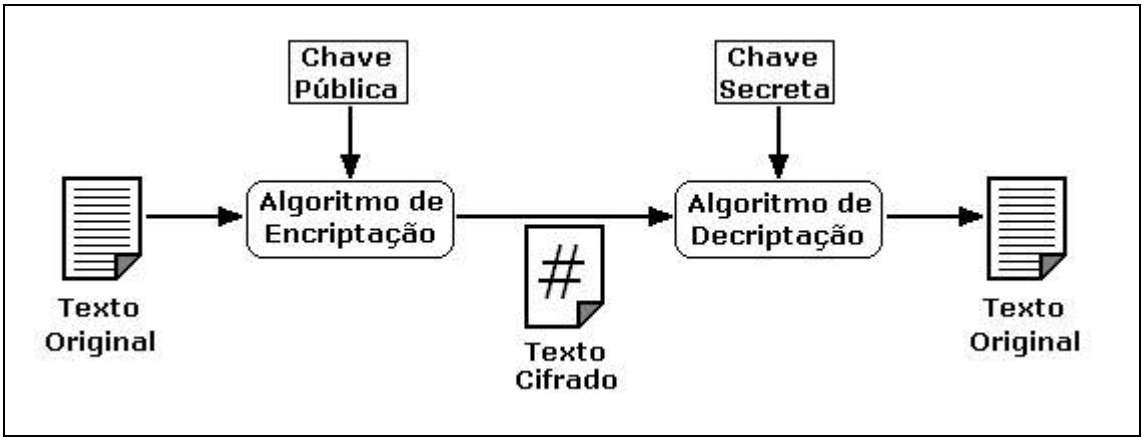


FIGURA 3 – Processo de Criptografia por Chave Pública

A chave de ciframento é publica, sem que haja quebra de segurança. Dessa forma cada usuário tem uma chave de ciframento, de conhecimento público, e outra de deciframento, secreta. Se um usuário **A** deseja enviar uma mensagem para um usuário **B**, ele utiliza a chave de ciframento pública PB, este de posse de sua chave de deciframento SB decodifica a mensagem.

Segundo Lucchesi (1986), o principal algoritmo que utiliza este sistema é o RSA, anacrônico de seus autores Rivest, Shamir e Adleman. Sua segurança baseia-se na intratabilidade da fatoração de produtos de dois primos. Este algoritmo é comentado na seção 3.4.3.

3.3.3 CRIPTOGRAFIA SIMÉTRICA X ASSIMÉTRICA

A diferença entre os métodos de criptografia simétricos e assimétricos é que, nos simétricos, a chave K usada no procedimento de codificação é igual a chave K' usada no procedimento de decodificação, isto é, $K = K'$; e nos assimétricos $K <> K'$.

Analisando os dois métodos, pode-se observar que a criptografia por chave pública tem a vantagem sobre a chave privada no sentido de viabilizar a comunicação segura entre pessoas comuns. Com a chave pública também acaba o problema da distribuição de chaves existente na criptografia por chave secreta, pois não há necessidade do compartilhamento de uma mesma chave, nem de um pré-acordo entre as partes interessadas. Com isto o nível de

segurança é maior. A principal vantagem da criptografia por chave secreta está na velocidade dos processos de cifragem/decifragem, pois estes tendem a ser mais rápidos que os de chave pública (MACÊDO e TRINTA, 1998).

3.4 ALGORITMOS DE CRIPTOGRAFIA

Serão vistos a seguir alguns dos principais algoritmos de criptografia para chave secreta e chave pública.

3.4.1 DATA ENCRYPTION STANDARD (DES)

A história do DES começou em 1973, quando o *National Bureau of Standards* (NBS) americano solicitou proposta para desenvolvimento de um algoritmo criptográfico. Entre as propostas apresentadas foi escolhida a desenvolvida pela IBM que foi validada pelo governo americano e recomendado como mais do que adequado para aplicações civis (LUCCHESI, 1986).

O DES é um ciframento composto que cifra blocos de 64 bits (8 caracteres) em blocos de 64 bits, para isso ele se utiliza de uma chave composta por 56 bits mais 8 bits de paridade (no total são 64 bits também). Assim, para cada chave, o DES faz substituição monoalfabética sobre um alfabeto de $2^{64} \cong 1{,}8 \times 10^{19}$ letras. A rigor, é uma substituição monoalfabética, mas as técnicas publicadas de quebra de substituições monoalfabéticas se aplicam apenas a alfabetos pequenos (LUCCHENSI, 1986).

O algoritmo de codificação é parametrizado por uma chave *K* de 56 bits e possui 19 estágios diferentes (Figura 4), o primeiro estágio realiza uma transposição dos bits do texto (modificação da posição) independente da chave. O último estágio realiza uma transposição inversa a do primeiro estágio. O penúltimo estágio realiza a permutação dos 32 bits mais significativos com os 32 bits menos significativos do bloco de dados. Os outros 16 estágios são funcionalmente idênticos (executam a mesma transformação nos dados, transposições e substituições), porém são parametrizados por chaves *Ki*, obtidas pela aplicação de funções, que variam de um estágio *i* para outro, nos bits da chave *K* original (Figura 5). O método permite que a decodificação seja feita com a mesma chave usada na codificação, através da execução das mesmas etapas na ordem inversa.

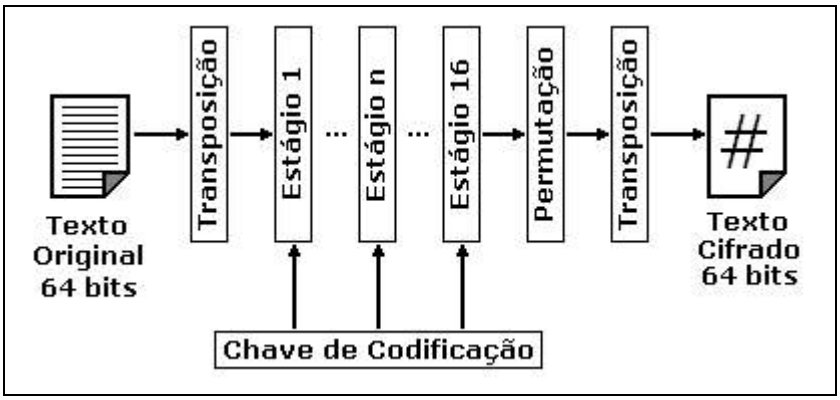


FIGURA 4 – Processo de Criptografia utilizando DES

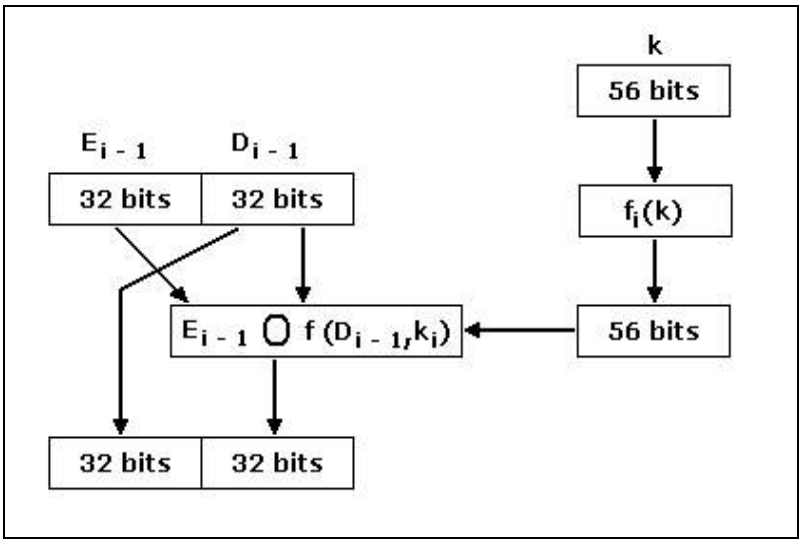


FIGURA 5 – Processo de Criptografia das chaves Ki do DES

Segundo Soares et al (1995), o principal problema do método DES, e de todos os algoritmos de criptografia simétricos, é a exigência que o transmissor e o receptor de uma mensagem conheçam a chave secreta e única usada na codificação e na decodificação. O acordo em torno da chave secreta entre o transmissor e receptor, quando eles estão distantes um do outro, não é um problema trivial, pois envolve a transmissão da chave e nem sempre é possível garantir que essa transmissão não seja interceptada. Se for interceptada, o responsável pelo ataque poderá ler todas as mensagens que serão criptografadas utilizando a referida chave "secreta". Um complicador para este problema é o fato de que em um sistema com *n* usuários, comunicando-se dois a dois, são necessárias (*n.(n-1))/2* chaves secretas. A tarefa de gerar, transmitir e armazenar chaves em um sistema de segurança é denominada *gerenciamento de chaves*.

O algoritmo DES possui uma variação chamada Triple DES (ou apenas 3DES). O 3DES, utiliza o DES em três ciframentos sucessivos, podendo empregar uma versão com duas ou com três chaves diferentes (112 ou 168 bits). É seguro, porém muito lento para ser um algoritmo padrão.

3.4.2 ADVANCED ENCRYPTION STANDARD (AES)

Em outubro de 2000, o *National Institute of Standards and Technology* (NIST) anunciou um novo padrão de uma chave secreta de cifragem, escolhido entre 15 padrões candidatos. Este novo padrão pretendia substituir o velho algoritmo DES, cujo tamanho das chaves estava se tornando muito pequeno. O Rijndael - um nome comprimido, originário dos seus inventores Rijmen e Daemen - foi escolhido para se tornar o novo padrão, que se chamou *Advanced Encryption Standard* (AES).

Este sistema de encriptação é dito ser um "bloco" de cifragem à medida que as mensagens são encriptadas em blocos inteiros, com unidades de 128 bits. Existem múltiplas idéias que propõem a utilização de chaves com 128, 192, 256 bits. Fazendo uma comparação, o DES encripta blocos de 64 bits com uma chave de 56 bits, e o DES triplo, normalmente, encripta blocos de 64 bits com uma chave de 112 bits.

Segundo Hinz (2000), o algoritmo Rijndael diferencia-se da maioria dos outros algoritmos usados atualmente, pois não usa uma estrutura do tipo *Feistel* na sua fase de rotação. Numa estrutura *Feistel* os bits de estado intermediário são transpostos em uma outra posição sem serem alterados. No Rijndael, a fase é composta de transformações uniformes inversíveis distintas chamadas de *layers*.

O Rijndael é um cifrador de substituição e permutação. Ele cifra um texto claro de 128 bits em um texto cifrado de 128 bits, usando para isso *n* fases, onde cada resultado intermediário entre as transformações é chamado de “estado”. O número de fases *n* definido para a cifra depende do tamanho de bloco e do tamanho de chave que estão sendo utilizados. O menor números de fases é 10 (correspondendo ao tamanho de bloco de 128 bits e tamanho de chave de 128 bits), sendo este limite válido para todos os tamanhos de blocos e de chaves.

Cada fase consiste na aplicação seguida das transformações de: substituição (*ByteSub*), deslocamento de linhas (*ShiftRow*), mesclagem de colunas (*MixColumn*) e adição de chaves (*AddRoundKey*).

3.4.3 RIVEST, SHAMIR E ADLEMAN (RSA)

O RSA é um sistema de criptografia de chave pública tanto para cifrar quanto para autenticação de dados, foi inventado em 1977 por Ronald **R**ivest, Adi **S**hamir e Leonard **A**dleman, pesquisadores do MIT. A principal desvantagem de se utilizar algoritmos de chave pública é a velocidade, o método de chave pública é muito mais lento na cifragem do que o método de chave secreta (PUTTINI, 2000).

