FUNDAÇÃO DE ENSINO “EURÍPIDES SOARES DA ROCHA”
CENTRO UNIVERSITÁRIO “EURÍPIDES DE MARÍLIA” – UNIVEM
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RAPHAEL RODRIGUES HONDA

ANÁLISE E IMPLEMENTAÇÃO DE ALGORITMOS PARA MANIPULAÇÃO DE ESTEGANOGRAFIA EM IMAGENS

MARÍLIA
2011

RAPHAEL RODRIGUES HONDA

ANÁLISE E IMPLEMENTAÇÃO DE ALGORITMOS PARA MANIPULAÇÃO DE ESTEGANOGRAFIA EM IMAGENS

Trabalho de Curso apresentado ao Curso de Ciência da Computação da Fundação de Ensino "Eurípides Soares da Rocha", mantenedora do Centro Universitário Eurípides de Marília – UNIVEM, como requisito para obtenção do grau de Bacharel em Ciência da Computação.

Orientador:
Prof. Ms. MAURICIO DUARTE

MARÍLIA
2011

CENTRO UNIVERSITÁRIO EURÍPIDES DE MARÍLIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO – AVALIAÇÃO FINAL

Raphael Rodrigues Honda

ANÁLISE E IMPLEMENTAÇÃO DE ALGORITMOS PARA MANIPULAÇÃO DE ESTEGANOGRAFIA EM IMAGENS

Banca examinadora da monografia apresentada ao Curso de Bacharelado em Ciência da Computação do UNIVEM/F.E.E.S.R., para obtenção do Título de Bacharel em Ciência da Computação.

Nota: 7.0 (sete)

Orientador: Mauricio Duarte

1°. Examinador: Ildeberto de Gênova Bugatti

2°. Examinador: Leonardo Castro Botega

Marília, 30 de novembro de 2011.

Pela memória de Irene Ernestina Werklin da Silva.

AGRADECIMENTOS

Agradeço a todos que me ajudaram direta ou indiretamente durante a faculdade e no desenvolvimento deste trabalho: meus amigos Benito, Zé, Márcio, Jonathan, Fernandão, Fernando Yokota, Laura e Helena. Agradeço também meu professor e orientador Mauricio Duarte pela paciência.

Agradeço principalmente minha família, em especial minha mãe, minha avó e minhas tias que sempre me ajudaram e me incentivaram nesses quatro anos de formação.

Também não posso esquecer da minha namorada Amanda, que foi muito paciente e compreensiva na reta final do meu trabalho, quando as coisas não iam tão bem. Amanda, obrigado pela companhia e por me dar força sempre!

Sou grato também a minha coordenadora do ensino médio, Ana Dolores, por acreditar em mim mesmo quando eu não merecia.

Renatão, Dog, Gordo e Alice: Oriente nunca mais!

"Let it be, let it be, let it be..."
(John Lennon and Paul McCartney)

HONDA, Raphael Rodrigues. **Análise e Implementação de Algoritmos para Manipulação de Esteganografia em Imagens.** 2011. 55 f. Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha de Marília, 2011.

RESUMO

Este trabalho, intitulado "Análise e Implementação de Algoritmos para Manipulação de Esteganografia em Imagens", visa estudar e analisar a historia da esteganografia e as diferentes técnicas esteganográficas existentes, bem como implementar algoritmos esteganográficos para imagens digitais.

Palavras-chave: esteganografia, criptografia, segurança, imagem.

HONDA, Raphael Rodrigues. **Análise e Implementação de Algoritmos para Manipulação de Esteganografia em Imagens.** 2011. 55 f. Centro Universitário Eurípides de Marília, Fundação de Ensino Eurípides Soares da Rocha de Marília, 2011.

ABSTRACT

This work, entitled "Analysis and implementations of Algorithms for Manipulating Steganography in Images" aims to study and analyze the history of steganography and the different steganography techniques, as well as implementing steganographics algorithms for digital images.

Keywords: steganography, cryptography, security, image.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ASCII: *American Standard Core for Information Interchange*

BMP: *Bitmap Image*

CAPTCHA: *Completely Automated Public Turing Test to Tell Computers And Humans Apart*

DES: *Data Encryption Standard*

DNS: *Domain Name System*

DNSSEC: *Domain Name System Security Extensions*

IDEA: *International Data Encryption Algorithm*

IDS: *Intrusion Detection Systems*

HTTP: *Hypertext Transfer Protocol*

IP*: Internet Protocol*

IPSec: *Internet Protocol Security*

JPG: *Joint Photographic Group*

JPEG: *Joint Photographic Expert Group*

LSB: *Least Significant Bit*

MD5: *Messa-Digest Algorithm 5*

NSA: *National Security Agency*

PGP: *Pretty Good Privacy*

PKI: *Public Key Infrastructure*

RGB*: Red Green Blue*

SAAS: *Software As a Service*

SHA: *Secure Hash Algorithm*

SSH: *Secure Shell*

SSL: *Secure Sockets Layer*

TI: Tecnologia da Informação

LISTA DE TABELAS

LISTA DE GRÁFICOS

SUMÁRIO

INTRODUÇÃO

O objetivo principal deste trabalho é fazer uma análise de como a esteganografia, em conjunto com outros elementos da criptologia, é importante para a segurança das informações que trafegam na rede entre clientes e servidores, através do estudo da história, conceitos e algoritmos da esteganografia. Algoritmos esteganograficos para imagens digitais serão implementados e analisados para que se possa confirmar, ou não, sua viabilidade como ferramenta para segurança da informação. Também é objetivo deste trabalho ser uma fonte de consulta sobre histórico da esteganografia e áreas relacionadas.

CAPÍTULO 1 – SEGURANÇA DA INFORMAÇÃO

1.1. Introdução

Quando se fala de segurança de sistemas, o que exige maior atenção são a segurança das informações armazenadas nos bancos de dados desses sistemas e os dados que trafegam entre clientes e servidores.

Nos primórdios da informática, as aplicações eram desenvolvidas e os dados gerados por essas aplicações eram armazenados localmente, em arquivos ou banco de dados administrados também localmente. Mas com a popularização da internet e o aumento significativo de velocidade das redes de computadores, surgiram alternativas para o modelo local de processamento e armazenamento de dados. Termos como *Cloud Computing* (Computação em Nuvem, quando o processamento e os dados de uma aplicação ficam em servidores administrados por terceiros) e *SaaS – Software as a Service* (Software como Serviço, quando um software é vendido no formato de um serviço na web, não sendo necessária a instalação do software na máquina cliente) surgiram, aumentando a quantidade de informações importantes circulando na internet.

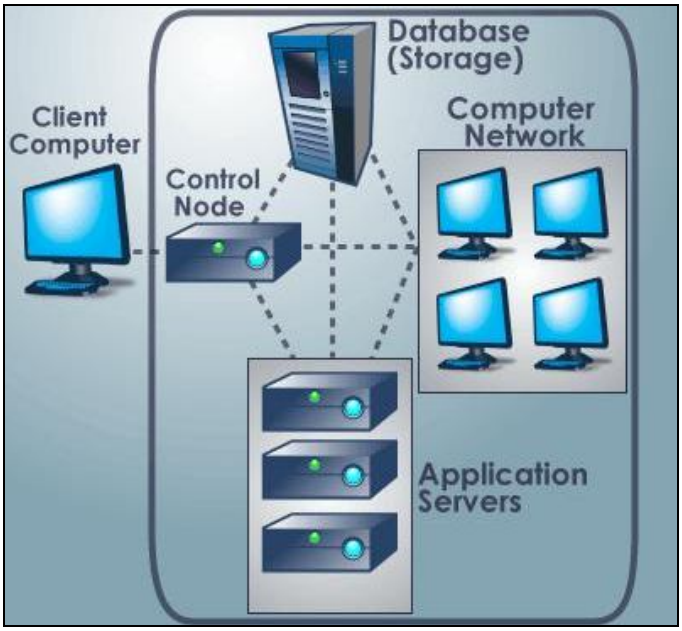

Figura 1 - “A Typical Cloud Computing System” (HowStuffWorks, 2011).

Os dados importantes das empresas costumavam ficar dentro do espaço físico da mesma, logo, as preocupações com segurança eram menores. Porém, com a modernização de alguns meios, muitas empresas agora mantêm suas aplicações e banco de dados em servidores

administrados por terceiros, como mostra a figura 1. Isto acontece devido a uma série de vantagens que a computação em nuvem ou o modelo de software como serviço proporciona, reduzindo gastos com infra-estrutura e funcionários especializados.

Vantagens da computação em nuvem:

- Reduz problemas de compatibilidade com sistema operacional e hardware (Parte física do computador), já que o software é geralmente acessado por um *browser* (navegador) através do protocolo *HTTP* (*Hypertext Transfer Protocol* - Protocolo de Transferência de Hipertexto).
- Infraestrutura mais simples, gerando menos gastos e manutenção.
- Trabalho corporativo e compartilhamento de arquivos mais fácil.

Em contrapartida, as informações das empresas que circulam na rede ficam mais vulneráveis a ataques, por isso é muito importante que se tenha uma política de segurança a fim de evitar o vazamento desses dados.

Segundo uma matéria do site especializado no mercado de TI (Tecnologia da Informação), COMPUTERWORLD (ComputerWorld, 2010), 5% de todo o investimento em TI vai para a área de segurança da informação.

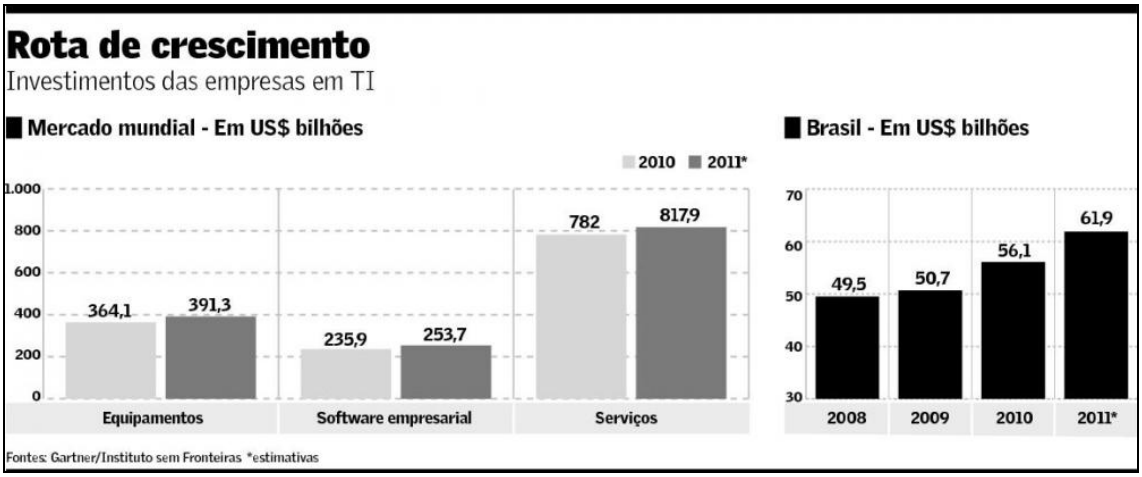

Figura 2 - Gartner/Instituto sem Fronteiras –“ Rota de crescimento – Investimento das empresas em TI” (Sassi, 2011).

Levando em consideração a figura 2, que mostra o gráfico do crescimento do investimento em TI das empresas no Brasil e no mundo, observa-se que foi gasto, em média, nos anos de 2010 e 2011, uma quantia de aproximadamente 71 bilhões de dólares em segurança da informação no mercado mundial de TI.

Essa quantidade de dinheiro investido em segurança da informação mostra que a cada dia as empresas estão mais preocupadas em proteger suas informações sigilosas de ataques.

1.2. Política de Segurança e Ataques

Segundo Lyra (2008), um ataque é “um tipo de incidente de segurança caracterizado pela existência de um agente que busca obter algum tipo de retorno, atingindo algum ativo de valor”.

