

UNIVERSIDADE FEDERAL DE GOIÁS – UFG
CAMPUS CATALÃO – CC
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – DCC

Bacharelado em Ciência da Computação

Projeto Final de Curso

**Esteganografia em Áudio e Imagem utilizando a
técnica LSB**

Autor: Hugo Silva Cantanhede

Orientador: Ms. Márcio Antônio Duarte

Hugo Silva Cantanhede

Esteganografia em Áudio e Imagem utilizando a técnica LSB

Monografia apresentada ao Curso de
Bacharelado em Ciência da Computação da
Universidade Federal de Goiás Campus Catalão
como requisito parcial para obtenção do título de
Bacharel em Ciência da Computação

Área de Concentração: Segurança da Informação

Orientador: Ms. Márcio Antônio Duarte

S. Cantanhede, Hugo

**Esteganografia em Áudio e Imagem utilizando a técnica LSB/Ms.
Márcio Antônio Duarte- Catalão - 2009**

Número de paginas: 62

Projeto Final de Curso (Bacharelado) Universidade Federal de Goiás, Campus Catalão, Curso de Bacharelado em Ciência da Computação, 2009.

Palavras-Chave: 1. Esteganografia. 2. Segurança da Informação. 3. Técnica LSB

Hugo Silva Cantanhede

Esteganografia em Áudio e Imagem utilizando a técnica LSB

Monografia apresentada e aprovada em _____ de _____
Pela Banca Examinadora constituída pelos professores.

Ms. Márcio Antônio Duarte – Presidente da Banca

Dr. Alexsandro Santos Soares

Veríssimo Guimarães Junior

Dedico este trabalho de conclusão de curso
à minha mãe que não mediu esforços para
que minhas realizações pessoais
pudessem tornar realidade.

AGRADECIMENTOS

Agradeço a Deus por ter me acompanhado ao longo da trajetória do curso e da minha vida.

À minha família, especialmente minha mãe e a meu irmão pela dedicação, incentivo e por serem parte fundamental em minha vida.

Ao Orientador Márcio Antônio Duarte.

Aos professores do Curso de Ciência da Computação pelos ensinamentos.

Aos colegas que de uma forma ou de outra contribuíram com mais essa conquista.

RESUMO

Cantanhede, H. Esteganografia em Áudio e Imagem utilizando a técnica LSB.

Curso de Ciência da Computação, Campus Catalão, UFG, Catalão, Brasil, 2009, 62p.

A proteção digital é uma área de pesquisa que está crescendo em vários campos da ciência. A esteganografia, a arte da escrita escondida, se inclui nesse processo de novas áreas de pesquisa. Esse projeto final de curso apresenta a implementação de um software utilizando a técnica esteganográfica LSB (*Least Significant Bit*) que inclui mensagem texto em arquivos de imagem e em um arquivo de áudio com o propósito de analisar o arquivo original com o arquivo que possui a imagem escondida e apontar as suas diferenças relevantes além de, propiciar uma comunicação segura pela rede de computadores (*Internet*).

Palavras-Chaves: Esteganografia, Segurança da Informação, Técnica LSB

Sumário

1	Introdução	1
1.1	Motivação	1
1.2	Objetivo	2
1.3	Organização da Monografia	3
2	Segurança da Informação, Esteganografia e Criptografia	4
2.1	Segurança da Informação	4
2.1.1	Medidas de Proteção	4
2.2	Esteganografia	7
2.2.1	História	7
2.3	Criptografia	9
3	Terminologia, Técnicas Esteganográficas, Aplicações da Esteganografia, Esteganálise e seus Tipos de Ataques	11
3.1	Terminologia	11
3.2	Técnicas Esteganográficas	12
3.2.1	Técnicas de inserção no bit menos significativo (LSB)	12
3.2.2	Técnicas baseadas em Algoritmos e Transformações	14
3.2.3	Técnicas de Espalhamento de Espectro	14
3.2.4	Técnicas de Esteganografia em Vídeos	14
3.3	Aplicações	15
3.3.1	Aplicações Militares	15
3.3.2	Direitos Autorais	15
3.3.3	Aplicações Escondendo o Remetente	15
3.3.4	Aplicações Médicas	15
3.3.5	Certificação e Controle de Acesso	16
3.3.6	Economia de Banda	16
3.3.7	Autenticação	16
3.3.8	Ética	17
3.4	Esteganálise	17