O algoritmo RSA faz uso de expressões com exponenciais e o texto é cifrado em blocos com cada bloco tendo um valor binário menor que um número *n*, ou seja, o tamanho do bloco tem que ser menor ou igual a $log_2 (n)$. Na prática tem-se que o tamanho do bloco é 2^k, sendo que $2^k < n <= 2^{k+1}$.

Segundo Puttini (2000), para usar o RSA deve-se tomar dois números primos *p* e *q*, e definir o módulo *n* como sendo $n = p * q$. Os fatores *p* e *q* devem ser mantidos em segredo. Em seguida deve-se estabelecer um inteiro qualquer *d* que seja relativamente primo ao inteiro $(p–1)*(q-1)$. Deve-se obter também um número *e* na faixa $1 <= e <= (p-1)*(q-1)$ tal que $(e*d) \bmod ((p-1)*(q-1)) = 1$, essa última equação indica que o resto da divisão de $(e*d)$ por $(p-1)*(q-1)$ é igual a 1. Sendo assim é conhecida a chave pública, formada pelo par de inteiros (*e*,*n*). Representar a mensagem *M* que será transmitida como um inteiro na faixa *{1, ..., n}*, se a mensagem M for muito grande, deve-se colocá-la em blocos. Criptografar *M* num criptograma *C* da seguinte forma: $C = M^e \pmod n$. Decriptografar *M* utilizando a chave privada *d* e a fórmula $D = C^d \pmod n$.

O remetente deve conhecer o valor de *e*, e somente o destinatário deve conhecer o valor de *d*, porém ambos devem saber o valor de *n*. Portanto, é um algoritmo de criptografia de chave pública com uma chave pública *KU = {e,n}* e uma chave privada *KR = {d,n}*.

4 ESTEGANOGRAFIA

4.1 HISTÓRIA

A palavra esteganografia vem do grego: *steganos*: oculto, secreto e *graphy*: escrita.

No decorrer do tempo, muitos métodos têm sido usados para ocultar informação. Um dos primeiros documentos descreve o uso da esteganografia desde a Grécia antiga, onde os textos eram escritos em taboas cobertas de cera, e só após derreter a cera era possível ler a mensagem.

Outro método era raspar a cabeça dos mensageiros, tatuar a mensagem que se queria transmitir, esperar o cabelo crescer e então enviar o mensageiro ao seu destino, onde novamente seria raspada a sua cabeça para poder ler a mensagem tatuada.

Ou então, escrever a mensagem no papel utilizando vinagre como tinta, após secar não se percebe a presença de qualquer texto, e para ler a informação teria apenas que molhar o papel com água, assim o texto escrito com o vinagre ficaria em destaque tornando possível a sua leitura.

Outra forma ainda é esconder uma mensagem em outra, por exemplo, escrever a mensagem sigilosa utilizando a primeira letra de cada palavra de uma outra mensagem qualquer.

Na Segunda Guerra Mundial, começou a ser utilizado o microponto. Uma mensagem secreta era fotograficamente reduzida a medida de um ponto e este podendo ser o ponto de uma letra “i” de uma outra mensagem qualquer.

Outro exemplo de esteganografia seria tirar uma fotografia com uma mensagem escondida na própria imagem, por exemplo, fotografar pessoas que, sem levantar suspeito formem uma mensagem com as mãos.

4.2 DEFINIÇÃO

A palavra esteganografia literalmente significa “escrita encoberta”. Isto inclui uma vasta coleção de técnicas e métodos para comunicações secretas que ocultam a existência de uma mensagem.

A esteganografia é a arte de comunicar-se secretamente, ocultando uma mensagem sigilosa dentro de outra informação sem importância, de maneira que não exista forma de detectar que há uma mensagem escondida. Na computação essa outra informação pode ser um arquivo de som, imagem ou texto.

Arquivos como os de imagem e som possuem áreas de dados que não são usadas ou são pouco significativas. A esteganografia tira proveito disso, trocando essas áreas por informação.

4.3 ESTEGANOGRAFIA X CRIPTOGRAFIA

A diferença principal é que na criptografia há presença de uma mensagem mesmo estando na forma codificada, enquanto na esteganografia não há.

Uma mensagem criptografada permite ser detectada, interceptada e até mesmo modificada se for violado o sistema de encriptação. Quando se usa apenas criptografia, a informação pode ser ilegível, mas é óbvio que ali existe uma mensagem, que ali existe um segredo.

Se a informação estiver apenas oculta e não criptografada, torna-se mais trabalhoso o processo para identificá-la, buscando todos os arquivos suspeitos de conter informação oculta e varrendo-os até encontrar algum tipo de sequência lógica de dados que possa resultar numa mensagem.

Atuando juntos, torna-se praticamente impossível detectar a mensagem original. A mensagem é encriptada por algum algoritmo de criptografia que utiliza chave secreta ou chave pública, o que já dificulta bastante o processo de detecção, e através de algum método de esteganografia, essa informação codificada é escondida dentro de um arquivo que aparentemente é inofensivo. Digamos que mesmo ciente de que um arquivo possua uma mensagem oculta, teria que varrer bit a bit para procurar por essa mensagem, e mesmo encontrando a técnica de esteganografia que foi utilizada, ainda sobraria saber a chave de criptografia para então poder decriptografar a informação codificada. Não é impossível, mas impraticável.

4.4 ESTEGANOGRAFIA COM IMAGENS

A esteganografia evoluiu bastante a partir de 1990 com a chegada dos computadores. A tecnologia digital possibilitou novas formas de aplicar as técnicas de esteganografia. Uma delas, a mais utilizada, é esconder informações em imagens digitais, utilizando os bits menos significativos ou áreas não utilizadas da imagem.

Informações podem ser escondidas de várias maneiras diferentes utilizando imagens como objeto de cobertura. Cada uma dessas técnicas pode ser aplicada para armazenar informações em imagens:

a) inserção no bit menos significativo (LSB): utiliza o bit menos significativo de cada pixel da imagem para ocultar a informação;
b) técnicas de filtragem e mascaramento: oculta a informação através da marcação de uma imagem, de modo similar à marca d'água aplicada em papel;
c) algoritmos de transformações: utilizam o brilho, saturação e compressão das imagens.

O método de inserção no bit menos significativo é provavelmente uma das melhores técnicas de esteganografia em imagem. Este método é detalhado na seção 4.4.2.

4.4.1 ARQUIVOS DE IMAGENS

Para o computador, uma imagem é uma matriz de números que representam intensidades de cores em vários pontos.

Existem diferentes formatos de arquivos para o armazenamento de imagens, que envolve basicamente três elementos principais: a forma como a imagem está representada, o tipo de compactação empregado e o cabeçalho contendo as informações desta imagem. Um mesmo tipo de arquivo pode inclusive permitir o armazenamento de diferentes classes de imagens e também permitir a utilização de vários métodos de compactação.

Segundo Casacurta et al (1998), uma imagem pode ocupar muito espaço em memória ou disco, por isso é bastante comum na computação o emprego de técnicas de compactação de dados. As técnicas de compactação de dados, de uma forma geral, buscam se aproveitar de uma possível repetição de informações para reduzir o tamanho do arquivo através de sua recodificação de forma otimizada.

Os métodos de compactação de imagens mais utilizados são:

a) linear: sem compactação;
b) *Run-Length Encoding* (RLE): compacta repetições seguidas de pixels iguais;
c) *Lempel-Ziv-Welch* (LZW): algoritmo com alto grau de compactação;
d) JPEG: proposta de padrão de compactação de imagens surgida em função da *High-Definition TeleVision* (HDTV).

Uma imagem é composta por um conjunto de pontos, denominados *pixels* (*Picture Elements*) ou *dots*. Estes *pixels* estão dispostos na tela do computador formando uma matriz de pontos que é denominada de *bitmap* ou "Mapa de Bits". Este mapa de bits é um reticulado onde cada elemento da matriz possui uma informação referente à cor associada àquele ponto específico. Uma determinada imagem possuirá também uma "resolução" associada a ela, que é o número de elementos que esta imagem possui na horizontal e na vertical. Cada elemento da imagem possuirá uma localização, que é definida pela suas coordenadas (CASACURTA et al, 1998).

Uma imagem típica no formato *bitmap* tem 640 x 480 *pixels* e 256 cores (ou 8 bits por *pixel*), com um tamanho de mais ou menos 300 Kb. Porém hoje em dia a maioria das imagens possui qualidade de 24 bits, ou seja, 16 milhões de cores. Isso favorece a prática de esteganografia, pois possibilita que uma maior quantidade de informação seja escondida ao longo dessa imagem.

Como citado acima, imagens de 24 bits são ideais para esconder informações, pois são imagens grandes no tamanho armazenado. Estas imagens utilizam 3 bytes por cada *pixel* para representar um valor de cor. Estes bytes podem ser representados em decimal, neste caso os valores variam de 0 a 255. Cada um destes bytes representa uma cor: roxo, verde e azul. Por exemplo, um *pixel* em branco teria 255 de roxo, 255 de verde e 255 de azul.

Portanto, uma imagem é uma matriz de pontos ou *pixels*, com uma determinada resolução horizontal (eixo X) e vertical (eixo Y), onde para cada ponto desta matriz tem-se uma cor associada. A cor pode ser obtida de forma direta ou através de uma tabela de acesso indireto, denominada de "tabela de palette", que serve para possibilitar o acesso a um grande número de cores em um dispositivo com características gráficas limitadas. O princípio de funcionamento da tabela de palette é o de que na maioria das situações o usuário não precisa utilizar todo o conjunto de cores disponíveis em termos de hardware de uma forma simultânea, ou seja, define um subconjunto de cores com apenas as cores que estão em uso para aquela imagem (CASACURTA et al, 1998).

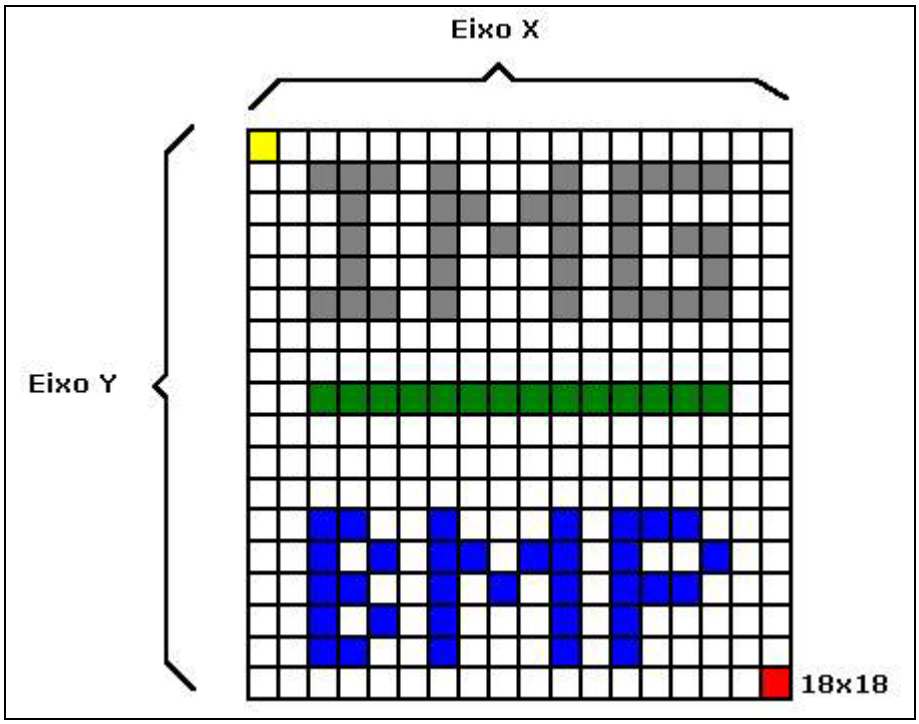


FIGURA 6 – Exemplo de uma imagem no formato *bitmap*

No exemplo ilustrado na figura 6, cada quadrado representa um *pixel* ou um "ponto de cor", formando então uma imagem de 18 x 18 *pixels*. A imagem é lida iniciando no primeiro *pixel* da esquerda para a direita e de cima para baixo.