Existem vários tipos de ataques, cada um deles utiliza-se de uma técnica e têm como objetivo simplesmente afetar a disponibilidade de informações ou capturar as mesmas, a seguir serão listados os principais:

1.2.1. *Denial of Service* DOS/DDOS (Negação de Serviço)

O objetivo deste tipo de ataque é tornar um serviço indisponível para os usuários. Ele foca em derrubar serviços enviando uma quantidade excessiva de dados ao mesmo tempo para o servidor, fazendo com que este não suporte a demanda, causando assim, a interrupção do mesmo. Também é conhecido como *flood.*

Prevenção/Solução: existem algumas estratégias para evitar este tipo de ataque. Entre elas, a de sempre manter os sistemas atualizados (pois se sabe que este tipo de ataque usufrui de vulnerabilidades conhecidas de sistemas operacionais e de segurança), utilizar filtros anti-*spoofing* (evita ataques utilizando a técnica de *spoofing*, onde o IP verdadeiro do atacante é alterado, dificultando assim a identificação da origem do ataque) e a limitação de banda por tipo de tráfego, onde a banda para tráfego de pacotes específicos é limitada, evitando que haja o *flood*.

1.2.2. Spoofing

O *Spoofing* é uma técnica que pode ser utilizada para dois principais fins, um deles é mascarar o IP do atacante com um IP geralmente de redes locais, para dificultar sua identificação e assim poder enviar pacotes a fim de derrubar o serviço atacado, e o outro é descobrir um host confiável, mascarar o próprio IP com este host confiável e então conseguir acesso a uma rede, enviando pacotes como se fosse outro remetente conhecido na rede.

Prevenção/Solução: não implementar autenticação por endereço de origem e configurar a rede para não aceitar pacotes de outras redes que utilizem um endereço local.

1.2.3. Man in the Middle

Neste tipo de ataque, o atacante intercepta o tráfego se posicionando entre as partes de uma comunicação, fazendo com que as partes comunicantes não percebam sua presença, e assim captura informações trocadas entre os envolvidos. Sistemas de comunicação sem fio estão mais propícios a esse tipo de ataque.

Prevenção/Solução: manter o sistema atualizado, implantar estrutura de chave pública e realizar testes de latência que detectam se cálculos criptográficos estiverem sendo realizados por terceiros.

1.2.4. Sniffing

O *Sniffing* é um procedimento realizado por ferramentas conhecidas como *Packet Sniffer*, Analisador de Rede, Analisador de Protocolo, *Ethernet Sniffer* e *Wireless Sniffer*. Trata-se de uma ferramenta que é um software ou um hardware capaz de interceptar e armazenar o tráfego de uma rede. O *Sniffer* captura os pacotes que trafegam na rede e os decodifica e analisa seu conteúdo conforme o protocolo definido em uma especificação.

O *Sniffing* é geralmente utilizado por pessoas que queiram obter algum tipo de informação sigilosa ou causar algum tipo de dano, seja material ou moral há alguém.

Prevenção/Solução: "Encriptação total das comunicações, incluindo a autenticação de credenciais. Isto evita que os pacotes de *sniffing* possam ser usados por um invasor. SSL e IPSec (*Internet Protocol Security*) são exemplos de soluções de encriptação" (Microsoft Patterns & Practices, 2004).

1.2.5. DNS Poisoning

A técnica "DNS *Poisoning*" consiste em "envenenar" os arquivos DNS de um servidor, esta brecha foi revelada em julho de 2008 e consiste em direcionar os registros que apontam para *hosts* seguros para direções erradas. Por exemplo, o cliente digita a URL do site do banco, o atacante intervém entre o cachê do cliente e o servidor DNS e introduz um endereço falso dentro do servidor, redirecionando o cliente para o site especificado pelo atacante.

Prevenção/Solução: Alguns sites utilizam a tecnologia DNSSEC (*Domain Name System Security Extensions***)** para tentar evitar este tipo de ataque. O DNSSEC é um padrão

que estende do DNS. Trata-se de um sistema de resolução de nomes mais seguro que utiliza chave assimétrica, ou seja, um sistema de chave pública e privada.

1.2.6. Brute Force

O método *Brute Force* (Força Bruta) serve para quebrar senhas. Este método consiste em tentar todas as possibilidades até encontrar a senha correta para acessar algum tipo de serviço ou informação. Exemplo: Se uma senha só pode ter números de 0 a 9 e seu tamanho é de no máximo 4 dígitos, então o método irá tentar 9999 combinações (0000, 0001, 0002, 0003 ... 9999) ou parar quando encontrar a combinação correta.

Prevenção/Solução: Para prevenir este tipo de ataque é recomendado que as senhas sejam constituídas de caracteres de todos os tipos, como números, letras e caracteres especiais. Isto faz com que o tempo para realizar as tentativas do *Brute Force* seja muito mais alto.

Outra forma de se prevenir deste tipo de ataque é o uso do *Captcha* (*Completely Automated Public Turing test to tell Computers and Humans Apart*), que são sistemas de conferência humana, onde é necessário enxergar ou ouvir um determinado código aleatório e digitá-lo junto da senha para ter acesso. A figura 3 é um exemplo de *Captcha* utilizado no site Google.com.

Figura 3 - *Captcha* do site www.google.com.br (Google, 2011).

1.2.7. Phishing

Phishing é uma técnica fraudulenta utilizada por pessoas mal intencionadas a fim de obter dados sigilosos como número de cartões de credito, dados de agência e conta bancária, entre outros. A técnica de *phishing* é aplicada de várias formas, tanto em sites que imitam a

aparência do site original, até e-mails falsos que levam os destinatários a fornecer dados pessoas. Nesta técnica são utilizados meios da engenharia social para influenciar a vítima a crer que a mensagem, site ou e-mail são legítimos.

Prevenção/Solução: existem diversas formas de se prevenir este tipo de prática, na sua maioria essas maneiras de prevenção são na verdade iniciativas que o usuário deve ter ao utilizar a internet. Hábitos como nunca revelar dados pessoais ou bancários em e-mails (as instituições bancárias não costumam pedir dados via *e-mail*), verificar no rodapé do navegador se o endereço do site corrente corresponde com o site que se desejava estar e utilizar navegadores que possuam filtros *anti-phishing*.

1.2.8. Software Exploitation

Software Exploitation (Exploração de Software) é uma técnica que busca por brechas em sistemas operacionais e aplicativos e as usam para obter algum tipo de benefício ou até mesmo somente para causar algum tipo de dano a um determinado alvo. Sistemas operacionais e muitos aplicativos possuem vulnerabilidades que são freqüentemente exploradas por pessoas mal intencionadas.

Prevenção/Solução: as empresas estão sempre corrigindo e lançando atualizações para seus produtos. Para evitar ser alvo de um software mal intencionado, devem-se manter os *softwares* sempre atualizados, mas isso não assegura que o sistema estará totalmente seguro.

1.2.9. Buffer Overflow

Um *Buffer Overflow* (estouro de pilha) acontece quando um programa é mal escrito. Os *Buffers* são áreas criadas na memória para os programas armazenarem dados que estão sendo processados. A técnica de *buffer overflow* consiste em preencher um buffer de determinado tamanho com uma quantidade de caracteres acima de sua capacidade, dessa forma, os caracteres excedentes serão armazenados em posições de memórias próximas, ocasionando o encerramento do programa ou até a execução deste excedente. **Prevenção/Solução:** utilizar bibliotecas de programação que não possuam problemas com *buffer overflow* e tratar os limites nos parâmetros recebidos pelo *buffer* para que não ultrapasse a capacidade.

1.2.10. Port Scanning

O *Port Scanning* (Escaneamento de Portas) é executado por um *Scanner* de vulnerabilidade em um sistema para detectar portas abertas e possíveis problemas de segurança. Após detectar as vulnerabilidades, outras técnicas de ataque são utilizadas para explorar essas brechas de segurança.

Prevenção/Solução: para evitar este tipo de ataque, são utilizados *firewalls* com IDS (*Intrusion Detection Systems*), que são capazes de bloquear o acesso a determinadas portas e detectar esta técnica.

1.2.11. Engenharia Social

Engenharia Social é uma técnica utilizada para conseguir informações de vítimas que são manipuladas até fornecer os dados que o atacante deseja. O atacante utiliza técnicas que exploram as vulnerabilidades das próprias vítimas usando de meios que mexam com a vaidade, autoconfiança, relacionamentos, entre outros.

Muitas vezes o atacante tenta persuadir a vítima se passando por outra pessoa ou levando a mesma a revelar segredos sem saber da real intenção do solicitante.

Prevenção/Solução: uma das formas de se evitar este tipo de ataque é com o treinamento e conscientização de usuários sobre como agem os engenheiros sociais.

Devido a uma grande quantidade de ataques e pessoas empenhadas em descobrir novas vulnerabilidades, a segurança da informação é um assunto muito discutido. Ela possui algumas vertentes que a seguir serão descritas de forma detalhada a fim de se ter um melhor entendimento de quais são e quais suas peculiaridades.

Segundo Lyra (2008, p.03), “três vertentes se destacam, são elas: confidencialidade, integridade e disponibilidade”.

Confidencialidade: é a capacidade que um sistema pode ter de tornar determinada informação confidencial, ou seja, apenas usuários autorizados poderão acessá-la, enquanto outros, fora da lista de permissão, não terão acesso ou então nem saberão de sua existência. Quando há perda de confidencialidade significa que houve perda de segredo. Uma informação confidencial deverá ser guardada com segurança para que somente pessoas autorizadas tenham acesso. Em sistemas operacionais multiusuários, cada usuário possui permissões para acessar determinados diretórios e arquivos, as permissões são de leitura, escrita e execução,

que podem ser atribuídas a usuários ou grupos de usuários.

Integridade: para uma informação ser íntegra, precisa estar correta, de acordo com a verdade e não pode estar corrompida. Não pode ter sido modificada por acidente ou propositalmente por algum usuário.

Disponibilidade: uma informação deve sempre estar disponível. Mesmo em casos de acidentes, as informações devem ser rapidamente recuperadas para ficar novamente disponíveis para os usuários, que poderão utilizá-las quando necessário.

Os três itens descritos acima são os principais conceitos da segurança da informação, mas existem dois itens em que a aplicação da esteganografia e/ou criptografia faz-se mais útil, na autenticidade e no não-repúdio.

Autenticidade: quando uma mensagem é enviada para alguém, ou para um sistema através da internet, algo muito importante de se verificar é a autenticidade da mesma. É necessário se ter certeza de que a mensagem realmente veio do remetente que a assinou, garantindo assim a sua autenticidade. Uma forma de se garantir a autenticidade de uma informação é utilizando o PKI (*Public Key Infrastructure*), esse sistema funciona da seguinte maneira; cada pessoa possui uma chave pública e uma chave privada. Quando uma pessoa deseja enviar uma mensagem, ela deve então criptografá-la utilizando sua chave privada e enviá-la. A pessoa que receber a mensagem deverá descriptografar a mensagem utilizando a chave pública do suposto remetente, caso consiga, significa que a mensagem realmente foi enviada pelo assinante.

Não-Repúdio: o não-repúdio é incapacidade de se negar algo que tenha feito. Um sistema dotado desta capacidade pode assegurar de que uma mensagem veio mesmo de seu remetente. Quando algo é assinado digitalmente não é possível negar sua autoria. Se o algoritmo esteganográfico utilizar este tipo de recurso, ele pode garantir a origem da mensagem.

1.3. Considerações Finais do Capítulo

Este capítulo descreveu a importância da segurança da informação nas companhias, as formas de ataque utilizadas para se obter ou tornar indisponível as informações e as características que tornam um sistema ou meio de armazenamento e/ou transferência de informações seguro. Nos capítulos seguintes serão descritas as técnicas utilizadas para tornar a informação ilegível e oculta, mostrando suas vantagem e desvantagens.