3.5	Tipos de Ataques	17
3.5.1	Ataques Aurais	18
3.5.2	Ataques Estruturais	18
3.5.3	Ataques Estatísticos	18
4	Implementação do Programa	19
4.1	Principais Requisitos do Modelo	19
4.2	Especificação do Protótipo	20
4.2.1	Diagrama de caso de uso	20
4.2.2	Diagrama de Classe	21
4.2.3	Diagrama de Seqüência	25
4.2.4	Diagrama de Atividade	28
4.3	Implementação	33
4.3.1	Interface	33
4.3.2	Implementação Esteganografia em Imagem	37
4.3.3	Implementação Esteganografia em Áudio	41
5	Resultado e Considerações Finais	45
6	Conclusão	51
6.1	Conclusão	51
6.2	Trabalhos Futuros	52
	Referências	53
	Apêndices	54
A	Código Fonte	55
A.1	Classe Principal	55
A.2	Classe InsercaoDaMensagem_Imagem	57
A.3	Classe DeteccaoDaMensagem_Imagem	59
A.4	Classe InsercaoDaMensagem_Audio	60
A.5	Classe DeteccaoDaMensagem_Audio	61

Lista de Figuras

3.1	Terminologia	12
3.2	Figura antes de aplicar a técnica LSB	13
3.3	Figura depois de aplicar a técnica LSB	13
4.1	Diagrama de Caso de Uso	21
4.2	Diagrama da Classe Principal	22
4.3	Diagrama da Classe InsercaoDaMensagem_Imagem	23
4.4	Diagrama da Classe DeteccaoDaMensagem_Imagem	23
4.5	Diagrama da Classe InsercaoDaMensagem_Audio	24
4.6	Diagrama da Classe DeteccaoDaMensagem_Audio	24
4.7	Diagrama de Classe	24
4.8	Diagrama de Seqüência: Inserção da Mensagem em Imagem	25
4.9	Diagrama de Seqüência: Remoção da Mensagem em Imagem	26
4.10	Diagrama de Seqüência: Inserção da Mensagem em Áudio	27
4.11	Diagrama de Seqüência: Remoção da Mensagem em Áudio	28
4.12	Diagrama de Atividade: Inserção da Mensagem em Imagem	29
4.13	Diagrama de Atividade: Remoção da Mensagem em Imagem	30
4.14	Diagrama de Atividade: Inserção da Mensagem em Áudio	31
4.15	Diagrama de Atividade: Remoção da Mensagem em Áudio	32
4.16	Interface: Inserção de Conteúdo em Imagem	33
4.17	Interface: Detecção de Conteúdo em Imagem	34
4.18	Interface: Inserção de Conteúdo em Áudio	35
4.19	Interface: Detecção de Conteúdo em Áudio	36
4.20	Comparação entre bits de uma Imagem	38
4.21	Código de inserção de mensagem em uma imagem 1	39
4.22	Código de inserção de mensagem em uma imagem 2	40
4.23	Verifica se existe alguma mensagem oculta ou condição de parada	41
4.24	Comparação entre bits do arquivo de áudio	43
4.25	Código de inserção de mensagem em áudio	44
5.1	Histograma da Imagem Original	46

5.2	Histograma da Imagem Esteganografada	46
5.3	Áudio Original	47
5.4	Áudio Esteganografado	47
5.5	Histograma da Imagem Original	48
5.6	Histograma da Imagem Esteganografada	48
5.7	Áudio Original	49
5.8	Áudio Esteganografado	49

Capítulo 1

Introdução

1.1 Motivação

Com o desenvolvimento da tecnologia ocorreu o surgimento de mecanismos com o intuito de melhorar e dar mais conforto ao dia a dia do ser humano. Equipamentos como computadores, máquinas fotográficas digitais, internet rápida, discos rígidos com Gigabytes não eram populares como são atualmente. Com a popularização desses equipamentos (principalmente dos computadores e da internet), o acesso a informação aumentou significativamente.

A detenção de informação possui um valor expressivo, já que esta pode ser considerada sinônimo de poder. Em uma sociedade informatizada como a de hoje, o maior patrimônio de uma empresa é a sua informação [DIAS, 2000]. A informação representa o diferencial entre empresas, além de significar a diferença entre o sucesso e o fracasso.

Com isso, deve-se destacar a questão da segurança, já que não se pode admitir a perda do sigilo de uma determinada informação, visto que à medida que o tempo passa, aumenta o número de técnicas capazes de violar a privacidade e a segurança de uma informação.

A segurança da informação é caracterizada pelo controle do acesso da informação, sendo que esse acesso deve ser autorizado apenas para um grupo seletivo de pessoas que possuem o direito de visualizar, criar, apagar ou modificar as informações [JASCONE, 2003].

A facilidade de acesso à informação também trouxe grandes problemas. Um dos principais problemas relacionado com a segurança é a pirataria, que é uma forma de se apropriar de tecnologia e de informação sem pagar o seu devido valor. Por conta disso, a indústria de CDs e DVDs visualiza a queda das vendas, assim como pode-se citar a disseminação das músicas no formato MP3 pela internet.

Dessa forma, é evidente a necessidade de se investir em técnicas que tornem a informação mais segura. Uma nova tecnologia nesse ramo consiste no ocultamento de informações, com o intuito de assegurar os direitos autorais além de permitir uma maior

segurança da informação.