Segundo Casacurta et al (1998), o ser humano possui em seu sistema visual três tipos de sensores capazes de identificar três faixas diferentes de "espectros de energia". Estas faixas correspondem às tonalidades de vermelho, verde e azul. Logo o ser humano vê na realidade a combinação resultante da mistura destas três cores básicas.

O sistema de cores mais utilizado nos computadores é usualmente o RGB (*Red-Green-Blue*), onde o que se faz é controlar a intensidade da geração destas três cores básicas. Ao definirmos uma determinada cor em um computador, o que está se especificando na realidade é a intensidade (valor associado) aos emissores R, G e B. Através de testes realizados com o ser humano chegou-se a conclusão que a utilização de 256 variações diferentes de intensidade em cada uma das cores básicas é capaz de gerar um número de cores superior a capacidade visual do ser humano, ou seja, fica praticamente impossível de distinguir entre duas cores "vizinhas". No sistema RGB, o valor (0,0,0) equivale a cor preta com intensidade zero nas três componentes. O valor (255,255,255) equivale a cor branca onde as três componentes estão presentes com a sua intensidade máxima. As diferentes combinações entre RGB serão capazes de gerar qualquer tipo de cor, sendo que se as três componentes tiverem sempre valores exatamente iguais teremos definida uma escala de tons de cinza do preto ao branco, é a chamada *gray scale* (CASACURTA et al, 1998).

Imagens do formato *Graphics Interchange Format* (GIF) utilizam o método LZW de compactação. Este método compacta uma série de símbolos repetidos em um único símbolo multiplicado pelo número de vezes que ele aparece. A informação RGB de cada pixel é substituída por um índice do mapa de cores. Uma limitação do formato GIF é permitir apenas imagens de 256 cores.

4.4.2 INSERÇÃO NO BIT MENOS SIGNIFICATIVO (LSB)

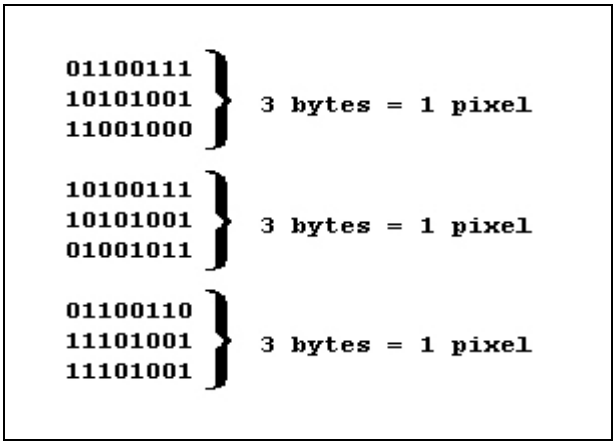
O método *Last Significant Bit* (LSB) é o mais comum utilizado para armazenar informação em imagens digitais. Consiste em utilizar o bit menos significativo de cada *pixel* (ou de cada cor) da imagem, para ocultar a mensagem.

Embora as técnicas de LSB consigam esconder os dados aos olhos humanos, elas podem ser facilmente destruídas computacionalmente utilizando algoritmos de compressão com perdas de dados. Estes algoritmos selecionam apenas as partes mais significativas do objeto, ou seja, os bits menos significativos têm uma chance bem menor de serem selecionados. Por exemplo, se a mensagem for escondida em uma imagem no formato *bitmap*, e esta for convertida para um formato JPEG, a informação é perdida (ROCHA, 2003).

Para esconder uma informação em uma imagem de 24 bits, onde cada *pixel* possui 3 bytes, utilizando o método de inserção do último bit significativo, pode-se armazenar 3 bits

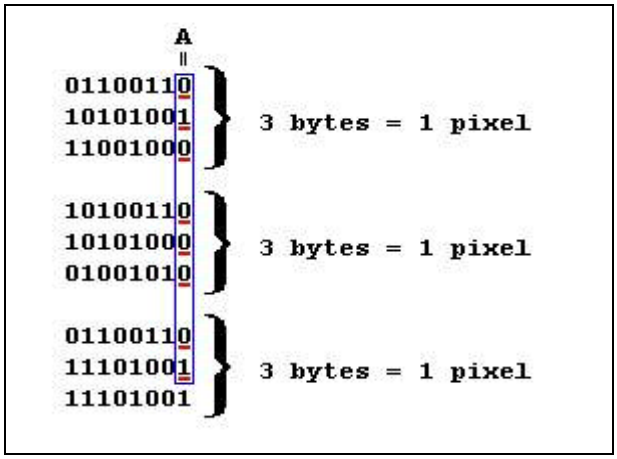
em cada *pixel*, utilizando 1 bit de cada byte desse *pixel*, ou seja, cada byte da informação a ser escondida irá ocupar 8 bytes da imagem. Por exemplo, em uma imagem de 24 bits com resolução de 1024 x 768 (786.432 *pixels*), que possui um tamanho real de 2.359.296 Kb (cada *pixel* ocupa 3 bytes), pode-se armazenar uma informação de 294.912 Kb (8 bytes da imagem para armazenar 1 byte do texto).

Se por exemplo em uma imagem tem-se os valores dos *pixels* como mostra o quadro 2.



QUADRO 2 – Exemplo de *pixels* de uma imagem

Suponha-se que se deseja armazenar a letra “A” nesta imagem. A letra “A” é o código 65 da tabela ASCII e, por tanto, seu valor binário é: 0 1 0 0 0 0 0 1. Colocando esse valor binário, bit a bit, ao longo da imagem, utilizando o bit menos significativo de cada byte do *pixel*, a imagem resultante seria como mostra o quadro 3.



QUADRO 3 – Exemplo de uso do método LSB

Os valores (bits) sublinhados são os que foram alterados, causando uma imperceptível alteração na cor do *pixel*, para o olho humano é impossível distinguir esta diferença. Esses valores combinados formam o byte que representa a letra “A”.

4.4.3 FILTRAGEM E MASCARAMENTO

São técnicas restritas a imagens em tons de cinza (*gray scale*). Consiste em esconder a informação através da marcação de uma imagem, de modo similar ao funcionamento das marcas d’água aplicadas em papel. Uma das vantagens dessa técnica é que pode ser aplicada em imagens que passam por métodos de compressão, devido ao fato da marca d’água ser integrada na imagem. Outra vantagem é que as técnicas de máscara e filtragem também passam despercebidas pelo sistema visual humano que não consegue distinguir algumas mudanças na imagem.

4.4.4 ALGORITMOS DE TRANSFORMAÇÕES

Os algoritmos de transformação geralmente trabalham com formas mais sofisticadas de manuseio de imagens como: brilho, saturação e compressão das imagens.

As técnicas de transformação tomam como aliado o principal inimigo da inserção LSB: a compressão. Por isso configuram-se como as mais sofisticadas técnicas de mascaramento de informações em imagens conhecidas.

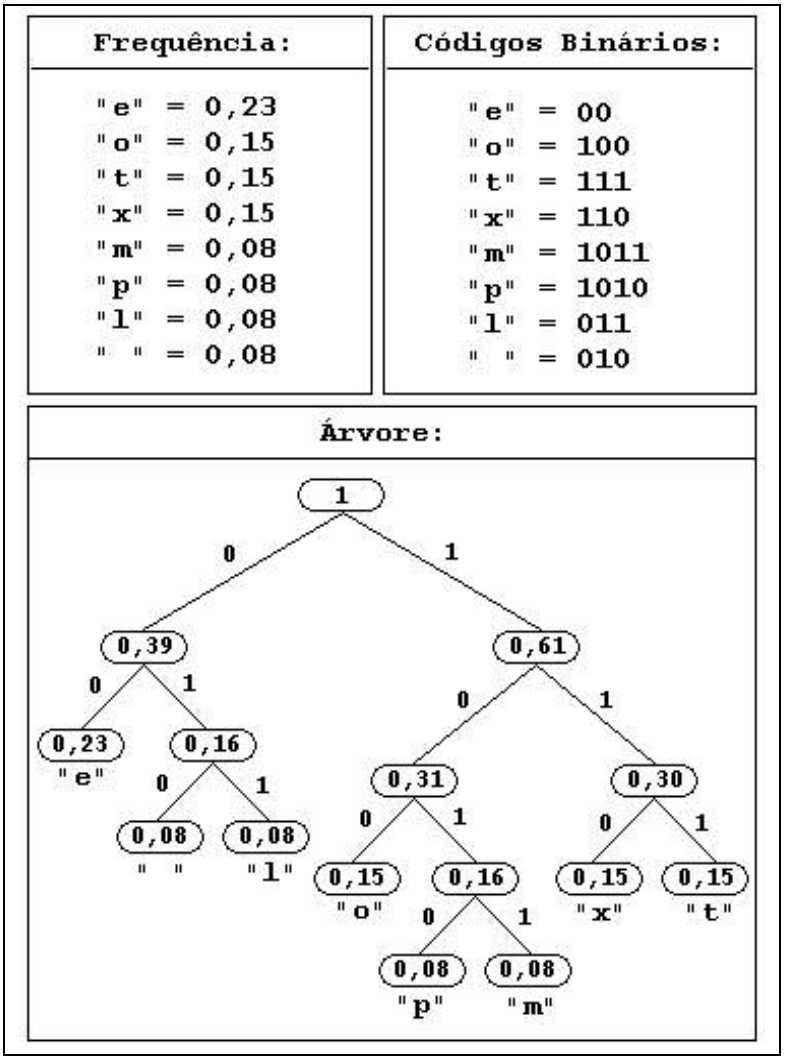
4.5 COMPRESSÃO DE HUFFMAN

O código de Huffman é um método estatístico, utilizado para compactação de dados. Os métodos estatísticos são aqueles que utilizam esquemas de codificação de tamanho variável aos seus símbolos ou grupos de símbolos. Utilizando os menores valores aos símbolos que aparecem mais, ou tem mais probabilidade de aparecerem.

Segundo Kunz (2003), a codificação de Huffman é uma forma de compressão de dados em que se representa cada um dos caracteres de um texto com códigos binários de comprimento variável. O tamanho do código varia conforme a freqüência com que ocorre no texto, atribuindo-se códigos menores aos caracteres mais freqüentes e maiores aos menos freqüentes.

O algoritmo consiste na construção de uma árvore, chamada árvore de prefixo, onde cada folha conterá um caracter e cada nodo conterá dois filhos, podendo ser o da esquerda com o valor “0” e o da direita com o valor “1”. Esta árvore é uma lista encadeada com os caracteres que aparecem no texto e suas respectivas freqüências, no início cada caracter representa uma árvore isolada. A criação é feita a partir dos dois caracteres, ou nodos, com menor frequência, cria-se então um nodo pai para eles, cujo valor será a soma das freqüências dos filhos, inclui-se esse nodo pai na lista e retiram-se os nodos filhos. Esse processo é repetido até existir apenas um nodo na lista, que é a raiz da árvore.

O quadro 4 ilustra como ficaria a árvore gerada para a frase “texto exemplo”, que possui 13 caracteres no total (8 distintos).