CAPÍTULO 2 – CRIPTOLOGIA

A criptologia é a área de conhecimento à qual a criptografia e esteganografia pertencem. Neste capítulo serão descritos estes dois ramos da criptologia, que são pertinentes ao contexto deste trabalho.

A necessidade do estudo da criptografia se dá devido à mescla que ocorre ao se desenvolver uma ferramenta de esteganografia, onde técnicas criptográficas e esteganográficas são utilizadas em conjunto para criptografar e ocultar uma mensagem.

2.1. Criptografia

Segundo Kahn (1967), “A criptografia é uma ciência que, ao contrário da esteganografia, não se dedica a esconder a presença da mensagem secreta, mas tornar as mensagens ininteligíveis para pessoas externas à comunicação através de diversas transformações do texto original”.

A criptografia é o estudo de uma série de técnicas que possibilitam a transformação da forma original de uma mensagem, para outra forma ilegível para uma pessoa que esteja fora do contexto de comunicação entre emissor e receptor, ou seja, o objetivo principal da criptografia é mascarar o real significado da mensagem.

A criptografia é classificada como uma ramificação da criptologia, e que, por sua vez, possui suas próprias ramificações, como pode ser observado na figura 4.

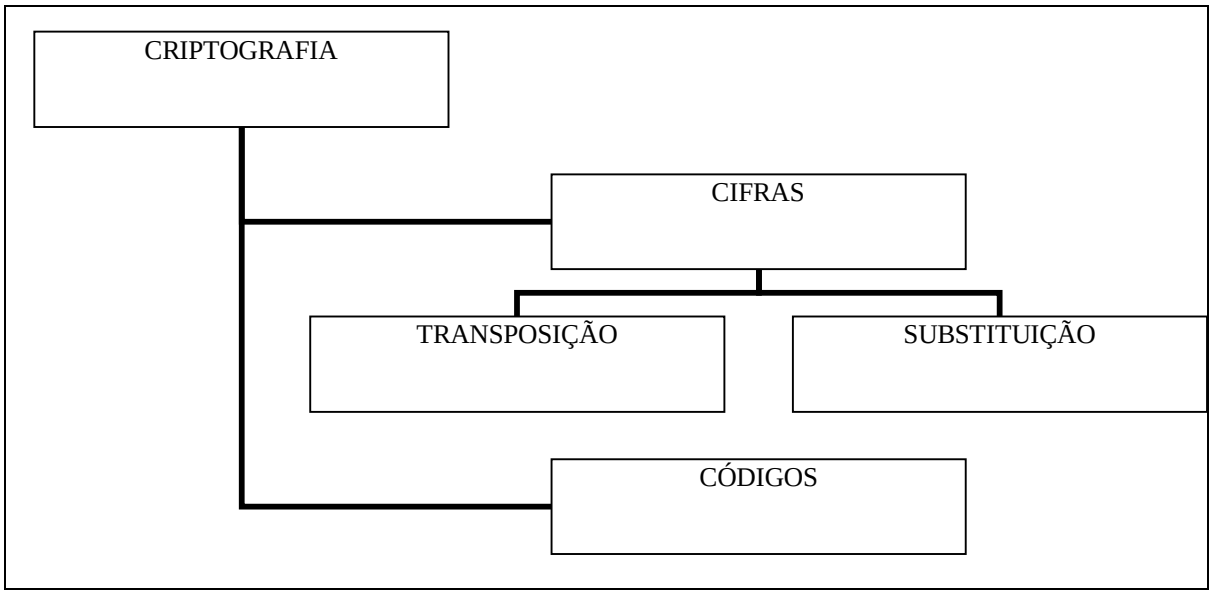

Figura 4 - Criptografia e suas ramificações (Própria).

Existem inúmeros procedimentos criptográficos que utilizam códigos, cifras de

transposição e cifras de substituição, os mais conhecidos serão descritos adiante, porém, um algoritmo criptográfico utilizado em meios digitais funciona basicamente da seguinte maneira:

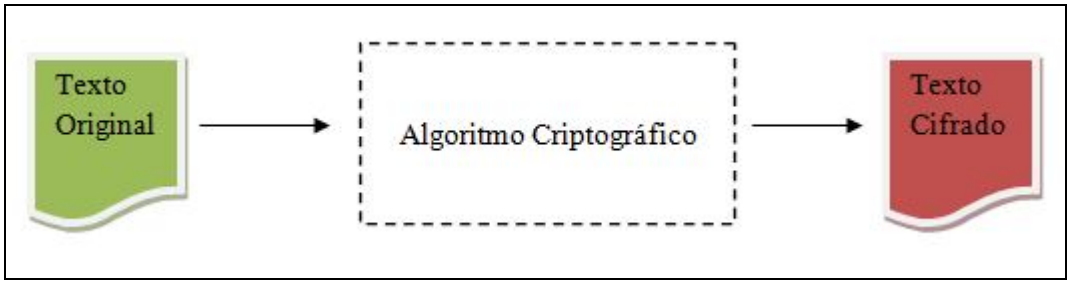

Figura 5 - Processo Criptográfico (Própria).

Como mostra a figura 5, um processo criptográfico nada mais é que a aplicação de um algoritmo criptográfico em um texto. Um processo criptográfico deve, então, possuir os seguintes elementos: algo a ser criptografado (geralmente um texto), uma chave compartilhada e um algoritmo criptográfico.

A chave criptográfica é a chave utilizada pelo algoritmo criptográfico para cifrar o texto e, posteriormente, decifrá-lo. Portanto, neste caso, a chave é compartilhada.

2.1.1. Códigos

A criptografia utilizando códigos manipula o significado substituindo palavras ou frases, enquanto a cifra substitui as letras, conjunto de letras ou, no caso do uso em computadores, os bits.

2.1.2. Cifras

“As cifras são técnicas nas quais a informação é cifrada por meio da transposição e/ou substituição das letras da mensagem original. Assim, as pessoas autorizadas podem ter acesso às informações originais conhecendo o processo de cifragem” (Dacencio *et. al.*, 2005).

A criptografia com cifras é divida em duas modalidades: transposição e substituição, como mostram os itens 2.1.2.1 e 2.1.2.2.

Em ambas as modalidades de criptografia com cifras, a intenção é alterar a forma como a informação é apresentada. Basicamente, uma cifra rearranja ou substitui as letras de uma mensagem.

2.1.2.1. Cifras de Transposição

Cifras de transposição rearranjam a informação. Em uma mensagem, por exemplo, as letras poderiam ser mudadas de posição para alterar a representação da mensagem. Em uma cifra de transposição onde a posição das letras é rearranjada, é necessário que se faça o processo inverso para retornar ao estado original.

Existem diversos tipos de cifra de transposição, que serão descritos no próximo item.

2.1.2.1.1. Cifra de Colunas

Um método clássico de cifra de transposição é a cifra das colunas. A cifra das colunas é um método em que os caracteres de um texto são inseridos até que se atinja o número máximo de linhas, passando então, para a coluna seguinte. A figura 6 demonstra o funcionamento da cifra das colunas.

E	U	M	E	A	L
S	M	P	C	D	U
T	E	L	I	E	N
E	X	O	F	C	A
E	E	D	R	O	S

Figura 6 - Exemplo de Cifra das colunas (Própria).

Na figura 6, foi usada a mensagem “ESTE E UM EXEMPLO DE CIFRA DE COLUNAS” para ilustrar esta técnica. Observe que neste caso, o número de linhas é 5, ou seja, as letras da mensagem são inseridas até completar 5 linhas e então inicia-se outra coluna. A frase cifrada ficaria da seguinte forma “EUMEAL SMPCDU TELIEN EXOFCA EEDROS”

2.1.2.1.2. Cifra de Colunas com Permutação

A cifra de colunas comum não é muito complexa, podendo ser facilmente quebrada pela criptoanálise. Já a cifra de colunas com permutação possui um nível maior de complexidade. Segundo Stallings (2008), a cifra de colunas com permutação consiste em

“escrever a mensagem em um retângulo, linha por linha, e ler a mensagem coluna por coluna, mas permutar a ordem das colunas. A ordem das colunas, então, torna-se a chave para o algoritmo.” Observa um exemplo dessa técnica na tabela 1.

Chave	4	3	1	2	6	5
Texto Original	O	R	A	T	O	R
	O	E	U	A	R	O
	U	P	A	D	O	R
	E	I	D	E	R	O
	M	A				
Texto cifrado	Auad tade repia oouem roro oror					

Tabela 1 - Cifra de Colunas com Permutação (Própria).

2.1.2.2. Cifras de Substituição

“O uso mais antigo que conhecemos de uma cifra de substituição, é o mais simples, foi feito por Júlio César. A cifra de César consiste em substituir cada letra do alfabeto pela letra que fica três posições adiante no alfabeto” (Stallings, 2008).

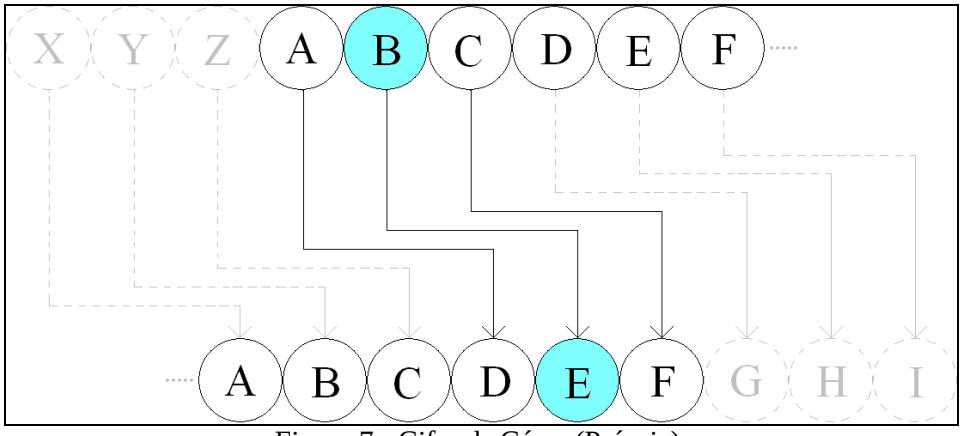

Figura 7 - Cifra de César (Própria).

Como mostra a figura 7, o funcionamento da cifra de césar é simples, para a palavra “COMPUTACAO”, por exemplo, após a aplicação da técnica teríamos “FRPSXWDFDR”, a tabela 2 contém o alfabeto cifrado

Normal	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cifrado	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Tabela 2 - Alfabeto Cifrado (cifra de césar) (Própria).

2.1.3. Técnicas de Criptografia Computacional

Como foi visto no capítulo 1, existem muitos riscos na transmissão de arquivos e mensagens pela internet, isso torna a criptografia muito importante para manter a integridade, disponibilidade, confidencialidade e autenticidade da informação.

A segurança eletrônica nunca foi tão amplamente discutida: casos de violação de contas bancárias, acesso a informações sigilosas, invasão e destruição de sistemas são cada vez mais comuns. Informações são transmitidas com mais eficiência e velocidade, mas como se sabe, nem sempre de forma segura (Dacencio *et. al.*, 2005).

Como foi visto, apesar de seguras, as técnicas não são infalíveis, podendo ser quebradas através da criptoanálise. Isso é um motivo para se usar criptografia juntamente com a esteganografia, unindo duas técnicas que podem, juntas, tornar uma transição ainda mais segura.

2.1.3.1. Principais Técnicas da Criptografia Moderna

A seguir, algumas técnicas criptográficas modernas:

2.1.3.1.1. Funções de *Hash* Criptográfico

Um *Hash* criptográfico(ou tabela *hash*) é uma palavra (sequência de *bits*) gerada por um algoritmo. O *hash* transforma uma informação em outra informação não condizente com a original. Essa sequência de *bits* identifica unicamente uma informação. Em sistemas em que são necessários *login* e senha, por exemplo, quando o usuário cadastra uma senha, ela passa por uma função de *hash* para ser gravada no banco de dados, dessa forma, o administrador do banco de dados não tem acesso a senha dos usuários, e a senha, através da função de *hash*, pode ser verificada a qualquer momento. Nas funções *hash* utlizadas em criptografia não possível se obter a informação original através da sequencia de *bits*.