Com relação aos CDs e DVDs não existe uma técnica eficiente capaz de evitar a cópia destes, por enquanto. Uma das possíveis soluções seria colocar dados dentro dos vídeos e das músicas com a intenção de eliminar esse tipo de problema.

Departamentos de inteligência da França tentam inibir a utilização de criptografia com o argumento de dificultar o acesso à informação por meio de escutas e rastreamento. Para isso, eles restringem a potência dos algoritmos de encriptação além de solicitar a disponibilização das chaves de segurança referentes a esses algoritmos [FLACKE, 1998].

Em contrapartida, essa ação desrespeita as leis dos direitos do cidadão que diz respeito a sua privacidade. Uma alternativa para isso seria a utilização de imagens com o objetivo de ocultar uma informação.

Dessa forma, não seria classificada como criptografia, com isso não estaria indo de encontro com os departamentos de inteligência e a informação estaria protegida. Utilizando essa alternativa, dificilmente uma mensagem seria percebida e caso fosse, dificilmente seria compreendida.

Assim, o desenvolvimento deste trabalho propões um estudo sobre algumas técnicas esteganográficas existentes e a implementação de um protótipo que realize o ocultamento de informações em áudio e imagens digitais.

1.2 Objetivo

O objetivo desse trabalho é realizar uma análise comparativa entre os arquivos originais de imagem e áudio e seus respectivos arquivos com mensagens escondidas através do desenvolvimento de um software e da utilização da técnica esteganográfica LSB (bit menos significativo). Esse software tem como finalidade, ocultar qualquer tipo de informação no formato texto em um arquivo tipo áudio ou imagem.

Dessa forma, serão apresentadas as metas que se deve alcançar na conclusão deste trabalho. São elas:

- Desenvolver um software que irá utilizar a técnica de esteganografia LSB;
- O software irá esconder um texto digitado pelo usuário em imagem e áudio;
- O software irá extrair, caso exista, um texto do arquivo de imagem e áudio e apresentá-lo para o usuário;
- Realizar uma avaliação sobre o método de esteganografia LSB aplicado em imagem e áudio.

1.3 Organização da Monografia

Este trabalho está dividido em sete capítulos, sendo este o primeiro deles.

O Capítulo 2 apresenta os principais conceitos que envolvem segurança de informação, esteganografia e criptografia.

O Capítulo 3 apresenta as terminologias esteganográficas adotadas, indica as principais técnicas esteganográficas e suas aplicações além de proporcionar os principais conceitos referentes à esteganálise e seus tipos de ataque.

O Capítulo 4 é referente à implementação do software esteganográfico baseado na técnica LSB, detalhando os diagramas e a funcionalidade do sistema.

O Capítulo 5 apresenta os resultados obtidos e as considerações finais obtidas com o programa.

O Capítulo 6 apresenta as conclusões encontradas.

Capítulo 2

Segurança da Informação, Esteganografia e Criptografia

2.1 Segurança da Informação

Esse tópico pretende explicar melhor sobre o tema segurança da informação, já que pessoas, empresas e corporações possuem a necessidade de garantir que sua informação esteja realmente em segurança, de forma a evitar que essa sofra qualquer tipo de fraude, erro, uso indevido, dano, roubo de informação e outros males que fariam com que diminuísse a produtividade dos usuários através de um ambiente menos organizado e de pouco controle sobre os recursos computacionais inviabilizando assim, qualquer aplicação crítica eficiente.

2.1.1 Medidas de Proteção

Medidas de proteção é o conjunto de técnicas ou medidas que devem ser utilizadas com o propósito de detectar ou prever um ataque a segurança. Abaixo serão especificadas algumas delas.

Segurança Física

Não basta possuir sistemas atualizados, uma boa política de segurança, profissionais qualificados e muito bem instruídos se os equipamentos não estão bem protegidos, ou seja, não basta possuir técnicas eficazes de proteção dos dados se a segurança física não for garantida [MEDEIROS, 2001]. É necessário prestar atenção em todos os detalhes como: Incêndios, desabamentos, relâmpagos, alagamentos, problemas elétricos.

Medidas muitas vezes esquecidas são muito importantes para a proteção física como: serviço de guarda, a utilização de no-break, alarmes, fechaduras, utilização de áreas restritas e circuito interno de televisão auxiliam na proteção da informação [MEDEIROS, 2001].

Sistemas de Detecção de Intrusão

Sistemas de detecção de intrusão (IDS - Intrusion Detection Systems) é um campo de pesquisa e investimento na segurança de redes que está em crescimento. Isso ocorre graças ao número de computadores que estão conectados a internet em todo o mundo além do número, cada vez maior, de ataques.