QUADRO 4 – Exemplo de uma árvore de Huffman

Toda a importância do algoritmo está na construção da árvore, de onde sairá a informação de como será codificado e decodificado o texto. Por isso, após comprimir o texto, é imprescindível gravar o conteúdo da árvore juntamente com o texto comprimido para tornar possível a descompressão (ERIGSON, 2003). Segundo Kunz (2003), no início da seqüência de bits do arquivo de imagem, é gravada a lista de caracteres da árvore em percorrimento pré-ordem. Para o exemplo visto no quadro 4, teríamos a seguinte representação para a árvore: "@@e@ l@@o@pm@xt", onde o caracter "@" representa cada nodo pai (nodo com dois filhos) e a árvore é percorrida iniciando pela raiz até cada folha, da esquerda para a direita.

Então, na parte inicial do arquivo de imagem é gravada a *string* com a representação da árvore, em binário, seguida de um caractere especial que indica o fim da árvore e o início da seqüência codificada.

Antes da codificação do texto, é acrescentado um outro caractere especial para indicar o final do texto, ou seja, indicar o fim da leitura dos bits da imagem na decodificação. Ele é incluído na árvore assim como os outros caracteres.

Depois de criada a árvore, os caracteres do texto são substituídos pelo respectivo código binário que é o caminho da raiz da árvore até as folhas. Isso garante que, percorrendo bit a bit, será encontrado apenas um caracter com aquela representação binária.

Para descomprimir o texto, deve-se ler a seqüência binária e andar na árvore a esquerda se for "0" e a direita se for "1" até encontrar uma folha, que é o caracter.

4.6 ÁREAS DE APLICAÇÃO

Uma importante aplicação moderna da esteganografia digital é como "marca d'água", mensagem oculta de direitos autorais, usada juntamente com uma "impressão digital" do produto, número de série ou conjunto de caracteres que autentica uma cópia legítima. A falta da "impressão digital" aponta violação de direito autoral e a ausência da "marca d'água" comprova o fato (DANTAS, 2002).

Outra aplicação é para a comunicação secreta. Hoje em dia existem alguns programas para que qualquer pessoa possa esteganografar mensagens, como por exemplo: *JPHSWin*, *S-Tools* e *WinHIP*. O usuário disponibiliza, por exemplo, uma imagem com uma mensagem codificada em um site e em seguida avisa os destinatários para que visitem esse site. Através do mesmo aplicativo, os destinatários decodificam a mensagem escondida na imagem.

A esteganografia é bastante útil também para guardar informações como autor, título, data, entre outras, em arquivos de imagens ou em outra mídia. Numa base de dados de imagens, por exemplo, palavras chave podem ser inseridas na imagem, facilitando uma eventual pesquisa através de mecanismos de busca.

A esteganografia era apenas um interesse das forças armadas, mas está ganhando popularidade entre as massas. Logo, todo o tipo de usuário de computador poderá por sua própria marca d'água sobre suas criações artísticas.

5 DESENVOLVIMENTO DO TRABALHO

Neste capítulo serão apresentadas a especificação e implementação do modelo proposto neste trabalho.

5.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

A ferramenta desenvolvida neste trabalho utiliza criptografia em conjunto com a esteganografia para ocultar mensagens de texto em imagens digitais. A criptografia é feita através do algoritmo *Rijndael* de chave secreta e a esteganografia através do método de inserção do último bit significante.

Serão dois tipos de usuários:

a) emissor: este terá como dados de entrada a mensagem a ser enviada, uma imagem do tipo *bitmap* e uma chave de criptografia, e como saída terá a imagem com a mensagem oculta e criptografada;
b) receptor: a partir da imagem esteganografada e de posse da chave criptográfica, terá como saída a mensagem que estava criptografada e oculta na imagem informada.

Deste modo é possível, através do protótipo, fazer uma comunicação segura, sem levantar suspeitas e sem que emissor e receptor estejam *on-line* no mesmo momento. Pode-se, por exemplo, disponibilizar uma imagem esteganografada em algum *site*, o receptor (ou receptores), conhecendo a localização da imagem, pode ler a mensagem a qualquer hora, e para outros usuários que acessassem aquele *site* seria apenas uma imagem como outra qualquer.

Para fazer a especificação deste protótipo foi utilizada uma metodologia orientada a objetos, a *Unified Modeling Language* (UML), usando como ferramenta o *Rational Rose*.

5.2 ESPECIFICAÇÃO

5.2.1 CASOS DE USO

O protótipo possui 2 casos de usos:

a) insere mensagem: responsável por entrar com a mensagem a ser transmitida, a

chave de criptografia e ocultar a mensagem na imagem;

b) consulta mensagem: responsável por ler a mensagem transmitida a partir de uma chave de criptografia dada pelo usuário;

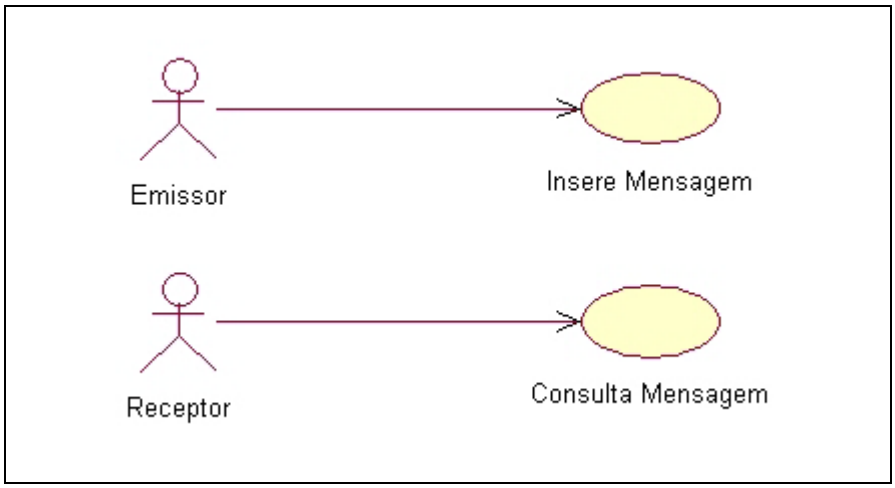


FIGURA 7 – Casos de uso

5.2.2 DIAGRAMA DE CLASSES

As classes utilizadas no protótipo são:

a) TImgEstegano: principal classe do protótipo, contém a imagem no formato *bitmap* e é responsável pela esteganografia, ou seja, ocultar e extrair a informação da imagem, seja ela criptografada ou não;

b) TBinUtil: contém rotinas de tratamento e conversão de binário. Não precisaria ser uma classe, porém, foi criada por causa dos atributos de status que servem para dar *feedback* ao usuário, pois os processos envolvidos podem ser demorados;

c) Tc3DBCrypt: classe que faz parte do componente c3DBCrypt do pacote de componentes ce3po, desenvolvido por terceiros (LIMA, 2003). Classe herdada da TCustomEdit do Delphi, responsável pela criptografia do texto utilizando o algoritmo simétrico AES. A figura 8 mostra a paleta de componentes ce3poDB onde está o componente c3DBCrypt;

FIGURA 8 – Paleta de componentes ce3po.

O Diagrama de Classes está demonstrado na figura 9.

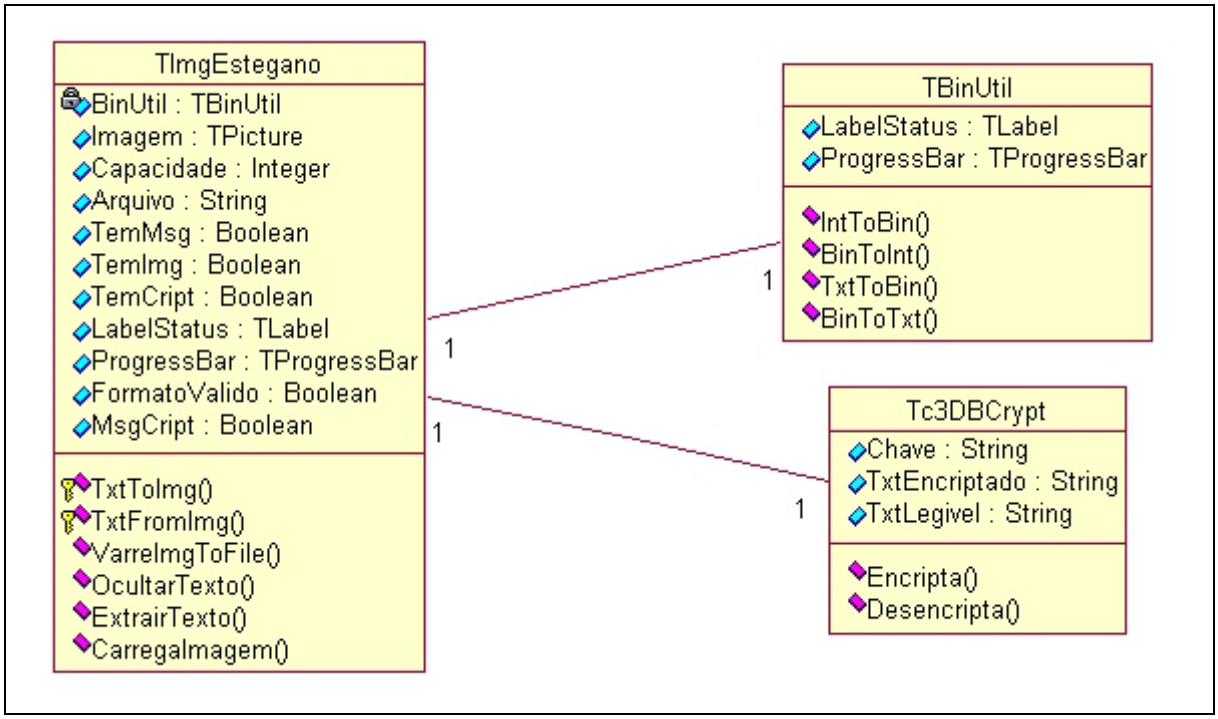


FIGURA 9 – Diagrama de classes

5.2.3 DIAGRAMAS DE SEQUÊNCIA

Os diagramas de seqüências representam, como o próprio nome diz, a seqüência em que as ações ocorrem dentro do protótipo. Eles demonstram como é feita a troca de mensagens entre as classes. Para cada caso de uso, há um diagrama de seqüência, conforme descrição a seguir.

5.2.3.1 INÍCIO

Este diagrama de seqüência é executado sempre que o protótipo é inicializado e uma imagem é carregada. Na inicialização é instanciado um objeto da classe TImgEstegano e um

objeto da classe Tc3DBCrypt. Ao abrir uma imagem, é executado o método "CarregaImagem" do objeto TImgEstegano, o qual instancia um objeto da classe TBinUtil.