2.1.3.1.1.1. MD5

O MD5 (*Messa-Digest Algorithm* 5) é um algoritmo *hash* de 128 bits unidirecional. Ele é utilizado em softwares de protocolo ponto-a-ponto na verificação de integridade dos arquivos e usuários.

2.1.3.1.1.2. SHA

SHA, ou *Secure Hash Algorithm* é uma família de algoritmos de *hash* criptográficos. O SHA-1 (Considerado o sucessor do MD5) é a função mais utilizada de família SHA, utilizada em muitas aplicações e protocolos de segurança, como TLS, SSl, PGP, SSH, S/MIME e IPSec. O SHA foi publicado em 1993 pela *National Security Agency* (NSA).

2.1.3.1.1.3. Sistemas *Free/Open Source*

2.1.3.1.1.3.1. PGP

PGP, ou *Pretty Good Privacy* é um algoritmo criptográfico de chave pública de alta segurança. O PGP foi escrito por Philip Zimmermann e nos últimos anos conquistou milhares de adeptos ao redor do mundo, tornando-se praticamente um padrão para criptografia de *e-mail*.

2.1.3.1.1.3.2. GPG

O GNU *Privacy Guard* (GnuPG ou GPG) é uma alternativa de *software* livre para a suíte de criptografia PGP. O GPG trabalha com criptografia assimétrica.

2.1.3.1.1.3.3. SSH

SSH ou *Secure Shell* é um *software* computacional e protocolo de rede capaz de realizar conexões com outros computadores em uma rede de forma remota. Ele é semelhante ao antigo TELNET, mas o SSH possui conexão criptografada.

2.1.3.1.2. Algoritmos Assimétricos ou de Chave Pública

A criptografia assimétrica, também chamada de criptografia de chave pública, utiliza-se de duas chaves matematicamente relacionadas, uma chave pública e outra chave privada. Se for utilizada a chave pública para cifrar, deve-se utilizar a chave privada para decifrar, e vice-versa (Misaghi, 2001).

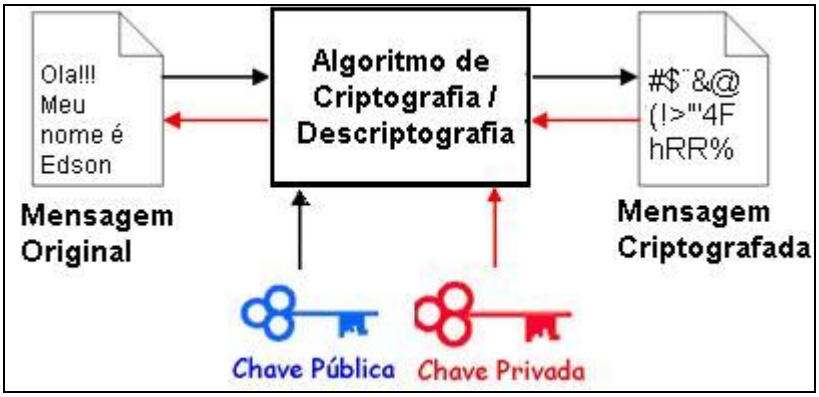

Figura 8 – Processo de Criptografia Assimétrica (*PUBLIC KEY*) (DevMedia , 2011).

Os algoritmos assimétricos, ou de chave pública têm seu funcionamento demonstrado na figura 8, a seguir alguns algoritmos que utilizam essa técnica.

2.1.3.1.3. Criptografia de Curvas Elípticas

Trata-se de uma variante da criptografia assimétrica que se baseia na matemática das curvas elípticas. Ela se apóia em uma proposta de seus criadores de que é capaz de ser mais rápida e gerar chaves mais curtas que de outros métodos semelhantes.

Diffie-Hellman: em 1976 surgiu o algoritmo *Diffie-Hellman*, que tornou possível a troca de informações criptografadas entre duas partes sem o compartilhamento de chaves secretas.

RSA: desenvolvido no instituto MIT por três professores, este algoritmo criptográfico é considerado a mais bem sucedida implementação de um algoritmo de chave assimétrica.

2.1.3.1.4. Sistema de Chaves Simétricas

A criptografia simétrica, também conhecida por criptografia convencional ou criptografia de chave privada, é um método que faz uso de uma chave secreta ou chave única para realizar os processos de encriptação e decriptação (Petri, 2004).

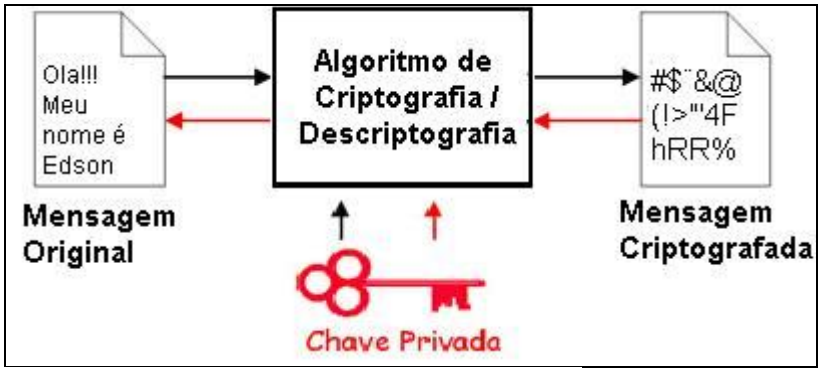

Figura 9 – processo de criptografia convencional (Fonte: http://www.devmedia.com.br/)

A seguir alguns algoritmos que utilizam esta técnica, que é demonstrada na tabela 3.

Algoritmo	Tamanho da Chave	Descrição
DES (Data Encryption Standard)	64 Bits	É um codificador composto, que cifra blocos de 64 bits (8 caracteres) em blocos de 64 bits, para isso se utiliza de uma chave composta por 56 bits, com 8 bits de paridade totalizando 64 bits. Utiliza o Algoritmo de Feistel.
DES Triplo	112 Bits	Alternativa do DES original, com variação de três diferentes chaves. O DES é aplicado três vezes, com a mesma chave ou com chaves diferentes.
IDEA	128 Bits	A filosofia que norteou este projeto foi "misturar operações de grupos algébricos diferentes" para misturar os caracteres iniciais de forma a ficarem incompreensíveis.
RC5	Tamanho Variável	A mensagem original é fornecida ao algoritmo sob a forma de dois blocos de **w** bits, correspondendo ao alinhamento mais conveniente para o sistema em causa, os valores típicos para **w** são: 16, 32 e 64. A mensagem cifrada possui forma idêntica.

Tabela 3 - Sistemas Simétricos (Jasper, 2009).

2.1.4. Criptoanálise

A criptoanálise é a ciência de “quebrar” os métodos criptográficos, para que se possa decifrar e ler as mensagens anteriormente criptografadas (Tkotz, 2005b). A criptoanálise é o estudo de métodos e técnicas utilizadas para atingir procedimentos criptográficos.

Normalmente, o objetivo de atacar um sistema de criptografia é recuperar a chave em uso, em vez de simplesmente recuperar o texto claro de um único texto cifrado. Existem duas técnicas para um ataque a um esquema de criptografia convencional; Criptoanálise e Força Bruta (Stallings, 2008).

Segundo Stallings (2008), os ataques criptoanalíticos contam com a natureza do algoritmo e talvez mais algum conhecimento das características gerais do texto claro, ou ainda

alguns pares de amostra de texto claro e texto cifrado. Esse tipo de ataque explora as características do algoritmo para tentar deduzir um texto claro específico ou deduzir a chave utilizada. Já no ataque por força bruta, ainda segundo Stallings, o atacante experimenta cada chave possível em um trecho do texto cifrado, até obter uma tradução inteligível para texto claro. Na média, metade de todas as chaves possíveis precisam ser experimentadas para se obter sucesso.

A tabela 4 mostra os tipos de ataques a mensagens criptografadas e o que é conhecido ao criptoanalista, em cada caso.

Tipos de ataque	**Conhecido ao criptoanalista**
Apenas texto cifrado	• Algoritmo de criptografia • Texto Cifrado
Texto claro conhecido	• Algoritmo de criptografia • Texto cifrado • Um ou mais pares de texto claro/texto cifrado formados com a chave secreta
Texto claro escolhido	• Algoritmo de criptografia • Texto cifrado • Mensagem de texto claro escolhida pelo criptoanalista, juntamente com seu texto cifrado correspondente, gerado com a chave secreta
Texto cifrado escolhido	• Algoritmo de criptografia • Texto cifrado • Texto cifrado pretendido, escolhido pelo criptoanalista, juntamente com seu texto claro decriptografado correspondente, gerado com a chave secreta
Texto escolhido	• Algoritmo de criptografia • Texto cifrado • Mensagem de texto claro escolhida pelo criptoanalista, juntamente com seu texto cifrado correspondente, gerado com a chave secreta • Texto cifrado pretendido, escolhido pelo criptoanalista, juntamente com seu texto claro decriptografado, gerado com a chave secreta

Tabela 4 - Tipos de ataques a mensagens criptografadas (Stallings, 2008).

Se qualquer tipo de ataque tiver sucesso na dedução da chave, o efeito será catastrófico: todas as mensagens futuras e passadas, codificadas com essa chave, estarão comprometidas (Stallings, 2008).

2.2. Esteganografia

Esteganografia é a arte de esconder mensagens e informações, tendo como objetivo a

comunicação em segredo. A esteganografia grava uma mensagem confidencial dentro de outra, que funciona como uma portadora, só que mais extensa. O objetivo é alterar a mensagem portadora de tal forma que o resultado seja imperceptível (Petri, 2004).

A esteganografia e a criptografia, que foi vista no tópico anterior, diferem uma da outra por uma série de aspectos, como pode ser visto na tabela 4.

ESTEGANOGRAFIA	CRIPTOGRAFIA
Oculta a mensagem em outro objeto de aparência inocente, como imagens, vídeos, sons ou arquivos.	A mensagem é visível, mas cifrada toma-se uma mistura de caracteres sem qualquer significado.
Uma coleção de imagens, vídeos ou sons que não geram suspeitas.	Pelo fato da existência da mensagem estar em evidência. A coleção de caracteres aleatórios gera muita suspeita e curiosidade.
Requer cuidado quando reutilizar arquivos de som ou imagem.	Requer cuidado ao reutilizar chaves.
Não existem leis regulamentando ou proibindo o uso de esteganografia	Existem leis que, inclusive, proíbem seu uso.

Tabela 5 - Diferenças entre esteganografia e criptografia (Kipper, 2004).

Ou seja, segundo a tabela descrita por Kipper (2004), enquanto a criptografia tem como objetivo tornar a mensagem ilegível para um agente fora da comunicação, a esteganografia se preocupa em ocultar a existência da mesma através de diversas técnicas clássicas e/ou computacionais.

2.2.1. Histórico

Segundo Jasper (2009), o primeiro registro de uso de esteganografia da história se dá a mais de 3000 anos do início da idade antiga.

Através dos tempos, foram registradas algumas técnicas de esteganografia utilizadas pelas civilizações, a seguir, na tabela 5, serão descritas algumas técnicas mais conhecidas.