Para grande parte das aplicações atuais, está sendo inviável a utilização de métodos que diminuam a probabilidade de ataques ocasionais já que, ao sofrer um ataque de força bruta, pode causar interrupções totais ao sistema para que se possa realizar um lento e dispendioso processo de auditoria [CAMEPELLO, 2001].

Dessa forma, o IDS é uma ferramenta inteligente que possui a finalidade de ir além da prevenção de um sistema e possuir a capacidade de descobrir tentativas de invasão em tempo real, mas garantir o contínuo funcionamento do sistema mesmo em vista de alguma falha, podendo assim, apenas alertar sobre a tentativa de invasão ou até mesmo executar ações que inibem tal ataque.

As técnicas de reconhecimento de ataque podem ser divididas em duas [MEDEIROS, 2001]:

- **Sistemas Baseados em Regras:** Esse sistema baseia-se em uma base de dados que possuem assinaturas de ataques. Dessa forma, ao se perceber que um tráfego se assemelha a um dos critérios pertencentes a base de dados, ele é identificado como uma tentativa de intrusão. Porém, para que esse sistema funcione de maneira eficiente é necessária a atualização constante dessas bases de dados vista que esse tipo de ataque só detecta ataque já conhecidos;
- **Sistemas Adaptáveis:** Esse tipo de ataque utiliza técnicas como inteligência artificial para poder identificar ataques que não pertencem a sua assinatura (desconhecidos). O principal problema dessa técnica é o elevado custo e a dificuldade de gerenciamento.

Proteção Contra Malware

Normalmente, o computador se infecta com algum tipo de Malware devido a falta de conhecimento ou da curiosidade do usuário, visto que, para que grande parte desses ataques ocorram, é necessário que o usuário autorize a sua entrada no computador. Isso ocorre, na maioria das vezes, quando o usuário abre um arquivo desconhecido ou sujeito a algum tipo de risco.

Abaixo estão algumas maneiras de como se deve proceder para manter um computador afastado dos vírus [MEDEIROS, 2001].

- Nunca abrir arquivos anexados em seu e-mail de pessoas que não fazem parte da sua lista de contatos ou arquivos de contatos que não esteja esperando;

- Instalar um bom antivírus e manter suas atualizações sempre em dia, já que novas ameaças surgem todos os dias.
- Se possível, utilizar firewall ao conectar a internet. Firewall funciona como um empecilho entre duas ou mais redes (normalmente entre uma rede local e a internet). Impedindo assim, que tráfego não autorizado adentre em uma determinada rede.

Política de Segurança

Política de segurança é tática adotada pela gerencia de uma organização, com a finalidade de determinar o nível de segurança de um sistema ou da rede que será adotado para melhor proteger as suas informações [MEDEIROS, 2001].

Para estabelecer uma política de segurança de uma organização é necessário conhecer os seus objetivos para verificar suas possíveis falhas de segurança. Somente após esse passo, que se devem formalizar as regras a respeito dessas políticas em forma de documento [MEDEIROS, 2001]. Informando assim aos usuários do que podem ou não fazer no âmbito da organização.

Para desenvolver uma política de segurança eficaz é interessante seguir algumas características [MEDEIROS, 2001]:

- Faz-se necessário que esteja de acordo com as normas da organização e que seja possível a sua implementação;
- Deve ser curta já que não é necessário um grande número de páginas para formalizar tal política (no máximo três páginas). Dessa forma os usuários podem possuir o manual sempre em mão e de rápida consulta;
- A política adotada pela empresa deve ter aceitação pelos funcionários e devem segui-las à risca, independentes do cargo ocupado na organização (valendo tanto para o presidente da organização como também para o estagiário).
- Deve ser simples e de fácil entendimento por todos. O documento não deve possuir termos técnicos de difícil entendimento.

A direção e os altos cargos da empresa devem assinar tal política com a finalidade de mostrar a importância de tal atitude.

Logicamente, cada organização irá desenvolver uma política de segurança. Porém existem pontos que em todas devem existir como: Controle de Acesso; Responsabilidades; Privacidade do Usuário; Medidas que serão adotadas quando ocorrer o descumprimento das regras; Mostrar que a informação é o bem mais importante da empresa [MEDEIROS, 2001].

A política de segurança deve ser um dos elementos adotados para manter a segurança do sistema, ou seja, ele deve ser aplicado em conjunto com outras formas de proteção, pois, apenas a utilização de um conjunto de ações permitirá alcançar tal objetivo (exemplo segurança física, *malwares*).

Outras medidas de Proteção

Existem vários outros mecanismos de proteção tal como criação de cópias de seguranças, firewalls, controle de tráfego. Além desses, existem dois mecanismos que se pode destacar, são eles: A Esteganografia e a Criptografia. Ambas são técnicas que permitem com que uma informação permaneça segura independente do local que esteja (em um computador, na internet).