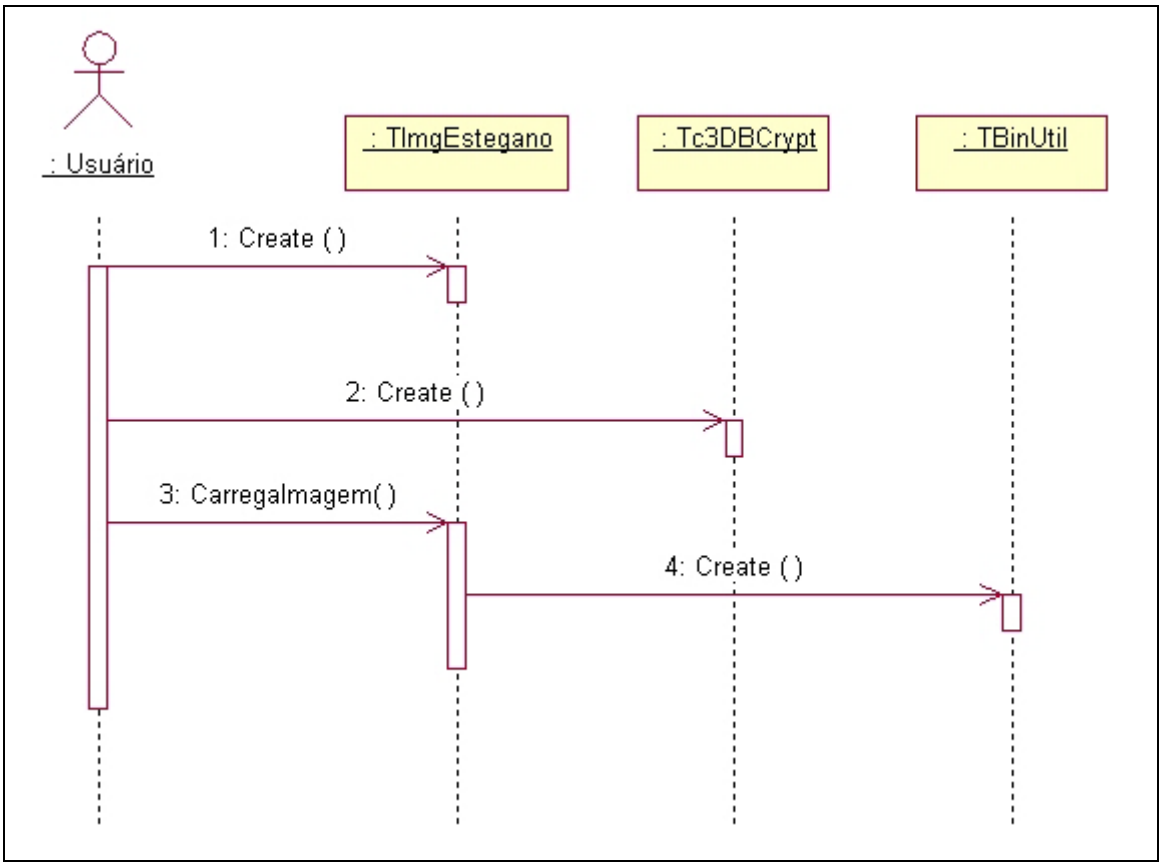


FIGURA 10 – Diagrama de sequência: "Início"

5.2.3.2 INSERE MENSAGEM

Este diagrama de sequência ocorre no evento para ocultar o texto na imagem, onde são executados os métodos "Encripta" do objeto Tc3DBCrypt, também criado na inicialização do protótipo, e "OcultarTexto" do objeto TImgEstegano. O método "OcultarTexto" chama internamente o método "TxtToImg" que processa o texto na imagem através das rotinas de conversão de binário do objeto TBinUtil: "IntToBin" para converter o valor inteiro que representa a cor do pixel, em binário e "BinToInt" para converter a informação já esteganografada novamente em inteiro.

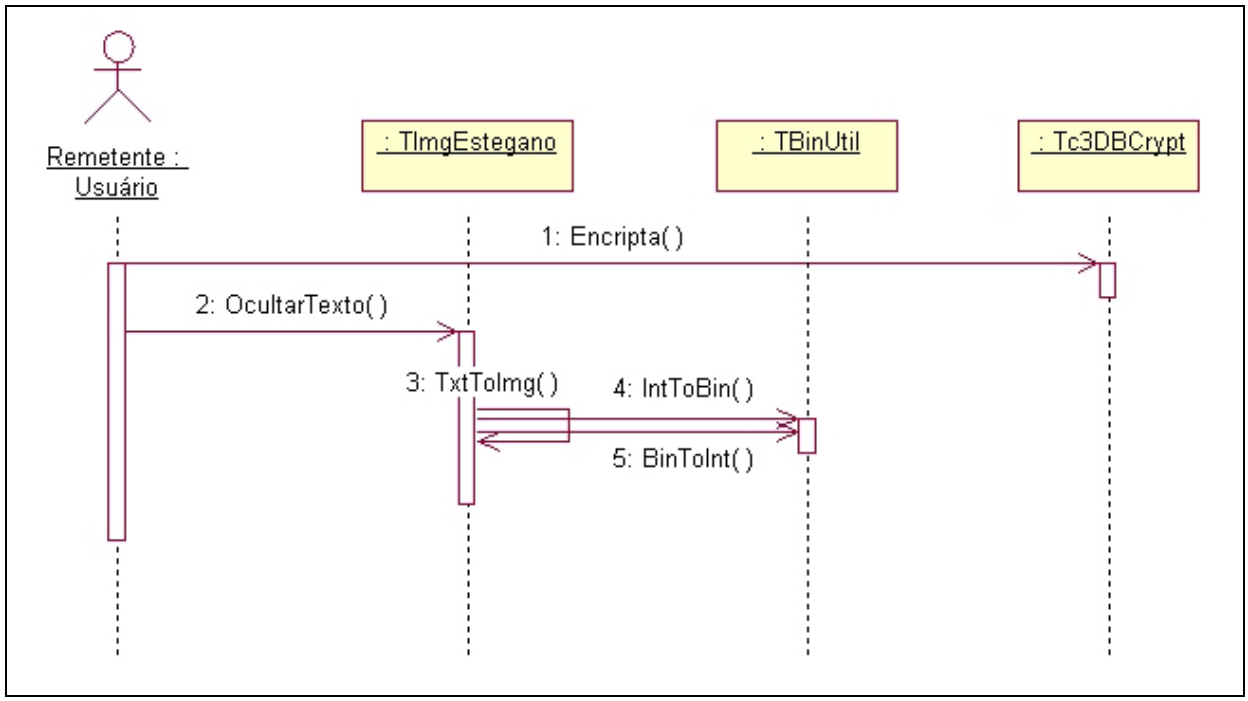


FIGURA 11 – Diagrama de sequência: “Insere Mensagem”

5.2.3.3 CONSULTA MENSAGEM

Este diagrama de sequência ocorre no evento para extrair o texto da imagem onde são executados os métodos “ExtrairTexto” do objeto TImgEstegano e “Desencripta” do objeto Tc3DBCrypt. O método “ExtrairTexto” chama internamente o método “TxtFromImg” que processa a imagem e retira o texto através das rotinas de conversão de binário (“IntToBin” e “BinToTxt”).

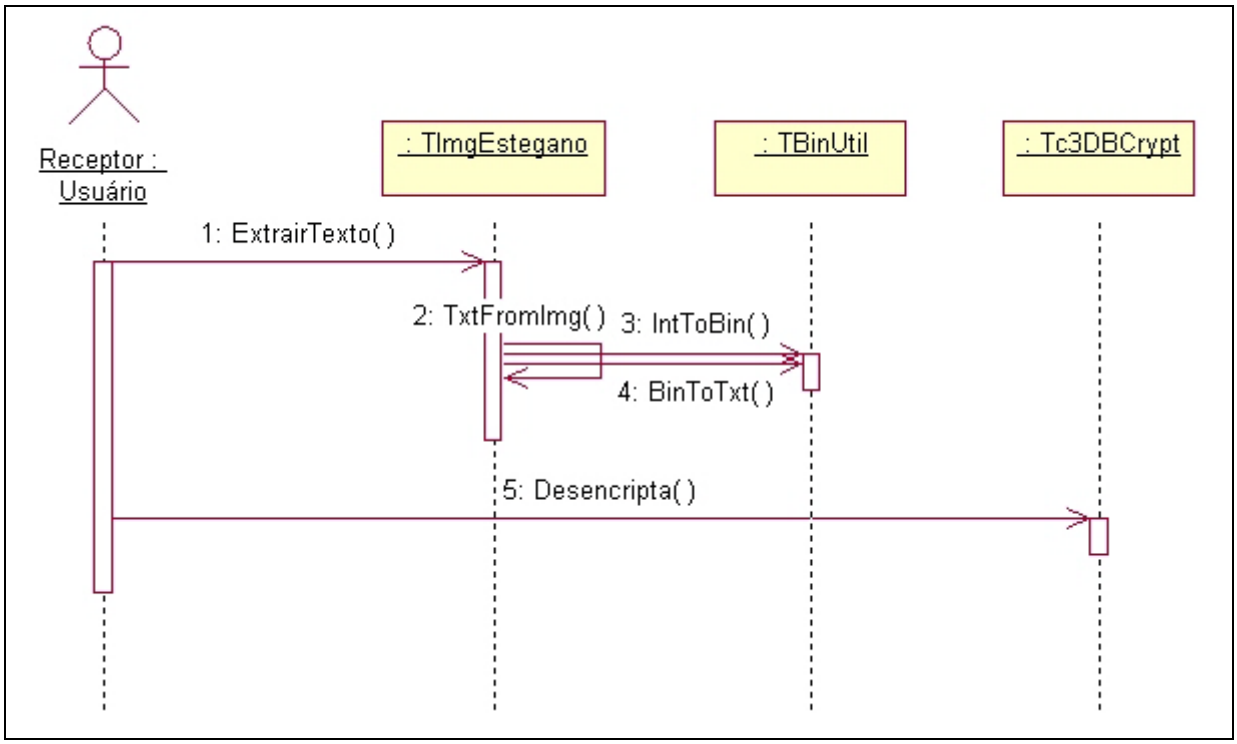


FIGURA 12 – Diagrama de sequência: “Consulta Mensagem”

5.3 IMPLEMENTAÇÃO

A seguir é apresentada a implementação do protótipo desenvolvido no ambiente de programação *Borland Delphi 7.0*.

5.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS

A implementação do protótipo foi dividida em duas partes:

a) criptografia: responsável por encriptar e decriptar a mensagem transmitida através da imagem. Esta parte do protótipo faz uso de um componente de terceiros, que utiliza o algoritmo simétrico *Rijndael* para criptografar os dados. O protótipo interage com este componente enviando como parâmetro a mensagem digitada pelo usuário e a chave secreta de criptografia, e recebe o texto criptografado;

b) esteganografia: responsável por ocultar e extrair a mensagem da imagem, podendo essa estar criptografada ou não. Esta parte do protótipo tem como dado de entrada uma imagem no formato *bitmap* de 24 bits. Pode ser considerado ainda como dado de entrada a informação a ser escondida e ter como saída a imagem esteganografada, ou a partir apenas da imagem, ter como saída a informação que

está oculta.

Os processos que envolvem a esteganografia, ou seja, ocultar e extrair a mensagem da imagem, fazem uso de rotinas de tratamento e conversão de binário. A imagem é lida *pixel* a *pixel*, cada *pixel* com 24 bits ou 3 bytes, de cada byte é lido o bit menos significante, o conjunto desses bits da origem a mensagem (cada 8 bits equivale a um caracter), podendo esses serem modificados quando se deseja ocultar uma mensagem.

Para qualquer mensagem esteganografada, o protótipo utiliza identificadores criados para marcar o início (“#INI#”) e o final (“#FIM#”) da mensagem, como mostra o quadro 5.

#INI#MENSAGEM#FIM#

QUADRO 5 – Identificadores de início e final de mensagem

Toda essa informação (identificador de início + mensagem + identificador de fim) é inserida ao longo da imagem iniciando no primeiro byte do primeiro *pixel* da imagem (da esquerda para a direita, de cima para baixo). É através destes identificadores que o protótipo identifica, ao abrir a imagem, a existência de esteganografia.

Além desses dois identificadores é utilizado também um outro para indicar que a mensagem está criptografada (“#CRP#”), ou seja, quando a mensagem estiver criptografada é substituído o identificador de início pelo de criptografia, permanecendo o de fim, como mostra o quadro 6.