Técnica	**Descrição**
Tabuletas de Demarato	Utilizada por Xerxes, rei da Persia. Segundo Singh (2008), consistia em raspar a cera de um par de *tabuletas(rodapé)* de madeira, e escrever a mensagem e ser ocultada, depois, cobria-se novamente as tabuletas com cera para que a mensagem não fosse percebida.
Histeau e o Mensageiro	Consiste em raspar a cabeça do individuo, escrever a mensagem e esperar o cabelo crescer novamente.
Técnica de Giovanni Porta	Técnica desenvolvida pelo cientista italiano Giovanni Porta. A

	técnica consistia em esconder mensagens dentro de ovos cozidos. Para isso, utilizava-se uma tinta contendo uma onça (28,35g) de Alúmen, Sulfato duplo de potássio e Alumínio hidratado, diluída em cerca de meio litro de vinagre. A solução penetrava na casca e se depositava sobre a superfície branca do ovo, a clara cozida. Depois, bastava o destinatário descascar o ovo para ler a mensagem JÚNIOR E AMORIM (2008, p.38).
Micropontos	Os micropontos são fotografias que foram reduzidas para imagens com cerca de 1 milímetro de diâmetro, se tornando microfotografias. Os micropontos ficaram populares e foram muito utilizados na 2ª Guerra Mundial, onde eram colados em pontos finais das frases ou no pingo de alguma letra 'i' da mensagem. (Jasper, 2009). O processo consistia em fotografar a mensagem e depois reduzi-la ao tamanho aproximado de um selo. Em seguida a imagem é reduzida por um microscópio reverso, ficando com um milímetro de diâmetro. Essa imagem ou o seu negativo era então colocada com o auxílio de uma seringa ou agulha na mensagem que iria servir como portadora e enviada ao seu destino (Kipper, 2004).

Tabela 6 - Técnicas de esteganografia (Própria).

2.2.2. A importância da esteganografia

A esteganografia tem sua importância para a computação no que se refere a contribuir para a segurança da informação. Como foi visto no primeiro capítulo, cada vez mais há uma necessidade de se proteger informações de terceiros, geralmente, mal intencionados.

2.2.3. Principais Técnicas

Existem muitas técnicas de esteganografia digital, a seguir serão descritas algumas delas:

2.2.3.1. Ruídos

Mídias digitais, como fotografias, filmes e música, possuem uma quantidade

significativa de ruído gerada de sua conversão em sinal digital. Esconder a informação que se deseja transmitir nesse ruído é, provavelmente, a técnica esteganográfica mais utilizada (Wayner, 2002).

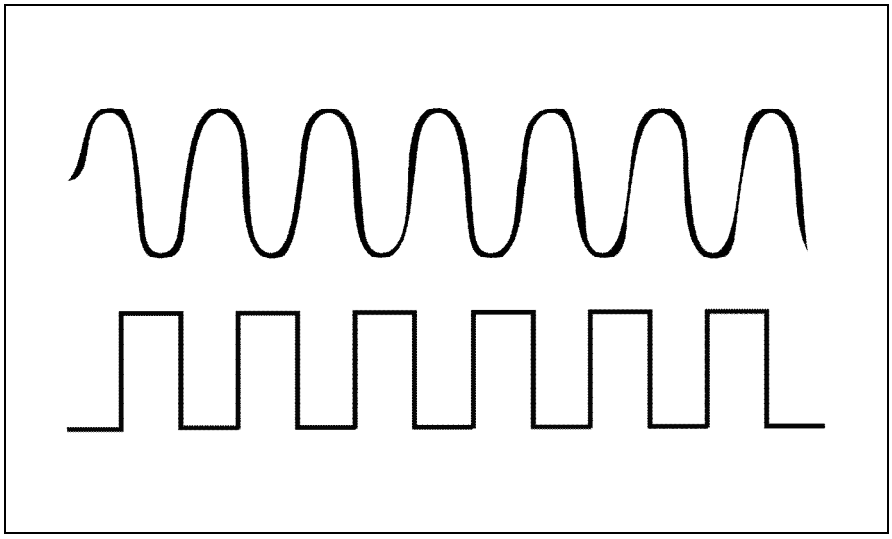

Figura 9 - Sinal analógico e sinal digital (Própria).

Nas técnicas que utilizam o ruído de conversão de sinal analógico para digital, esconde-se a mensagem como se fosse parte do ruído natural do objeto portador.

Ainda segundo Wainer (2002), fotografias coloridas digitais possuem 32 bits alocados para cada pixel. Desses 32 bits, existem 8 bits usados para guardar cada uma das quantidades de vermelho, azul e verde, ou das quantidades de ciano, magenta e amarelo de cada pixel. Com isso, são usados 24 bits. Se apenas um bit de cada uma das cores for alocado para esconder informação, essa quantidade corresponderá a 10% de todo o arquivo.

O problema desta técnica é que os formatos de mídias digitais muitas vezes comprimem o arquivo retirando este ruído gerado na conversão de sinal, tornando mais difícil a inserção da mensagem sem a alteração da aparência da imagem portadora.

2.2.3.2. Substituição

Como exemplo de técnica de substituição, temos a LSB, ou *Least Significant Bit*, que é uma técnica esteganográfica que consiste em ocultar informações em uma imagem inserindo a mesma nos bits menos significativos dos pixels da imagem.

Segundo Wayner (2002), trocar os bits menos significativos pode mudar a intensidade de um pixel em no máximo 1%, fazendo com que a técnica seja considerada uma ótima solução esteganográfica, pois a imagem fica praticamente inalterada ao olho humano.

Esta técnica é utilizada em imagens no formato JPG, pois as mesmas possuem

redundância. Segundo Provos *et. al.* (2003), Para cada componente de uma cor, a imagem JPEG usa a transformada de cosseno discreta (*Discrete Cosine Transform* – DCT) para transformar blocos sucessivos de 8 x 8 pixels da imagem em 64 coeficientes de uma DCT. Assim, os bits menos significativos dos coeficientes das DCT podem ser utilizados como bits redundantes para esconder uma mensagem. Esse tipo de técnica não deixa rastros perceptíveis para análises visuais do arquivo, uma vez que as modificações estão concentradas no domínio da freqüência, e não no domínio espacial.

2.2.3.3. Inserção

A técnica esteganográfica de inserção consiste simplesmente em inserir uma mensagem no objeto portador, sem se preocupar com o fato de a mensagem ficar muito mais vulnerável, já que dessa forma, técnicas de análise esteganográfica detectariam facilmente a mensagem.

2.2.3.4. Transformação de Domínio

A técnica de transformação de domínio utiliza os bits que "sobram" após uma conversão de formatos.

Segundo Kipper (2004), esta técnica é muito eficiente, pois insere a mensagem secreta nos espaços de transformação, que são espaços onde o formato original guarda certas informações que serão irrelevantes após a conversão.

Formatos de imagem como BMP, quando são convertidos para formatos comprimidos, como, por exemplo, JPG e formatos de áudio, como, por exemplo, WAV, quando são convertidos para MP3, podem ser objetos portadores desta técnica.

2.2.3.5. Geração de Cobertura

Na técnica de geração de cobertura, ao invés de se utilizar um objeto portador existente para inserir a mensagem, é gerado um novo objeto portador especialmente para este fim.

2.2.3.6. Distorção

A técnica de distorção produz uma mudança no objeto portador. Essa mudança

geralmente é a inserção de um ruído ou, no caso de imagens, a perda do foco em alguma região. Após a mudança, a mensagem é inserida nesta região problemática. Segundo Júnior e Amorim (2008), normalmente a distorção criada no objeto esteganográfico impossibilita a recuperação do estado original, sem a presença da distorção.

2.2.4. Esteganálise

Segundo Petri (2004), esteganálise são os estudos e pesquisas destinados a revelar a existência de mensagens secretas dentro de um objeto portador.

Uma vez que a esteganálise detecta a presença da mensagem secreta na imagem, segundo Wayne (2002), diversas técnicas podem ser utilizadas: destruição do conteúdo (o objeto portador e mensagem são simplesmente destruídos), adição de novas informações (novas informações sobrescrevem a mensagem secreta), Alteração do formato do arquivo (formatos diferentes alteram a estrutura do arquivo), compressão do arquivo (arquivos de imagem comprimidos descartam bits geralmente utilizados por técnicas de esteganografia).

Para detectar a mensagem, podem ser usados os seguintes métodos de ataque de esteganálise: Visual (análise apenas visual do objeto portador), Estrutural (Análise da estrutura do arquivo), Estatístico (análise estatística de redundância e diferença de bits vizinhos).

2.2.5. Desvantagens

Um das vantagens da esteganografia é o fato de uma mensagem passar despercebida, porém,também existem desvantagens no uso da técnica.

As técnicas de esteganografia necessitam sempre de um objeto portador. Por exemplo: para esconder uma mensagem utilizando a técnica LSB, é necessário uma imagem portadora. Se a intenção é enviar uma ou poucas mensagens ocultas em imagens, o fluxo de dados transmitidos não será um problema, mas imagine um sistema onde são realizadas milhares de transações por minuto. Em alguns casos, o fardo de carregar um objeto portador pode se tornar uma desvantagem para a esteganografia.

É necessário também tomar certo cuidado quando, ao utilizar uma técnica esteganográfica, a mesma não distorça o objeto portador a ponto de torná-lo facilmente detectável.

Para as técnicas que envolvem objetos portadores passíveis de conversão para

formatos comprimidos, deve-se estar ciente de que os bits utilizados para a mensagem secreta podem ser afetados pela conversão.

2.3. Considerações Finais do Capítulo

O capítulo 2, sobre criptologia e suas vertentes, mostrou que, apesar de algumas técnicas utilizadas para garantir a segurança da informação possuírem suas desvantagens, ainda assim são indispensáveis e necessitam de estudo e evolução constante para acompanhar o crescimento da estrutura computacional global.

Este capítulo também deixou clara a diferença entre criptografia e esteganografia. Itens como definições, características e principais técnicas foram abordados.

CAPITULO 3 – METODOLOGIA, IMPLEMENTAÇÃO E ANÁLISE

Este capítulo descreve a metodologia utilizada para desenvolver a proposta de implementação e análise de algoritmos esteganográficos em imagens digitais. Itens como linguagem de programação escolhida, algoritmos implementados e análise de desempenho e eficácia serão abordados a seguir.

3.1. Metodologia e Projeto

O projeto foi desenvolvido baseando-se nos algoritmos esteganográficos “Inserção Direta” e “*LSB – Least Significant Bit*”. O segundo foi desenvolvido em duas versões: LSB seqüencial e LSB randômico. O algoritmo inserção direta, apesar de simples, ilustra com clareza porque o estudo de técnicas mais avançadas de esteganografia é importante, uma vez que há uma grande diferença no resultado final quando comparado com o LSB.

Para a implementação dos três algoritmos foi escolhida a versão 1.6.0-26 da linguagem de programação Java. Esta é uma linguagem orientada a objetos, criada pela *Sun Microsystems*, que hoje pertence à *Oracle* e que contém diversas bibliotecas padrões para entrada e saída de dados (utilizadas para manipular as imagens), além de bibliotecas com métodos criptográficos e estruturas de dados prontas.

Apesar dos algoritmos implementados serem diferentes e possuírem características próprias, o procedimento para ocultar e revelar uma mensagem é o mesmo para os três. Para o processo de ocultação da mensagem temos os seguintes passos ilustrados na figura 10.

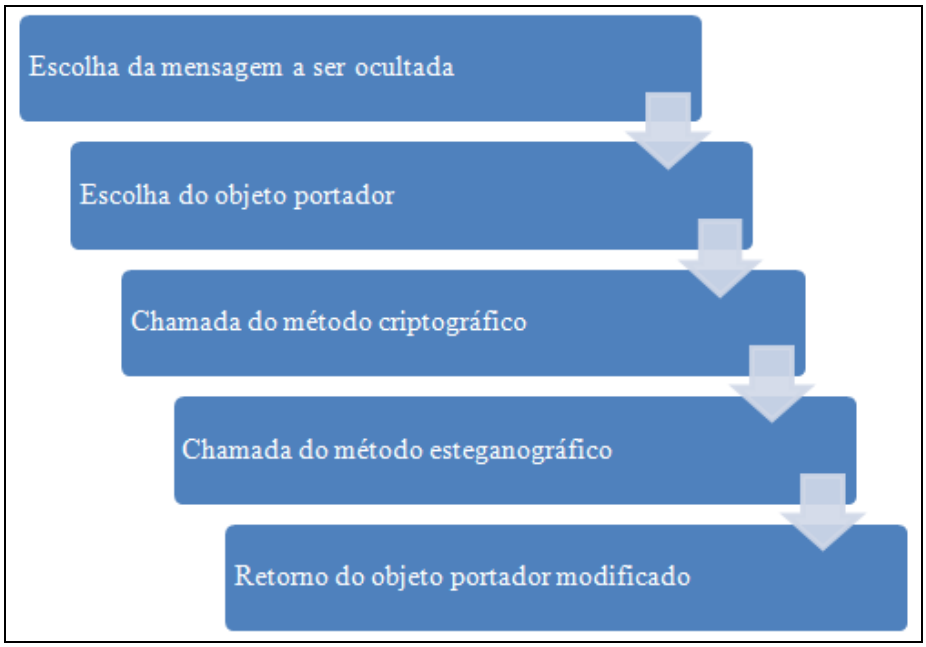

Figura 10 - Processo esteganográfico de ocultação da mensagem (Própria).