2.2 Esteganografia

Esteganografia é um método utilizado para ocultar uma informação, com a finalidade de impedir que pessoas não autorizadas, detectem o seu conteúdo, fazendo assim com que a comunicação ocorra em segredo [KOBUSZEWSKI, 2004]. Essa técnica de segurança da informação consiste em ocultar uma mensagem dentro de outra sendo que uma delas funciona como portadora da mensagem em questão [PETRI, 2004].

A palavra esteganografia significa a arte da escrita escondida (estegano = esconder, mascarar e grafia = escrita) [JULIO, 2007]. Existem diversas técnicas onde se pode aplicar esse método (militares, médicas, economia de banda). Nesse trabalho, será abordada a técnica LSB que será explorada melhor nos próximos capítulos.

2.2.1 História

Os primeiros registros conhecidos sobre a utilização da esteganografia são datados de 440 a.C. [ROCHA, 2003]. Dentre os primeiros registros conta-se que um homem chamado Harpagus matou uma lebre e colocou uma mensagem em suas entranhas. Dessa forma, o mensageiro se disfarçou de caçador e com isso, burlou os guardas e conseguiu conduzir a mensagem até o seu destinatário [ROCHA, 2003].

Por volta do Século V a.C., um grego chamado Histiaieus, era prisioneiro do Rei Davi, e queria convencer o seu cunhado, Aristágoras de Mileto a revoltar-se contra o rei Persa. Para passar a mensagem, a cabeça do seu mensageiro foi raspada e a mensagem tatuada. Dessa forma, esperou que o cabelo do mensageiro crescesse podendo assim ser conduzido sem problemas até a Grécia, onde a mensagem foi entregue ao destinatário [POLLON, 2007]. Esse método ainda foi muito utilizado pelos alemães na segunda guerra mundial.

Ainda na Grécia, o grego Enéas (escritor de táticas militares), inventou uma técnica denominada de astrogal. Essa técnica consiste de uma madeira com vários furos, onde cada furo representa uma letra do alfabeto. Para enviar a mensagem, o emissário passava um barbante por entre os furos para formar assim palavras ou até frases. Dessa forma, o receptor deveria percorrer o caminho dos furos para compreender a mensagem [PETRI, 2004].

Nessa mesma época, outros povos também fizeram uso de técnicas esteganográficas como os chineses e os egípcios. Os egípcios utilizavam escritas na forma de desenhos para esconder as suas informações já que, quando os mensageiros egípcios eram interceptados por inimigos, estes por sinal, não conseguiam entender os hieróglifos e pensavam que se tratava de desenhos comuns (sem nenhum significado).

Com isso não conseguia suspeitar do mensageiro [EDUARDO P. JULIO, 2007], podendo este entregar sem problemas a mensagem para o destinatário. Esse tipo de escrita é conhecido como hieróglifo.

No período da Renascença (século XV e XVI), Johannes Trithemius (1462 - 1516), um monge alemão, escreveu uma trilogia em latin denominada: "*Steganographia: hoe est ars per occultam scripturam animi sui voluntatem absentibus aperiendi certa*". No terceiro volume de sua obra, Trithemius ocultou o salmo 23 da Bíblia utilizando tabelas contendo números sendo que, essa informação só foi descoberta no século XX por pesquisadores da Universidade de Pittsburg [ROCHA, 2003]. Foi através da escrita da trilogia de Trithemius que o termo esteganografia surgiu.

A manipulação com tintas invisíveis se iniciou durante a idade média. Giovanni Porta escreveu alguns livros com receitas de tintas secretas com o propósito de camuflar algum tipo de informação [EDUARDO P. JULIO, 2007].

Durante os séculos XVI e XVII ocorreu o surgimento de uma grande quantidade de obras literárias referente à esteganografia, porém, essas técnicas necessitavam do auxílio das novas tecnologias referentes à codificação da informação que estavam sendo desenvolvidas naquela época.

As primeiras tintas eram formadas de fluídos orgânicos como, por exemplo, tintas baseadas em suco de limão, leite, urina. As tintas e os líquidos de revelação foram se tornando mais eficientes com o passar do tempo através da manipulação de elementos químicos durante a Primeira Guerra Mundial (1914-1918) [IGUCHI, 2007]. Antes disso, a revelação da maioria das mensagens era feita utilizando o calor.

Contudo, a utilização das tintas invisíveis para a transmissão de mensagens se tornou obsoleto pelo desenvolvimento de "reveladores universais" [IGUCHI, 2007]. Esses por sinal, expõem o local onde foi molhado por alguma substância e dessa forma inviabiliza a utilização de tal técnica.

Ultimamente, após um amplo período de desenvolvimento, surgiu a tinta fluorescente

que é sensível a luz ultravioleta. Esse tipo de tinta é utilizado em cheques de viagens, comumente conhecida como *traveler's check* com o intuito de evitar fraudes. Normalmente as copiadoras possuem uma grande quantidade de ultravioleta em suas luzes que reagem com a tinta fluorescente. Dessa forma quando algum cheque de viagem é foto copiado aparece escrito a palavra 'INVÁLIDO', descaracterizando, assim, o documento como sendo original e válido [IGUCHI, 2007].