#CRP#MENSAGEM_CRIPTOGRAFADA#FIM#

QUADRO 6 – Identificadores de criptografia e final de mensagem

Uma outra forma de se fazer isso sem utilizar identificadores é através de um gerador de número aleatório, responsável pela dispersão da mensagem na imagem. Esses números são gerados de forma pré-definida para que possam ser recuperados no processo de extração, ou seja, não haveria uma sequência de *pixels* alterados, a mensagem estaria em alguns *pixels* espalhados pela imagem.

Como pode ser visto na figura 13, a interface é simples e de boa usabilidade, atendendo conceitos de ergonomia. Possui um campo para a edição ou apenas para a visualização da mensagem transmitida, um campo para a edição da chave de criptografia, um painel com informações sobre a imagem, um painel para a visualização da mesma, uma barra de progressão, entre outros componentes.

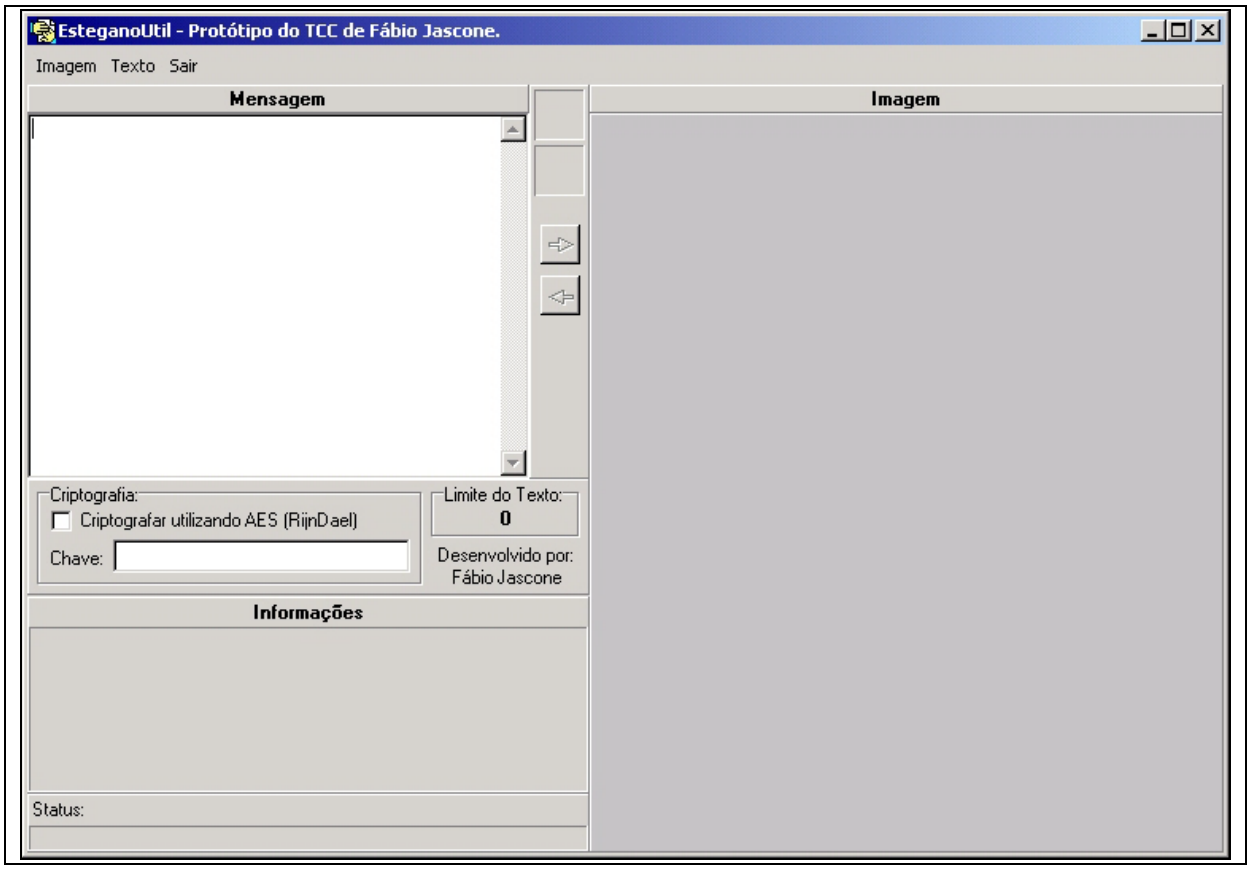


FIGURA 13 – Tela do protótipo

5.3.2 OPERACIONALIDADE DA IMPLEMENTAÇÃO

Para ser útil, o protótipo necessita de pelo menos uma imagem, pois dessa imagem pode ser extraída uma mensagem. Uma imagem que não possui esteganografia, necessita de uma mensagem para ser escondida. Os botões para extrair ou ocultar a mensagem ficam habilitados levando em consideração esses itens, ou seja, se não houver uma imagem aberta, ambos os botões ficam desabilitados; se houver uma imagem que possui esteganografia, o botão para extrair ficará habilitado e o botão para ocultar ficará habilitado apenas se houver uma mensagem informada; se a imagem não possuir esteganografia, o botão para extrair

ficará desabilitado e o botão para ocultar ficará habilitado apenas se houver uma mensagem informada.

Todos os processos realizados tanto para ocultar como para extrair uma mensagem de uma imagem, passam por uma barra de progressão, dando *feedback* ao usuário. Esse *feedback* é bastante util em casos de se ter mensagens longas em imagens grandes, onde os processos de esteganografia podem demorar um pouco.

Para um melhor entendimento sobre a execução do protótipo será demonstrada a execução dos casos de uso “Insere mensagem” e “Consulta mensagem”.

5.3.3 CASO DE USO “INSERE MENSAGEM”

O caso de uso “Insere mensagem” ocorre quando uma pessoa (usuário transmissor), oculta na imagem a mensagem que será enviada à outra pessoa (usuário receptor). O código do algoritmo para ocultar uma informação na imagem pode ser visto no anexo A.

Para isto, deve-se abrir uma imagem (menu “Imagem” opção “Abrir”) no formato *bitmap* de 24 bits e ao fazer isso, já é calculado e informado ao usuário o limite máximo de caracteres que é possível esteganografar nesta imagem, a partir da quantidade de bits menos significativos que a imagem possui. Isso é necessário para evitar que seja escondida uma informação além da capacidade da imagem. Esse valor varia de acordo com o tamanho da imagem. Pode-se digitar ou abrir um arquivo texto (menu “Texto” opção “Abrir”) com a mensagem a ser transmitida. Depois de informada a mensagem e a imagem é possível escolher se deseja que essa mensagem seja criptografada ou não. Escolhendo que sim, o usuário deve informar também uma chave secreta de criptografia. Esta chave deve ser de conhecimento também do receptor.

Todo esse processo é ilustrado na figura 14, onde se deseja ocultar a mensagem “Mensagem ultra secreta...” em uma imagem, utilizando criptografia e encriptando os dados com a chave “teste”.

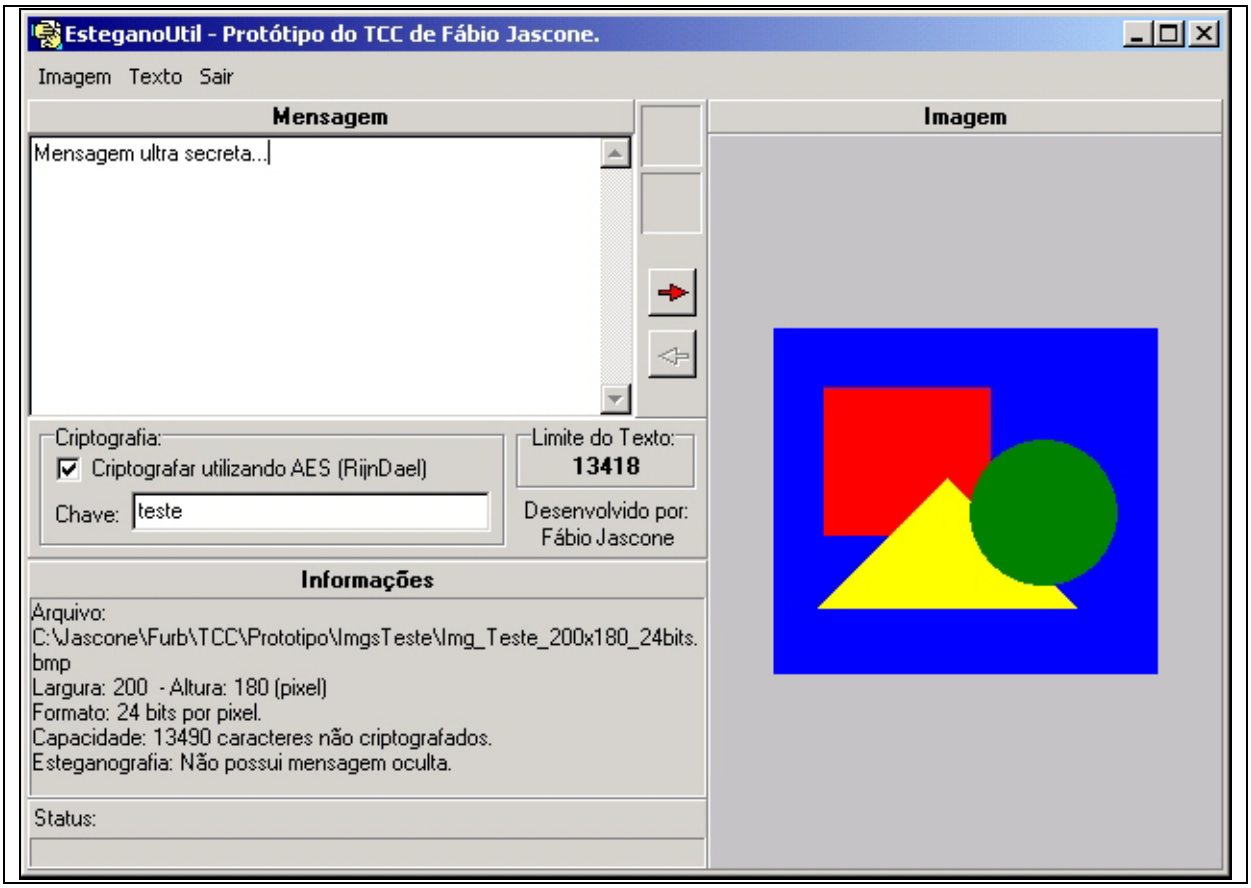


FIGURA 14 – Inserindo uma mensagem na imagem

Após isso, basta clicar no botão para ocultar a mensagem na imagem (botão com o ícone da "seta" apontando para a imagem), quando o processo terminar será apresentada uma mensagem avisando o usuário que o processo foi concluído e os indicadores de esteganografia e criptografia já estarão visíveis no centro superior da tela, como também o botão para extrair a mensagem da imagem (botão com o ícone da "seta" apontando para o campo da mensagem), como mostra a figura 15.