Observe que, na figura 10, antes da chamada do método esteganográfico, existe também a chamada para o método criptográfico. O método criptográfico, que utiliza o algoritmo DES, apresentado na tabela 3, na seção 2.1.3.1.4, transforma a mensagem de forma que, mesmo que ela seja descoberta através da esteganálise, seja difícil a sua compreensão. A classe que contém os métodos para criptografar e decriptografar a mensagem foram retiradas do site *Example Depot*[1].

Após a ocultação da mensagem é possível utilizar o método de revelação da imagem. Este método recupera as informações armazenadas na mensagem da seguinte forma, ilustrada na figura 11.

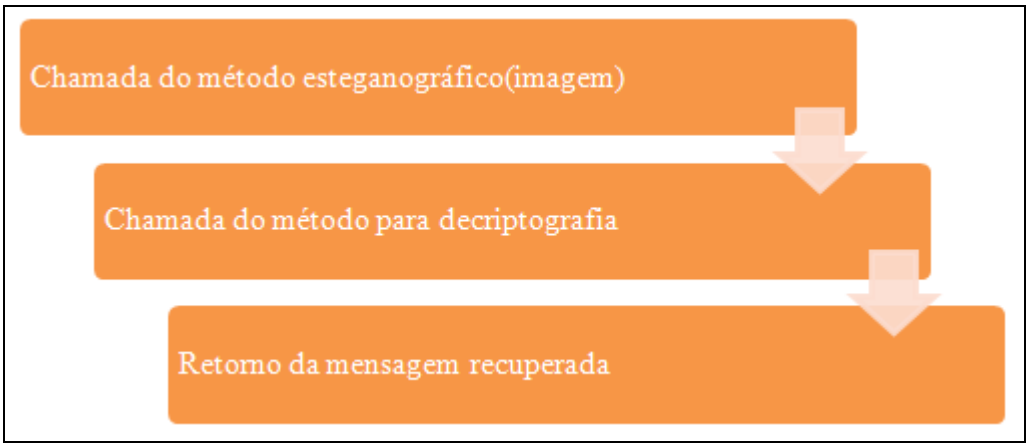

Figura 11 - Processo esteganográfico de revelação de mensagem (Própria).

Para a recuperação da mensagem oculta, é necessário, na chamada do método esteganográfico de revelação, informar a imagem que contém a mensagem. Após a extração dos dados da imagem, a mensagem é decriptografada e exibida.

3.1.1. Imagens Digitais e Sistema de Cores RGB

Imagens digitais são representadas por matrizes de tamanho definido. Cada célula dessas matrizes armazena um valor de cor que compõe um pixel da imagem através do sistema RGB (*Red* (vermelho), *Green* (verde), *Blue* (azul)). Para o uso da esteganografia em imagens digitais, é necessária a manipulação desses pixels e valores de cor. No caso do algoritmo LSB, por exemplo, é necessário alterar os bits menos significativos de vermelho, verde e azul.

[1] Example Depot – Code Samples. Disponível em: http://exampledepot.com/egs/javax.crypto/DesString.html.

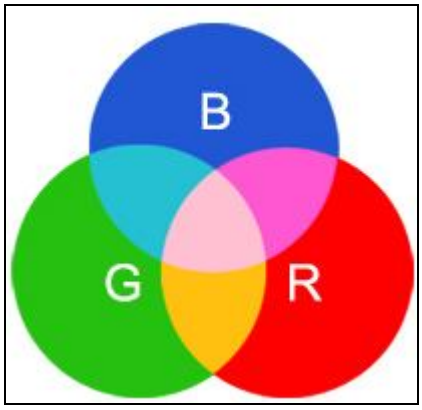

Figura 12 - Vermelho, Verde e Azul como cores primárias (Sistema RGB) (ER Associados, 2011).

No sistema RGB, as outras cores são o resultado da combinação do vermelho, verde e azul, figura 12. E, digitalmente, isso é representado por números. Cada uma das três cores que formam o sistema RGB são representadas por números que vão de 0 ao 255” (ER Associados, 2011).

3.2. Algoritmos

Esta seção do trabalho explica o funcionamento de cada algoritmo implementado. São tratados detalhes como lógica de funcionamento, dados de entrada e saída e resultados obtidos com a aplicação dos mesmos.

3.2.1. Inserção Direta

A técnica de inserção direta é de certa forma, uma técnica de esteganografia primitiva se comparada com outras mais elaboradas. Esta técnica consiste em simplesmente inserir a mensagem no objeto portador, sem se preocupar com o grau de modificação a que o objeto será submetido.

A figura 13 contém o código do método de ocultação do algoritmo de inserção direta implementado para este trabalho.

```
1  public void oculta(String mensagem){
2
3          mensagem = mensagem + "*";
4
5          BufferedImage img = null;
6          // cria vetor do tamanho da mensagem
7          int[] vetor = new int[mensagem.length()];
8          char letraC = 0;
9          // atribui letras da mensagem num vetor de int
10         for (int i = 0; i < mensagem.length(); i++) {
11             letraC = mensagem.charAt(i);
12             vetor[i] = letraC;
13         }
14
15         try {
16             // lê para um buffer uma imagem existente
17             img = ImageIO.read(new File("teste.bmp"));
18             int k = 0;
19             int letra = 0;
20             //percorrer os pixels da imagem
21             for (int i = 0; i < img.getWidth(); i++) {
22                 for (int j = 0; j < img.getHeight(); j++) {
23                     if (k < vetor.length) {
24                         letra = vetor[k];
25                         img.setRGB(i, j, letra);
26                         k++;
27                     }
28                 }
29             }
30             // grava imagem alterada
31             ImageIO.write(img, "BMP", new File("destinonovo.BMP"));
32         } catch (IOException e) {
33             e.printStackTrace();
34         }
35 }
```

Figura 13 - Algoritmo de inserção direta – método de ocultação (Própria).

Este método recebe como parâmetro uma mensagem a ser ocultada, e, após carregar uma imagem portadora, cria um vetor de inteiros que recebe o valor *ASCII* (*American Standard Code for Information Interchange*) das letras da mensagem. Dessa forma, o vetor terá, após o laço *For* da linha 10, a mensagem inteira em suas posições.

Os laços da linha 21 e 22 percorrem a imagem verificando primeiro se o vetor ainda contém letras a ser adicionadas na imagem e, caso tenha, substitui o valor do pixel nos índices corrente pelo valor inteiro da letra. Após a inserção de todas as letras da frase contidas no vetor, a imagem é salva com as alterações que foram realizadas.

Já o método de revelação, que pode ser observado na figura 14, carrega a imagem portadora com a mensagem oculta e a percorre recuperando as letras da mensagem até encontrar o caractere especial de condição de parada. Toda vez que uma letra é recuperada, a mesma é adicionada no final de uma variável do tipo *String*.

```java
public String revela() {
        // cria um buffer vazio
        BufferedImage img = null;
        int cont = 0;
        try {
            // lê para um buffer uma imagem existente
            img = ImageIO.read(new File("destinonovo.bmp"));
            String mensagem = "";
            int letra = 0;
            //percorrer os pixels da imagem
            for (int i = 0; i < img.getWidth(); i++) {
                for (int j = 0; j < img.getHeight(); j++) {
                    letra = (img.getRGB(i, j));
                    char teste = (char) letra;
                    if (teste == 42) {
                        cont++;
                    }
                    if (cont == 0) {
                        mensagem += teste;
                    } else {
                        break;
                    }
                }
            }
            return mensagem;
        } catch (IOException e) {
            e.printStackTrace();
            return "Erro ao revelar mensagem";
        }
    }
```

Figura 14 - Algoritmo de inserção direta – método de revelação (Própria).

O método principal, que faz as chamadas dos dois métodos descritos acima pode ser observado na figura 15.

```java
public static void main(String args[]){

        Insercao t = new Insercao();

        // criptografa
        SecretKey key = KeyGenerator.getInstance("DES").generateKey();
        DesEncrypter encrypter = new DesEncrypter(key);
        String mensagem_original = "MENSAGEM A SER OCULTADA"
        String mensagem_criptografada = encrypter.encrypt(mensagem_original);
        // esconde
        t.oculta(mensagem_criptografada);

        // remove
        String mensagem_encontrada = t.revela();

        // descriptografa
        System.out.println("Mensagem revelada: " + encrypter.decrypt(mensagem_encontrada));
}
```

Figura 15 - Algoritmo de inserção direta - método principal (Própria).

Na linha 6, uma chave criptográfica para o algoritmo DES é gerada e armazenada.

Esta chave é utilizada na linha 7 para criar um objeto da classe Criptografia que contém os métodos de cifragem e decifragem. Depois de definir os objetos das classes utilizadas, é escolhida uma mensagem a ser ocultada no objeto portador.

O método que cifra a mensagem é chamado na linha 9, passando a mensagem como parâmetro. Após cifrar a mensagem, o método esteganográfico de inserção direta é chamado na linha 11. Neste ponto, a imagem foi modificada e salva no diretório escolhido. As linhas 11 e 14 chamam, respectivamente, o método que recupera e retorna a mensagem oculta e o método que decifra a mensagem.

3.2.2. LSB Sequencial

O algoritmo LSB seqüencial difere do de “Inserção Direta” no que se refere ao cuidado de não mudar drasticamente a aparência do objeto portador. Esta técnica é mais elaborada, pois utiliza os bits menos significativos de cada cor do sistema RGB para inserir as letras da mensagem. A figura 16 ilustra o processo esteganográfico através do método LSB.

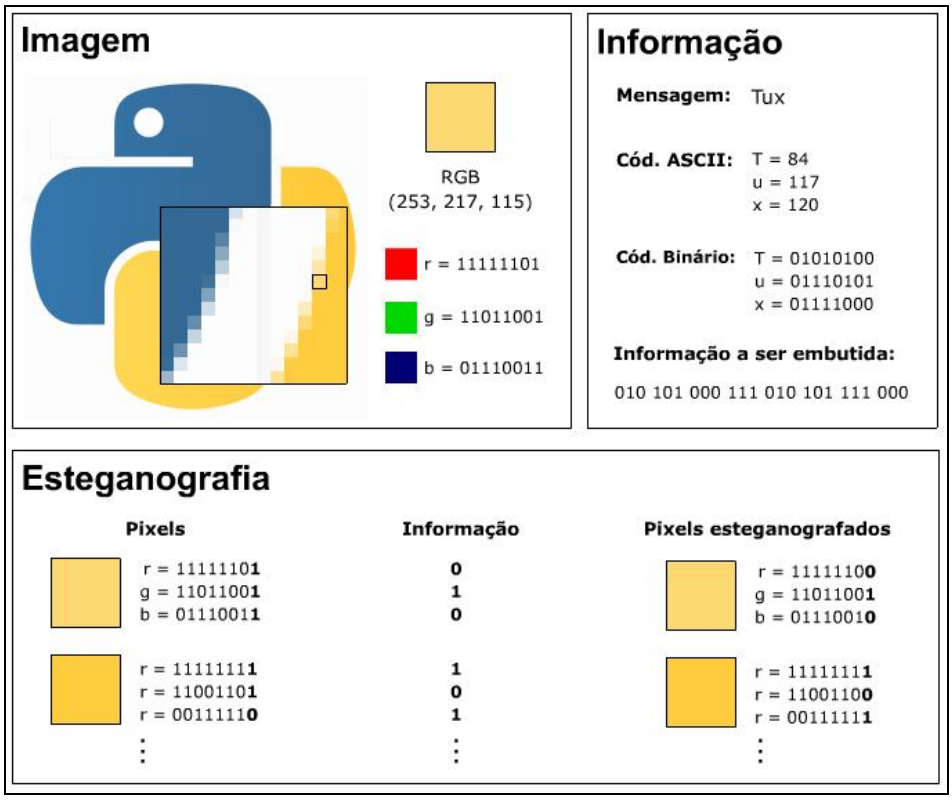

Figura 16 - Processo esteganográfico (Viva o Linux, 2011).