Com o desenvolvimento de novas tecnologias, principalmente durante a segunda guerra mundial (1939-1945), melhorou a qualidade dos equipamentos fotográficos e propiciou novas formas de comunicação secreta. Entre essas formas de comunicação está o uso de cifras nulas, micro-pontos e o semagramas.

A técnica de Cifras nulas (null ciphers) possui como finalidade esconder uma mensagem dentro de outra, ou seja, um texto possui letras que são utilizadas para formar um novo texto sendo que as outras letras são consideradas nulas, além disso, tanto o remetente como o receptor, deve utilizar o mesmo código para que a mensagem possa ser decodificada. Como exemplo, pode-se usar a primeira letra de cada palavra para formar uma mensagem [EDUARDO P. JULIO, 2007].

Uma técnica também utilizada é a do micro-ponto. Essa técnica consiste em tirar uma fotografia de uma informação e reduzi-la para ficar do tamanho de um ponto. A foto é reduzida para que tenha, aproximadamente, 0,125 cm de diâmetro [ROCHA, 2003]. A fotografia é colocada em um sinal de pontuação (ou de acentuação) dentro de um texto.

Semagramas é uma forma de comunicação onde não se faz uso da forma escrita [ROCHA, 2003]. Durante a guerra, os americanos interceptaram um caminhão com um carregamento de relógios, com medo da disposição dos relógios ou dos seus ponteiros significarem alguma mensagem, ambos foram modificados por precaução [ROCHA, 2003].

No Brasil, principalmente depois da década de noventa, é comum encontrar traficantes que escondem drogas no corpo para não serem apanhados com a droga. As drogas normalmente são engolidas. As pessoas que se sujeitam a tal atividade são chamadas de "mulas". Esse tipo de atividade, retomando a técnica que foi desenvolvida por Porta durante o Renascimento.

2.3 Criptografia

Criptografia surgiu a aproximadamente 4 mil anos. O nome criptografia vem do grego *kyptós* (ocultar) e *gráfos* (escrita). O objetivo desta não é ocultar a existência de uma mensagem, mas sim ocultar o seu significado [PETRI, 2004].

A criptografia aumenta consideravelmente o nível de proteção de uma determinada informação tornando-a quase inviolável a possíveis ataques. Esses ataques podem ocorrer durante a troca de informações (em uma rede) ou até mesmo em computadores locais.

Criptografia consiste em cifrar um texto legível em um texto cifrado com a finalidade de não permitir a identificação do mesmo. Para compreender o texto cifrado é preciso decifrar o texto e que esse seja igual ao original. Para cifrar e decifrar é necessário utilizar algoritmos de criptografia que empregam funções matemáticas [PETRI, 2004].

Os algoritmos de criptografia fazem uso de uma chave, que é utilizada como forma de cifrar uma mensagem ou identificar a mensagem cifrada.

O interessante é que se alguém interceptar uma mensagem criptografada, ela será ilegível, seu conteúdo não identificado e dessa forma, dificilmente será obtido a mensagem original já que não possuem a chave que é utilizada para decifrar a mensagem.

Pode-se perceber que uma mensagem mesmo estando criptografada pode ser percebida (visualizada), diferente da esteganografia que oculta em outro objeto (imagens, vídeos).

A finalidade da criptografia é promover autenticação (tem como característica legitimar a transmissão além de autenticar as entidades que fazem parte da comunicação), integridade (garantir que os dados recebidos pelo receptor sejam o mesmo transmitido pelo emissor), confidencialidade (manter o conteúdo seguro e disponível apenas para o devido receptor), uma comunicação segura, garantindo dessa forma a privacidade da informação além de promover a integridade dos dados [MISAGHI, 2001].

O ideal seria utilizar os dois métodos (esteganografia e criptografia) em conjunto, onde primeiramente, criptografaria uma mensagem antes de aplicar a técnica esteganográfica, aumentando assim a segurança da informação [KOBUSZEWSKI, 2004].

Capítulo 3

Terminologia, Técnicas Esteganográficas, Aplicações da Esteganografia, Esteganálise e seus Tipos de Ataques

3.1 Terminologia

Na esteganografia, existem alguns termos próprios para descrever a ocultação de uma informação. Essa terminologia foi definida no Information Hiding Workshop1 [PETRI, 2004]. Os principais termos são:

- Dado embutido (*embedded data*) - refere-se à informação que se quer ocultar. No caso desse projeto é um arquivo de texto;
- Recipiente ou objeto-cobertura (*container* ou *cover-object*) - refere-se ao objeto que irá ocultar a informação. Podendo ser um áudio (*cover-audio*), vídeo (*cover-video*), texto (*cover-text*), imagem (*cover-image*). No caso desse projeto é um áudio ou imagem digital;
- Estego-objeto (*stego-object*) - depois da inserção da mensagem no recipiente dá origem a um novo arquivo, esse arquivo possui o nome de estego-objeto;
- Estego-chave (*stego-key*) é responsável por controlar a inserção e a detecção do dado embutido.