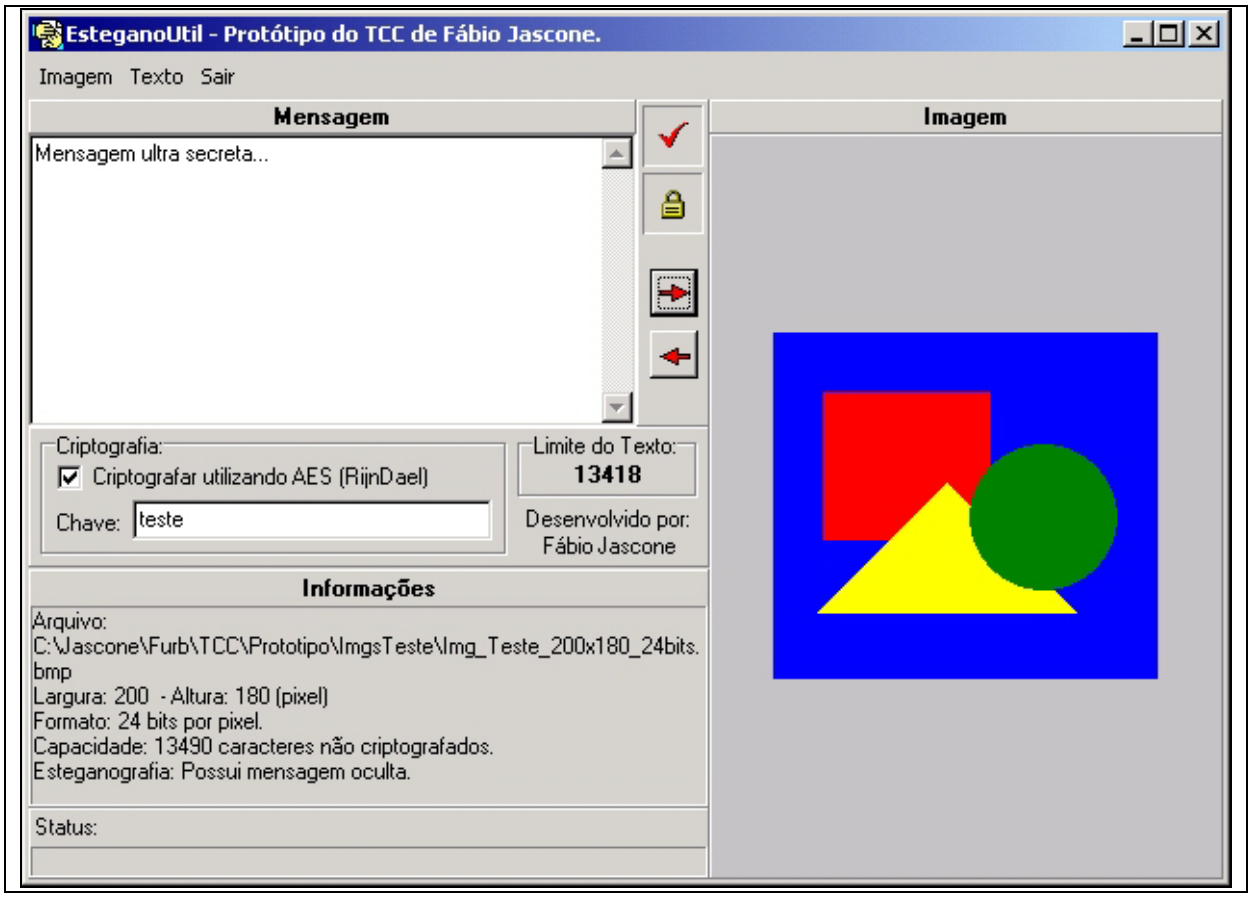


FIGURA 15 – Após inserir uma mensagem na imagem

Para concluir é necessário apenas salvar esta imagem (menu “Imagem” opção “Salvar”) que acabou de ser gerada, idêntica à imagem original.

5.3.4 CASO DE USO “CONSULTAR MENSAGEM”

O caso de uso “Consultar mensagem” ocorre quando o usuário receptor receber e abrir uma imagem (formato *bitmap* de 24 bits). O código do algoritmo para extrair uma informação da imagem pode ser visto no anexo B.

Ao fazer isso é possível perceber, através de indicadores na tela, se a imagem possui ou não mensagem oculta pelo protótipo e ainda se esta mensagem está ou não criptografada. Se estiver será exigida a chave de criptografia. Depois de aberta a imagem e informada a chave de criptografia (se necessário), deve-se clicar no botão para extrair a mensagem da imagem. Pode-se também salvar esta mensagem em um arquivo texto (menu “Texto” opção “Salvar”).

5.4 RESULTADOS E DISCUSSÃO

Para um melhor entendimento do funcionamento da ferramenta e para verificar os resultados obtidos, será feita uma comparação do código binário de uma imagem antes e depois de ser esteganografada.

Deseja-se ocultar a letra “A” em uma imagem. Como visto anteriormente, para qualquer mensagem, o protótipo utiliza um identificador de início (“#INI#” ou #CRP#) para marcar o início da mensagem, e um identificador de fim (“#FIM#”) para marcar o final da mensagem, concatenado os dois identificadores à mensagem, tem-se: “#INI#A#FIM#”.

Se a informação estiver criptografada, o processo não se altera. A letra “A” criptografada com uma chave qualquer representaria um conjunto de caracteres especiais codificados, e cada um desses caracteres é convertido para binário e processado na imagem da mesma forma como apresentado a seguir.

Convertendo a informação “#INI#A#FIM#” para binário fica: “00100011 01001001 01001110 01001001 00100011 01000001 00100011 01000110 01001001 01001101 00100011”. Essa sequência binária é que será inserida nos bits menos significativos de cada *pixel* da imagem. Para armazenar essa informação são necessários 88 bytes (11 caracteres x 8 bits), ou 30 *pixels*, sendo que do último *pixel* apenas 1 byte é utilizado.

A imagem utilizada como exemplo é relativamente pequena, 16x16 *pixels*, onde cada *pixel* possui uma cor distinta (as cores dos *pixels* não interferem em nada a esteganografia). Em uma imagem desse tamanho é possível esconder apenas 96 caracteres utilizando o método LSB.

A tabela 1 demonstra o processo de esteganografia para o exemplo acima apenas para os *pixels* que serão modificados. A primeira coluna mostra *pixel* a *pixel* (coluna, linha) que será modificado e a cor correspondente, a segunda coluna mostra os 3 bits da sequência binária da mensagem de texto que serão inseridos no *pixel* (1 bit para cada byte do *pixel*), a terceira coluna mostra a sequência binária original que representa a cor do *pixel* e a quarta coluna a sequência binária modificada, ou seja, com os 3 bits da entrada inseridos nos bits menos significativos de cada byte do *pixel* (bit em negrito).

Para extrair a informação lê-se a imagem *pixel* a *pixel*. A sequência binária que é formada com o conjunto dos bits menos significativos de cada byte do *pixel* compõe a mensagem (cada 8 bits representa 1 caracter). Essa leitura é feita até encontrar a sequência binária que representa o identificador "#FIM#" que indica o final da mensagem.

Tabela 1 – Exemplo de esteganografia em imagens com o método LSB.

x,y (*pixel*)	**Entrada**	**Sequência binária original**	**Sequência binária modificada**
1,1 (preto)	001	00000000 00000000 00000000	0000000**0** 0000000**0** 0000000**1**
2,1 (azul)	000	11111111 00000000 00000000	1111111**0** 0000000**0** 0000000**0**
3,1 (amarelo)	110	00000000 11111111 11111111	0000000**1** 1111111**1** 1111111**0**
4,1 (verde)	100	00000000 11111111 00000000	0000000**1** 1111111**0** 0000000**0**
5,1 (vermelho)	100	00000000 00000000 11111111	0000000**1** 0000000**0** 1111111**0**
6,1 (branco)	101	11111111 11111111 11111111	1111111**1** 1111111**0** 1111111**1**
7,1 (preto)	001	00000000 00000000 00000000	0000000**0** 0000000**0** 0000000**1**
8,1 (azul)	110	11111111 00000000 00000000	1111111**1** 0000000**1** 0000000**0**
9,1 (amarelo)	010	00000000 11111111 11111111	0000000**0** 1111111**1** 1111111**0**
10,1 (verde)	010	00000000 11111111 00000000	0000000**0** 1111111**1** 0000000**0**
11,1 (vermelho)	010	00000000 00000000 11111111	0000000**0** 0000000**1** 1111111**0**
12,1 (branco)	010	11111111 11111111 11111111	1111111**0** 1111111**1** 1111111**0**
13,1 (preto)	001	00000000 00000000 00000000	0000000**0** 0000000**0** 0000000**1**
14,1 (azul)	101	11111111 00000000 00000000	1111111**1** 0000000**0** 0000000**1**
15,1 (amarelo)	000	00000000 11111111 11111111	0000000**0** 1111111**0** 1111111**0**
16,1 (verde)	001	00000000 11111111 00000000	0000000**0** 1111111**0** 0000000**1**
1,2 (vermelho)	001	00000000 00000000 11111111	0000000**0** 0000000**0** 1111111**1**
2,2 (branco)	000	11111111 11111111 11111111	1111111**0** 1111111**0** 1111111**0**
3,2 (preto)	110	00000000 00000000 00000000	0000000**1** 0000000**1** 0000000**0**
4,2 (azul)	100	11111111 00000000 00000000	1111111**1** 0000000**0** 0000000**0**
5,2 (amarelo)	011	00000000 11111111 11111111	0000000**0** 1111111**1** 1111111**1**
6,2 (verde)	001	00000000 11111111 00000000	0000000**0** 1111111**0** 0000000**1**
7,2 (vermelho)	001	00000000 00000000 11111111	0000000**0** 0000000**0** 1111111**1**
8,2 (branco)	001	11111111 11111111 11111111	1111111**0** 1111111**0** 1111111**1**
9,2 (preto)	010	00000000 00000000 00000000	0000000**0** 0000000**1** 0000000**0**
10,2 (azul)	011	11111111 00000000 00000000	1111111**0** 0000000**1** 0000000**1**
11,2 (amarelo)	010	00000000 11111111 11111111	0000000**0** 1111111**1** 1111111**0**
12,2 (verde)	010	00000000 11111111 00000000	0000000**0** 1111111**1** 0000000**0**
13,2 (vermelho)	001	00000000 00000000 11111111	0000000**0** 0000000**0** 1111111**1**
14,2 (branco)	1	11111111 11111111 11111111	1111111**1** 11111111 11111111

6 CONCLUSÕES

Foi estudada a criptografia através do algoritmo simétrico AES e a esteganografia, sua aplicação em imagens digitais no formato *bitmap* e o método de inserção no último bit significativo (LSB), principal método para armazenar informação em imagens.

Em conjunto com a criptografia, a esteganografia torna-se bastante segura, sendo praticamente impossível violar qualquer mensagem oculta. Com a esteganografia é possível ocultar qualquer tipo de formato de arquivo, não apenas texto, pois o que importa é a informação binária que representa este arquivo. Pode-se, por exemplo, ocultar uma imagem dentro de outra.

O protótipo desenvolvido permite comunicar-se de maneira secreta e segura, sem que haja qualquer suspeita da existência da comunicação, o que não acontece utilizando apenas a criptografia. Com o protótipo também é possível, por exemplo, criar uma assinatura digital ou uma marca d'água autêntica.

Uma das limitações do protótipo é fazer uso apenas de imagens no formato *bitmap*. Com esse formato, as imagens costumam ser grandes no tamanho para armazenamento em disco e esse tamanho pode ser problema na hora de transmitir essa imagem ao receptor. Além disso, o método de esteganografia utilizado não permite que a imagem sofra compressão.

As maiores dificuldades encontradas foram no levantamento bibliográfico sobre esteganografia, por se tratar de um assunto pouco abordado.

6.1 EXTENSÕES

Como extensão deste trabalho, pode-se estudar outras técnicas de esteganografia, para outros formatos de imagens, que utilizam algoritmos de compressão, por exemplo. Outros dois métodos que podem ser estudados são: filtragem e mascaramento; e algoritmos de transformação.