Como somente os bits menos significativos de cada cor do pixel é modificado, o resultado é uma imagem muito parecida com a original.

Os bits das cores dos pixels utilizados pelo algoritmo desenvolvido neste trabalho são os descritos na Tabela 7. As cores vermelho e verde “cedem”, cada uma, 2 bits para a inserção da letra, enquanto a cor azul cede 4.

	BITS UTILIZADOS
RED	XXXXXX[XX]
GREEN	XXXXXX[XX]
BLUE	XXXX[XXXX]

Tabela 7 - Bits utilizados no algoritmo LSB sequencial (Própria).

A figura 17 mostra, a partir da linha 1, as principais diferenças entre o “Inserção Direta” e o LSB Sequencial. No LSB Sequencial, o pixel é decomposto em três partes, cada parte referente a uma cor: vermelho, verde e azul. Depois de extraídos os valores individuais das cores do pixel, os mesmos são deslocados para a direita e depois para a esquerda, para que os bits menos significativos fiquem zerados.

```
1  for (int i = 0; i < img.getWidth(); i++) {
2       for (int j = 0; j < img.getHeight(); j++) {
3           if (k < vetor.length) {
4               letra = vetor[k];
5               Color pixel = new Color(img.getRGB(i, j));
6               // separa as cores do pixel
7               r = pixel.getRed();
8               g = pixel.getGreen();
9               b = pixel.getBlue();
10              pixel = new Color(r, g, b);
11              // limpa os 2 bits menos dignificativos de RED
12              r = r >> 2;
13              r = r << 2;
14              .. Mesmo processo para o Green
15              // limpa os 4 bits menos dignificativos de BLUE
16              b = b >> 4;
17              b = b << 4;
18              // RED
19              par1 = letra >> 6;
20              // par1 recebe & bit a bit com 00000011
21              par1 = par1 & 3;
22              newR = r | par1;  // | bit-a-bit
23              ... Mesmo processo para o Green ...
24              // BLUE
25              quart = letra;
26              quart = quart & 15; // AND bit a bit com 00001111
27              newB = b | quart;
28              // pixel recebe novo valor de cor baseado em RGB
29              pixel = new Color(newR, newG, newB);
30              // seta nova cor
31              img.setRGB(i, j, pixel.getRGB());
32              k++;
33          }
34      }
35  }
```

Figura 17- LSB Sequencial (Própria).

Por exemplo, para zerar os dois bits menos significativos da cor verde de um pixel qualquer, o seguinte processo é realizado:

1. Valor original de verde: 10010101**11**
2. Valor de verde com deslocamento 2 à direita: **00**100101**01**
3. Valor de verde com deslocamento 2 à esquerda: 10010101**00**

Desta forma, os dois bits menos significativos ficam zerados para receber o novo valor. Para inserir o valor da letra, é necessário deixar a parte do byte da letra que se deseja inserir no valor da cor, nas posições menos significativos. Por exemplo:

- Letra: “A”, Valor ASCII: 65 (01000001).

Para que os dois bits mais significativos de “A” fiquem nas duas últimas posições, é necessário que se faça um deslocamento de 6 casas:

1. Valor original de “A”: 01000001
2. Valor de “A” com deslocamento 6: 000000**01**

Após deslocar os 2 bits mais significativos para as duas ultimas posições, um “AND” é realizado com o valor 00000011, para que fiquem no valor, somente os bits que serão inseridos no valor da cor.

3. 000000**01** AND 000000**11** == 000000**01**

Depois de deixar os 2 bits mais significativos de “A” nas duas ultimas posições e remover todos os outros possíveis uns do conjunto de bits, é necessário fazer um “OU” com o valor de verde que foi modificado para receber parte da letra.

4. 00000001 OU 10010100 == 10010101 valor de verde com a primeira parte de “A” nos dois bits menos significativos.

Então, tem-se a seguinte situação: os dois bits menos significativos do valor de verde do pixel foram zerados para receber a primeira parte da letra “A”. Este processo é repetido com os outros bits da letra até que a mesma seja armazenada por inteira no pixel.

Para o processo de revelação da mensagem oculta, é necessário que somente os bits referentes à letra da mensagem sejam extraídos da cor do pixel.

Por exemplo: se a letra “A” está escondida em um determinado pixel, para extraí-la, seria necessário o seguinte processo:

*A = 01000001 (Letra que foi inserida no pixel)

1. Recupera “01” de Pixel(X,Y) - RED = 11000101

2. Recupera “01” de Pixel(X,Y) - GREEN = 10010000

3. Recupera “0001” Pixel(X,Y) - BLUE = 00000001

4. Concatena os bits extraídos: 01 + 00 + 0001 = 01000001 (valor de “A”).

A figura 18 mostra a lógica utilizada para extrair a letra do pixel.

```
if (rr.length() >= 2) { // rr é o valor de RED
   // pega os dois últimos
   // * nrr recebe somente os valores da letra
   nrr = rr.substring(rr.length() - 2, rr.length());
   // senão, é porque só os valores da letra são significantes
} else {
   nrr = rr;
}
// mesmo processo para green
if (gg.length() >= 2) { // gg é o valor de Green
   ngg = gg.substring(gg.length() - 2, gg.length());
} else {
   ngg = gg;
}
// Se tiver 4 ou mais digitos, pega os ultimos 4
if (bb.length() >= 4) { // bb é o valor de Blue
   nbb = bb.substring(bb.length() - 4, bb.length());
   // senão, é porque somente os bits da letra são significantes
} else {
   nbb = bb;
}
// concatena os bits da letra
String nova = nrr + ngg + nbb;
```

Figura 18 - Extração dos bits da letra que foram ocultados nos pixels da imagem (Própria).

3.2.3. LSB Randômico

A desvantagem do LSB Seqüencial, que foi visto no tópico anterior é que, após a inserção da imagem, como os bits utilizados para armazená-la são todos vizinhos, facilita o trabalho da esteganálise. Para resolver esse problema, foi implementado o algoritmo LSB Randômico, que grava as letras da mensagem em posições aleatórias geradas através do método *Random()* da linguagem Java, utilizando uma semente geradora. O processo de geração das posições dos pixels que receberão as letras pode ser visto na figura 19.

```
1  Random gerador = new Random(seed);
2  // novo objeto da classe struct
3  Struct s = new Struct();
4
5  while (k < mensagem.length()) {
6    // I e J recebem as posições da matriz (dentro de seus limites)
7    indice_i = gerador.nextInt(imgWidth);
8    indice_j = gerador.nextInt(imgHeight);
9    // s recebe novo
10   s = new Struct();
11   // adiciona indices na struct
12   s.i = indice_i;
13   s.j = indice_j;
14   // zera variavel de auxílio
15   cont = 0;
16   // verifica se a posição já existe na lista ou se é posição [0][0] ou [0][1] ou [0][2]
17   for (Struct n : list) {
18     if ((n.i == s.i && n.j == s.j) || (n.i == 0 && n.j == 0)
19         || (n.i == 0 && n.j == 1) || (n.i == 0 && n.j == 2)) {
20        // caso a posição gerada caia em alguma das condições do IF, então cont é incrementado
21        cont++;
22     }
23   }
24   // se cont for igual a 0 (zero), significa que a posição gerada randomicamente é inédita
25   if (cont == 0) {
26       // adiciona na lista
27       list.add(s);
28       // incrementa índice do laço somente se adicionar na lista
29       k++;
30   }
31 }
```

Figura 19 - Geração de posições aleatórias para armazenamento da mensagem (Própria).

Na linha 3, é criado um objeto da classe "*Struct*", essa classe foi criada para armazenas em uma estrutura de dados de lista, as posições geradas aleatoriamente.

Depois de geradas as posições randômicas, as letras da mensagem são gravadas nessas posições. Porém, para recuperá-las posteriormente, é necessário que se conheça a semente geradora do método *Random()*, por isso, a semente é gravada em uma posição conhecida na imagem.

O método revela do LSB Randômico recupera essa semente, gera novamente as posições e extrai a mensagem, a lógica utilizada para geração das posições é a mesma do método da figura 19, e o processo de extração da mensagem é o mesmo da figura 18, vista no tópico 3.2.2.

3.3. Análise

Os algoritmos implementados possuem características próprias, vantagens e desvantagens. O algoritmo de Inserção Direta, por exemplo, não é tão eficaz quanto o LSB Randômico, porém é mais rápido, gastando menos tempo de processamento.

Para que fiquem mais claras as vantagens e desvantagens de cada um, no item a seguir, serão feitas algumas execuções dos algoritmos com marcação de tempo e análise de resultados.

1. Algoritmo: **<u>Inserção Direta</u>**

- Objeto portador: Imagem no formato .BMP
- Mensagem a ser ocultada: *"Esteganografia é a arte de esconder mensagens e informações, tendo como objetivo a comunicação em segredo. A esteganografia grava uma mensagem confidencial dentro de outra, que funciona como uma portadora, só que mais extensa. O objetivo é alterar a mensagem portadora de tal forma que o resultado seja imperceptível (Petri, 2004). Uma das vantagens da esteganografia é o fato de uma mensagem passar despercebida, porém, também existem desvantagens no uso da técnica. As técnicas de esteganografia necessitam sempre de um objeto portador. Por exemplo, para esconder uma mensagem utilizando a técnica LSB, é necessária uma imagem portadora. Se a intenção é enviar uma ou poucas mensagens ocultas em imagens, o fluxo de dados transmitidos não será um problema, mas imagine um sistema onde são realizadas milhares de transações por minuto. Em alguns casos, o fardo de carregar um objeto portador pode se tornar uma desvantagem para a esteganografia. É necessário também tomar certo cuidado quando, ao utilizar uma técnica esteganográfica, a mesma não distorça o objeto portador a ponto de torná-lo facilmente detectável. Para as técnicas que envolvem objetos portadores passíveis de conversão para formatos comprimidos, deve-se estar ciente de que os bits utilizados para a mensagem secreta podem ser afetados pela conversão."*
- Tempo de execução: 397ms
- Imagem Portadora sem a mensagem:

Figura 20 - Imagem portadora antes da inserção da mensagem (Algoritmo Inserção Direta) (Própria).

- Imagem Portadora com a mensagem:

Figura 21 - Imagem portadora depois da inserção da mensagem (Algoritmo Inserção Direta) (Própria).