A Figura 3.1 mostra a aplicação da terminologia explicada anteriormente.

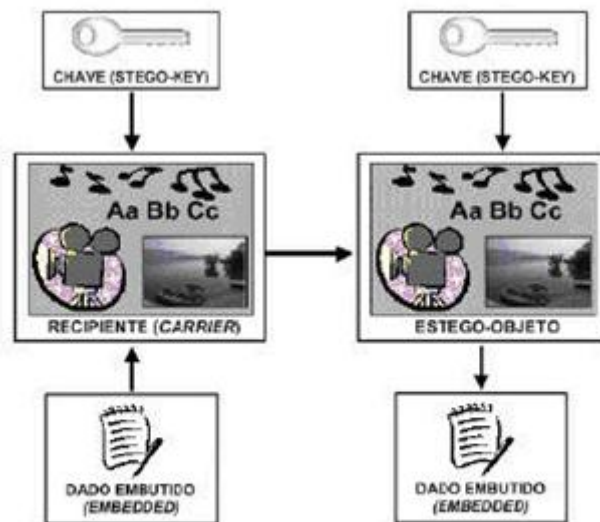


Figura 3.1: Terminologia

Fonte: Coelho e Bento, Laura C. M.; Ricardo J. Ferramentas de Esteganografia e seu uso na Infowar. Evidência Digital Magazine, 2004, p.10.

3.2 Técnicas Esteganográficas

Existem diversas técnicas que propiciam a utilização da esteganografia em diversos ambientes. Entre as abordagens mais conhecidas estão as técnicas que manuseiam com imagens (técnicas de inserção no bit menos significativo - LSB, técnica de filtragem e mascaramento, algoritmos e transformações além do espalhamento por espectro), áudio e vídeo.

3.2.1 Técnicas de inserção no bit menos significativo (LSB)

Essa técnica consiste na substituição do último bit menos significativos de uma imagem digital ou de um áudio por uma informação. Dessa forma, pode-se retirar ou alterar o último bit menos significativo de um byte que não irá causar uma perturbação significativa, ou seja, o ser humano não irá perceber que alguma informação foi alterada.

Como exemplo dessa técnica, pode-se supor que uma amostra de texto seja inserida em uma imagem ou em um áudio tomando o ultimo número em cada bloco (que está em **negrito**) como a posição que será inserida o texto [PETRI, 2004].

1011100**1** - 0111010**1** - 0001101**0** - 0011000**1** - 1110100**0**

Agora o texto, cujo código binário é 10010 será inserido em uma determinada mídia.

10111001 - 01110100 - 00011010 - 00110001 - 11101000

Pode-se perceber que dos dados inseridos apenas foi modificado o segundo bloco binário, fazendo com que a alteração fique de difícil percepção humana.

Esse tipo de técnica é considerado uma das mais difíceis de serem decodificadas já que a inserção das informações no bit menos significativo pode ser feita de forma aleatória [POPA, 1998].

LSB em Imagem

Essa técnica apenas terá resultado satisfatório se o ser humano puder visualizar uma imagem digital de forma limpa, sem distorções ou mudança de cor.

Cada pixel de uma imagem corresponde a 24 bits e cada pixel é formado por três bytes. Onde um primeiro byte corresponde ao canal vermelho, um segundo byte corresponde o canal verde, um terceiro byte para o canal azul [JULIO, 2007].

Dessa forma, pode-se retirar ou alterar o último bit menos significativo de um byte que não irá causar uma perturbação significativa para os olhos, ou seja, os olhos humanos não irão perceber as pequenas alterações que uma imagem pode sofrer.

As Figuras 3.2 e 3.3 podem melhor comprovar a afirmação anterior onde a Figura 3.2 é uma imagem digital comum (sem aplicação de técnicas de esteganografia) e a Figura 3.3 é uma imagem que possui uma mensagem inserida.



Figura 3.2: Figura antes de aplicar a técnica LSB



Figura 3.3: Figura depois de aplicar a técnica LSB

LSB em Áudio

Essa técnica apenas terá resultado satisfatório se o ser humano puder compreender o arquivo de áudio de forma aceitável (sem ruídos). O sistema auditivo humano (SAH) suporta uma enorme variação de frequências (entre 16 e 20.000 Hz - Hertz), ou seja, os seres humanos conseguem distinguir entre grandes intervalos de sons.