Pode-se estudar também a utilização de esteganografia em outras mídias, como arquivos de áudio e vídeo. E também tornar possível esconder outros formatos de arquivos, não apenas texto.

Outra sugestão é implementar a compressão de Huffman, método estatístico utilizado para compactar dados. A compressão de Huffman pode ajudar a esteganografia, comprimindo a mensagem que será escondida na imagem, isso permite que um maior número de caracteres possa ser esteganografado em uma imagem pequena. Apesar de abordado na fundamentação teórica deste trabalho, não foi implementado, ficando como sugestão para trabalhos futuros.

Outra possibilidade é a utilização de mais algoritmos de criptografia na codificação e decodificação da mensagem, algoritmos assimétricos (de chave pública), por exemplo.

REFERÊNCIAS BIBLIOGRÁFICAS

CASACURTA, Alexandre et al. **Computação gráfica – Introdução**, Rio Grande do Sul, set. 1998. Disponível em: <http://www.inf.unisinos.br/~osorio/CG-Doc/cg.pdf>. Acesso em: 8 set. 2003.

DANTAS, George Felipe de Lima. **Esteganografia digital**, Brasília, ago. 2002. Disponível em: <http://www.peritocriminal.com.br/esteganografia.htm>. Acesso em: 8 set. 2003.

ERIGSON, Marcelo. **Esteganografia**, Rio Grande do Sul, jan. 2003. Disponível em: <http://www.inf.ufrgs.br/~mierigson/cpd>. Acesso em: 15 set. 2003.

HINZ, Marco Antônio Mielke. **Um estudo descritivo de novos algoritmos de criptografia**, Rio Grande do Sul, dez. 2000. Disponível em: <http://www.ufpel.tche.br/prg/sisbi/bibct/acervo/info/2000/Mono-MarcoAntonio.pdf>. Acesso em: 9 set. 2003.

KANISHIMA, Eliana; MORAIS, Everson; RIBEIRO, Fabiano; TAVARES, Gislaine; OURA, Lilian. **Segurança em redes**, Paraná, jun. 2000. Disponível em: <http://proenca.uel.br/curso-redes-especializacao/2000-uel/trab-02/equipe-08>. Acesso em: 9 set. 2003.

KONKOL, Laura Mireile; STRINGARI, Sergio. **Protótipo de software para criptografia de dados usando o algoritmo Idea**. 1997. 61 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

KUNZ, Leonardo. **Esteganografia em imagens usando codificação de Huffman**, Rio Grande do Sul, jan. 2003. Disponível em: <http://www.inf.ufrgs.br/~lkunz/cpd>. Acesso em: 15 set. 2003.

LIMA, Fabrício Luis. **Componentes CE3PO**, [?], jun. 2003. Disponível em: <http://www.ce3po.hpg.ig.com.br/delphi/ce3po/index.html>. Acesso em: 15 set. 2003.

LUCCHESI, Claudio Leonardo. **Introdução à criptografia computacional**. Campinas: Papirus/UNICAMP, 1986. xiii, 132 p.

MACÊDO, Rodrigo; TRINTA, Fernando. **Um estudo sobre criptografia e assinatura digital**, Pernamuco, set. 1998. Disponível em: <http://www.di.ufpe.br/~flash/ais98/cripto/criptografia.htm>. Acesso em: 24 jul. 2003.

OLIVEIRA, Wilson José de. **Hacker**: invasão e proteção. 2.ed. Florianópolis: Visual Books, 2000. 386 p.

PUTTINI, Ricardo S.; SOUSA, Rafael T. de. **Criptografia e segurança de redes de computadores**, Brasília, dez. 2000. Disponível em: <http://www.redes.unb.br/security/criptografia/rsa/rsa.html>. Acesso em: 02 dez. 2003.

ROCHA, Anderson Rezende. **Desenvolvimento de um software para segurança digital utilizando esteganografia**, Lavras, jul. 2003. Disponível em: <http://www.comp.ufla.br/~undersun/ic/estego/src/projeto_orientado_1.pdf>. Acesso em: 02 dez. 2003.

SANTOS, Leandro dos; EMER, Cassio; AVER, Rodrigo. **Apostila de criptografia**, [?], 1996. Disponível em: <http://www.inf.ufsc.br/~prass/apostilas/criptografia.zip>. Acesso em: 9 set. 2003.

SOARES, Luiz Fernando G.; LEMOS, Guido; COLCHER, Sérgio et al. **Redes de computadores**: das LANs, MANs e WANs às redes ATM. 2.ed. Rio de Janeiro: Campus, 1995. 705 p.

STALLINGS, William. **Network and internetwork security principles and practice**. Englewood Cliffs: Prentice Hall, 1995. xiii, 462 p.

STANG, David J.; MOON, Sylvia. **Segredos de seguranca em rede**. Rio de Janeiro: Berkeley, 1994. xxvi, 986 p.

TANENBAUM, Andrew S. **Redes de computadores**. Rio de Janeiro: Campus, 1994. 786 p.

WILNER, Alessandro. **Terrorismo x privacidade**, São Paulo, nov. 2001. Disponível em: <http://www.ime.usp.br/~is/ddt/mac339/projetos/2001/alessandro>. Acesso em: 24 jul. 2003.

ZANELLA, Daniel. **Protótipo de software para inserção e extração de mensagens em arquivos raster através de esteganografia**. 2002. 79 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

ANEXO A – Algoritmo de esteganografia para ocultar a informação na imagem.

```
procedure TImgEstegano.TxtToImg(aTxt: String);
var
  xPixelBin : String[24];
  x, y: Integer;
  xTxtBin: String;
  xBit: Integer;
begin
  xTxtBin := FBinUtil.TxtToBin(aTxt);
  xBit := 0;

  if (FLabelStatus <> nil) then
  begin
    FLabelStatus.Caption := 'Codificando texto na imagem...';
    FLabelStatus.Refresh;
  end;

  if (FProgressBar <> nil) then
  begin
    FProgressBar.Min := 0;
    FProgressBar.Max := Trunc(Int(Length(xTxtBin) / 3)) * 3;
    FProgressBar.Step := 3;
  end;

  for y := 0 to (FImagem.Height - 1) do
  begin
    for x := 0 to (FImagem.Width - 1) do
    begin
      xPixelBin := FBinUtil.IntToBin(FImagem.Bitmap.Canvas.Pixels[x,y], 24);

      if (xBit < Length(xTxtBin)) then
      begin
        Inc(xBit);
        xPixelBin[8] := xTxtBin[xBit];
      end;

      if (xBit < Length(xTxtBin)) then
      begin
        Inc(xBit);
        xPixelBin[16] := xTxtBin[xBit];
      end;

      if (xBit < Length(xTxtBin)) then
      begin
        Inc(xBit);
        xPixelBin[24] := xTxtBin[xBit];
      end;

      FImagem.Bitmap.Canvas.Pixels[x,y] := FBinUtil.BinToInt(xPixelBin);

      if (FProgressBar <> nil) then
        FProgressBar.StepIt;

      if (xBit >= Length(xTxtBin)) then
        Break;
    end;
    if (xBit >= Length(xTxtBin)) then
      Break;
  end;

  if (FProgressBar <> nil) then
    FProgressBar.Position := 0;

  if (FLabelStatus <> nil) then
  begin
    FLabelStatus.Caption := '';
    FLabelStatus.Refresh;
  end;
end;
```

ANEXO B – Algoritmo de esteganografia para extrair a informação da imagem.

```
function TImgEstegano.TxtFromImg(aTipo: TImgToTxt): String;
var
  xPixelBin : String;      // Valor binário de cada pixel da imagem
  x, y, z,                 // Índices para percorrer os pixels da imagem
  xPixelIniY,              // Em qual pixel (linha) deve iniciar a leitura
  xPixelIniX: Integer;     // Em qual pixel (coluna) deve iniciar a leitura
  xTxtBin   : String;      // Texto em binário
  xByte     : Integer;     // Em qual byte do pixel (1, 2 ou 3) deve iniciar a leitura
  xSair     : Boolean;
  xStrBinID : String;
  xIDSize   : Integer;
begin
  Result := '';
  xTxtBin := '';
  xByte := 1;
  xSair := false;
  xStrBinID := '';

  { Converte o identificador para o formato binário }
  if (aTipo = ittID) then
  begin
    xStrBinID := FBinUtil.TxtToBin(cIniTxtID);
    xPixelIniX := 0;
  end
  else
  begin
    xStrBinID := FBinUtil.TxtToBin(cFimTxtID);
    xPixelIniX := Trunc(Int((Length(cIniTxtID) * 8) / 3)); // Int(56 bits / 3 bytes por pixel)
  end;

  { Inicializa o tamanho do identificador = nº de bits }
  xIDSize := Length(xStrBinID);

  { Pixel inicial do texto, após o identificador }
  xPixelIniY := 0;

  { Em qual pixel vai iniciar a leitura da imagem }
  if (xPixelIniX > FImagem.Width) then
  begin
    xPixelIniY := Trunc(Int(xPixelIniX / FImagem.Width));
    xPixelIniX := xPixelIniX - (xPixelIniY * FImagem.Width);
  end
  else
  if (aTipo = ittTexto) then
    { Verifica em qual dos 3 bytes do pixel (1, 2 ou 3) vai iniciar a leitura do texto }
    xByte := ((Length(cIniTxtID) * 8) mod 3) + 1; // (56 bits mod 3 bytes por pixel) + 1

  { Mostrar Status do Processo para o usuário }
  if (FLabelStatus <> nil) then
  begin
    FLabelStatus.Caption := 'Extraindo texto da imagem...';
    FLabelStatus.Refresh;
  end;

  if (FProgressBar <> nil) then
  begin
    FProgressBar.Min := (xPixelIniY * xPixelIniX);
    FProgressBar.Max := (FImagem.Height * FImagem.Width);
    FProgressBar.Step := 1;
  end;

  { Inicia a leitura da imagem }
  for y := xPixelIniY to (FImagem.Height - 1) {bits} do
  begin
    for x := xPixelIniX to (FImagem.Width - 1) {bits} do
    begin
      { Lê o binário do pixel }
      xPixelBin := FBinUtil.IntToBin(FImagem.Bitmap.Canvas.Pixels[x,y], 24);

      { Processa o(s) byte(s) do pixel }
      for z := xByte to Trunc(Int(Length(xPixelBin) / 8)) do
```

```
      begin
        xTxtBin := xTxtBin + xPixelBin[(z * 8)];

        { Verifica identificação de FIM DE TEXTO }
        if (Length(xTxtBin) >= xIDSize) and
           ((aTipo = ittID) or
            (CompareText(Copy(xTxtBin,(Length(xTxtBin) - (xIDSize - 1)),xIDSize), xStrBinID) =
0)) then
        begin
          { Retira o identificador de FIM DE TEXTO }
          if (aTipo = ittTexto) then
            xTxtBin := Copy(xTxtBin, 0, Length(xTxtBin) - xIDSize);

          xSair := true;
          Break;
        end;
      end;
      xByte := 1;

      if (FProgressBar <> nil) then
        FProgressBar.StepIt;

      if (xSair) then
        Break;
    end;
    xPixelIniX := 0;

    if (xSair) then
      Break;
  end;

  if (FProgressBar <> nil) then
    FProgressBar.Position := 0;

  if (FLabelStatus <> nil) then
  begin
    FLabelStatus.Caption := '';
    FLabelStatus.Refresh;
  end;

  { Transforma o código binário em texto }
  if (xTxtBin <> '') then
    Result := FBinUtil.BinToTxt(xTxtBin);
end;
```