- Tamanho original da imagem: 350Kb
- Tamanho da imagem modificada: 350Kb

2. Algoritmo: **LSB Sequencial.**

- Objeto portador: Imagem no formato ".BMP".
- Mensagem a ser ocultada: *"Esteganografia é a arte de esconder mensagens e informações, tendo como objetivo a comunicação em segredo. A esteganografia grava uma mensagem confidencial dentro de outra, que funciona como uma portadora, só que mais extensa. O objetivo é alterar a mensagem portadora de tal forma que o resultado seja imperceptível (Petri, 2004). Uma das vantagens da esteganografia é o fato de uma mensagem passar despercebida, porém, também existem desvantagens no uso da técnica. As técnicas de esteganografia necessitam sempre de um objeto portador. Por exemplo: para esconder uma mensagem utilizando a técnica LSB, é necessária uma imagem portadora. Se a intenção é enviar uma ou poucas mensagens ocultas em imagens, o fluxo de dados transmitidos não será um problema, mas imagine um sistema onde são realizadas milhares de transações por minuto. Em alguns casos, o fardo de carregar um objeto portador pode se tornar uma desvantagem para a esteganografia. É necessário também tomar certo cuidado quando, ao utilizar uma técnica esteganográfica, a mesma não distorça o objeto portador a ponto de torná-lo facilmente detectável. Para as técnicas que envolvem objetos portadores passíveis de conversão para formatos comprimidos, deve-se estar ciente de que os bits utilizados para a mensagem secreta podem ser afetados pela conversão."*

- Tempo de execução: 423ms
- Imagem Portadora sem a mensagem:

Figura 22 - Imagem portadora antes da inserção da mensagem (Algoritmo LSB Sequencial) (Própria).

- Imagem Portadora com a mensagem:

Figura 23 - Imagem portadora depois da inserção da mensagem (Algoritmo LSB Sequencial) (Própria).

- Tamanho original da imagem: 350Kb
- Tamanho da imagem modificada: 350Kb

3. Algoritmo: **LSB Randômico.**

- Objeto portador: Imagem no formato .BMP
- Mensagem a ser ocultada: *"Esteganografia é a arte de esconder mensagens e informações, tendo como objetivo a comunicação em segredo. A esteganografia grava uma mensagem confidencial dentro de outra, que funciona como uma portadora, só que mais extensa. O objetivo é alterar a mensagem portadora de tal forma que o resultado seja*

imperceptível (Petri, 2004). Uma das vantagens da esteganografia é o fato de uma mensagem passar despercebida, porém, também existem desvantagens no uso da técnica. As técnicas de esteganografia necessitam sempre de um objeto portador. Por exemplo: para esconder uma mensagem utilizando a técnica LSB, é necessária uma imagem portadora. Se a intenção é enviar uma ou poucas mensagens ocultas em imagens, o fluxo de dados transmitidos não será um problema, mas imagine um sistema onde são realizadas milhares de transações por minuto. Em alguns casos, o fardo de carregar um objeto portador pode se tornar uma desvantagem para a esteganografia. É necessário também tomar certo cuidado quando, ao utilizar uma técnica esteganográfica, a mesma não distorça o objeto portador a ponto de torná-lo facilmente detectável. Para as técnicas que envolvem objetos portadores passíveis de conversão para formatos comprimidos, deve-se estar ciente de que os bits utilizados para a mensagem secreta podem ser afetados pela conversão."

- Tempo de execução: 498ms.
- Imagem Portadora sem a mensagem:

Figura 24 - Imagem portadora antes da inserção da mensagem (Algoritmo LSB Randômico) (Própria).

- Imagem Portadora com a mensagem:

Figura 25 - Imagem portadora depois da inserção da mensagem (Algoritmo LSB Randômico) (Própria).

- Tamanho original da imagem: 350Kb
- Tamanho da imagem modificada: 350Kb

A execução dos algoritmos foi realizada em um computador *Intel Core* I3 (2.24Ghz – 2Gb DDR3) e os tempos de execução geraram o gráfico 1.

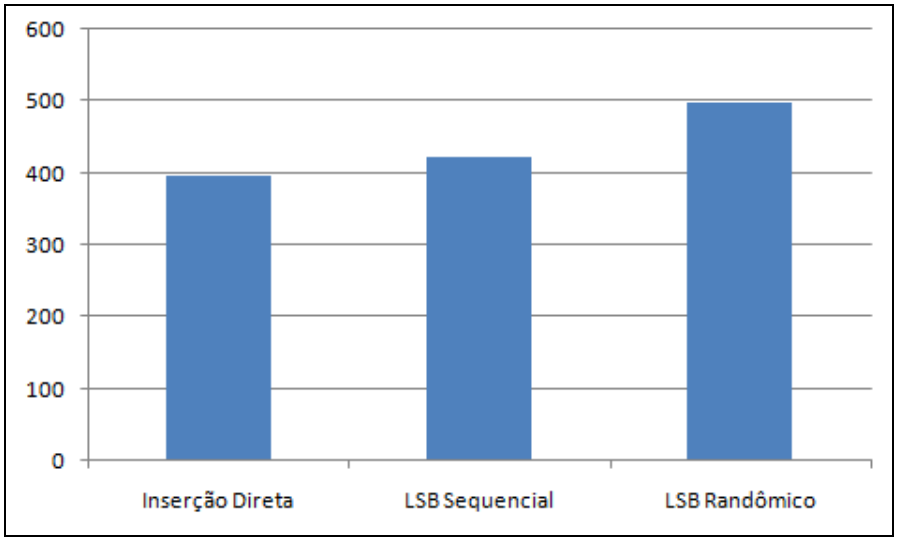

Gráfico 1 - Gráfico do tempo de execução dos algoritmos (Própria).

Já o tamanho da imagem portadora, antes e depois da inserção da mensagem foi igual para os três algoritmos, como mostra o gráfico 2.

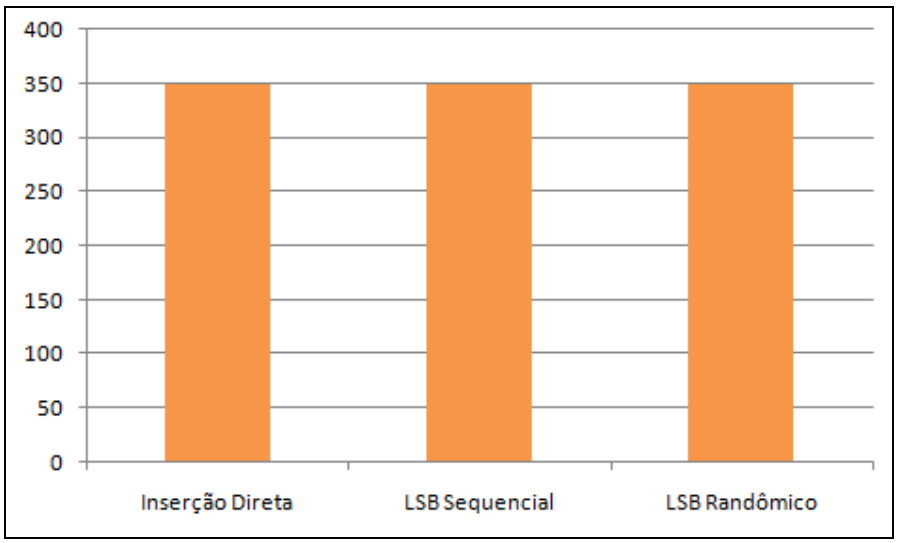

Gráfico 2 - Gráfico do tamanho da imagem portadora antes e depois da inserção da mensagem (em Kbytes) (Própria).

CAPITULO 4 – CONCLUSÕES

Ao observar os resultados obtidos com a execução dos algoritmos, no que se refere a modificação da imagem portadora após a inserção da imagem, pode-se dizer que o algoritmo de inserção direta não é uma opção viável para o objetivo principal da esteganografia, que é ocultar a existência do texto na imagem, pois a mesma sofreu uma alteração muito grande após a inserção. Esta alteração na imagem portadora não foi tão grande para os algoritmos LSB Sequencial e LSB Randômico, como pode ser visto nas figuras 22 e 23, 24 e 25, respectivamente. Mas, apesar da imagem portadora do algoritmo LSB Sequencial ser praticamente igual a do LSB Randômico a olho nu, técnicas de esteganálise poderiam detectar sem muitas dificuldades a presença na mensagem no algoritmo LSB Sequencial. Por isso, a melhor variável na questão de alteração de imagem portadora e dificuldade de detecção é o algoritmo LSB Randômico.

Outra questão importante que deve ser observada na imagem portadora é o tamanho do arquivo modificado em relação ao arquivo original. No caso dos três algoritmos, não houve aumento de tamanho de arquivo em disco, o que pode ser considerado um ponto positivo.

Os objetivos propostos para o trabalho foram alcançados, e concluiu-se, através dos resultados obtidos na análise, que a esteganografia como ferramenta para segurança da informação é viável. Além disso, O trabalho como fonte de informação para novas pesquisas também foi uma proposta alcançada.

4.1. Sugestões de continuidade

- Implementação de outras técnicas de esteganografia, como transformação de domínio e geração de cobertura.
- Construção de uma ferramenta com interface gráfica e comunicação usando *sockets* para ocultar mensagens e enviá-las pela internet.
- Esteganografia em arquivo de imagem no formato .JPEG
- Pesquisa aprofundada sobre esteganográfica em áudio e vídeo

REFERÊNCIAS

COMPUTER WORLD, **Gartner revê gastos globais com TI em 2010**, Disponível em: <http://computerworld.uol.com.br/negocios/2010/04/12/gartner-gasto-global-com-ti-deve-crescer-5-3-em-2010>. Acesso em: 04/2010.

DEVMEDIA. Disponível em: <http://www.devmedia.com.br/>. Acesso em: 11/2011.

ER. ASSOCIADOS. **O que é RGB?** . Disponível em: <http://erassociados.com/central-conteudo/detalhe/conteudo-o-que-e-rgb>. Acesso em: 11/2011.

GOOGLE. **Google**. Disponível em: <www.google.com.br>. Acesso em: 03/2011.

HOWSTUFFWORKS, **How Cloud Computing Works**, Disponível em: <http://computer.howstuffworks.com/cloud-computing/cloud-computing.htm>. Acesso em: 03/2011.

JASPER, N. A.. História, **Técnicas e Classificaçãode Algoritmos Esteganográficos.** Monografia apresentada à Faculdade de Tecnologia de São Paulo (FATEC-SP). 2009.

JÚNIOR, J. G. R.; AMORIM, E. S.. **Esteganografia: integridade, confidencialidade e autenticidade.** São Bernardo do Campo, 2008.

KIPPER, G.. **Investigator's Guide to Steganography**, Auerbach Publications, 2004.

LYRA, M. R.. **Segurança e Auditoria em Sistemas de Informação.** 1. ed. Rio de Janeiro: Editora Ciência Moderna Ltda., 2008.

MICROSOFT – PATTERNS & PRACTICES. Patterns & Practices: Developer Center Home. Disponível em: <http://msdn.microsoft.com/en-us/practices/bb190332>. Acesso em: 07/2011.

PEREIRA, F. D. ; ORDONEZ, E.D.M.; CHIARAMONTE, R.B..**Criptografia em Software e Hardware.** 1. ed. São Paulo: Novatec, 2005. v. 2000. 288 p.

PROVOS, N.; HONEYMAN, P. **Hide and Seek: An Introduction to Steganography,** IEEE Security & Privacy, 1(3): 32-44, maio/junho 2003.

RDS INFORMÁTICA. **Proteja-se dos Principais Tipos de Ataque.** Disponível em: < http://www.rdsinfor.com.br/proteja-se-dos-principais-tipos-de-ataque/>. Acesso em: 04/2011.

SASSI, M. **Investimentos retomam ritmo pré-crise, segurança da informação na pauta.** Disponível em: <http://marcosassi.com.br/investimentos-retomam-ritmo-pre-crise>. Acesso em: 03/2011.

VIVA O LINUX. **Esteganografia e Esteganálise: transmissão e detecção de informações ocultas em imagens digitais.** Disponível em:<http://www.vivaolinux.com.br/artigo/Esteganografia-e-Esteganalise-transmissao-e-deteccao-de-informacoes-ocultas-em-imagens-digitais>. Acesso em: 11/2011.

WAYNER, P. **Disappearing Cryptography – Information Hiding: Steganography and Watermarking.** Morgan Kaufmann Publisher, 2ª Edição, maio 2002.