O SAH não consegue diferenciar entre todos os sons que recebem. Isso é, quando o ouvido recebe um som forte em uma determinada frequência, a frequência de menor significância (menor amplitude) não é identificada pelo ouvido. O som mais alto tende a prevalecer sobre o mais baixo [EDUARDO P. JULIO, 2007].

Para a transmissão do arquivo de áudio não existe uma taxa fixa de transmissão de dados. Essa pode variar dependendo da amostragem. Segundo [EDUARDO P. JULIO, 2007] a taxa de transmissão pode variar devido a sua amostragem e o tipo de som, sendo que a variação pode variar de 2 bits por segundo (bps) até 128bps.

3.2.2 Técnicas baseadas em Algoritmos e Transformações

Essa técnica utiliza algoritmos e transformações para melhor ocultar uma informação como transformada de Fourier, transformada direta dos cossenos (que são as mesmas utilizadas na compressão do JPEG e MPEG) e transformada Z [GONZALEZ e WOODS, 1998]. Esse processo acontece da seguinte forma: A imagem é dividida em blocos de 8x8 pixels, sendo que em cada bloco é detectado o que existe de excessivo ou o que possui menor importância. Nesses locais são acrescentados os bits referentes ao dado embutido.

São consideradas como sendo as técnicas mais elaboradas, porém, mesmo assim são sujeitas os ataques de esteganálise [JULIO, 2007]. Esse método difunde melhor os bits do dado embutido em toda região da imagem. Dessa forma, as imagens ficam mais protegidas já que são mais difíceis de serem detectadas e compreendidas.

3.2.3 Técnicas de Espalhamento de Espectro

Nesse tipo de técnica, os dados ocultos são distribuídos em toda imagem. Faz uso de uma estego-chave para identificar aleatoriamente os canais de frequência. Os dados que fazem parte da imagem de cobertura são visualizados como interferência em um framework de comunicação. De acordo com [JULIO, 2007] *”os dados embutidos são primeiramente modulados com pseudo ruídos e então a energia é espalhada sobre uma faixa de frequência larga, alcançando somente um nível muito baixo de força de inclusão”*.

3.2.4 Técnicas de Esteganografia em Vídeos

Essa técnica faz uso do vídeo para fazer a ocultação de dados. Para que isso seja possível é utilizado o método das transformada discreta de cosseno (DCT). Essa técnica é utilizada na compressão de dados de imagens e vídeos.

Com isso, pode-se perceber que o método de ocultar dados em um vídeo é muito parecido com o das imagens, a diferença se dá no local de ocultamento dos dados. Nos

vídeos os dados são ocultados entre os frames enquanto nas imagens os dados são ocultados nos pixels.

Dá mesma forma que ocorre nas imagens, quanto maior o número de dados ocultos em um vídeo, maior será a chance de detecção do uso de técnicas esteganográficas.

3.3 Aplicações

Essa sessão tem o intuito de abordar as formas como estão sendo aplicadas a esteganografia atualmente e suas pesquisas atualmente.

3.3.1 Aplicações Militares

Durante uma simulação militar ou até mesmo diante de uma guerra, a comunicação deve ser mantida secretamente para evitar que o inimigo obtenha informações sigilosas referentes à situação. Portanto, o ocultamento da informação (a esteganografia) é utilizado como mais uma forma de aumentar a segurança e com isso, dificultar a detecção da informação.

3.3.2 Direitos Autorais

Esse tipo de aplicação tenta defender os direitos de um autor sobre a sua referente obra. Para isso o autor deve assinar a sua obra para que sejam respeitados os seus direitos autorais. De acordo com [GOIS, 2003], não basta apenas anexar um arquivo especificando o autor da obra, já que, pode ser facilmente retirado ou substituído. Devem-se utilizar técnicas esteganográficas para ocultar a 'assinatura' do autor em sua obra para que se possa descobrir se sua obra esta sendo utilizada ou não com a autorização do autor.

3.3.3 Aplicações Escondendo o Remetente

Há ocasiões, em que uma mensagem é enviada sem que o receptor descubra quem foi o remetente. Normalmente esse tipo de mensagem é utilizada por criminosos que não desejam que sua identidade se torne pública caso a mensagem tenha sido rastreada porém existe aplicações legais como votações online e registros médicos [EDUARDO P. JULIO, 2007].

3.3.4 Aplicações Médicas

Outro local de aplicação da esteganografia é na área industrial voltada para imagens médicas. A forma de comunicação do médico com o exame feito por um paciente é chamada de DICOM (*digital imaging and communications in medicine*). O DICOM

separa o exame em duas partes. A primeira seria uma imagem (por exemplo, um raio-X) e a segunda parte as informações do paciente (nome, data, etc.).

Porém pode acontecer da informação ou da imagem se perder. Portanto o ideal seria se as informações estivessem ocultas na imagem, ou seja, juntas em um mesmo documento. Apesar das poucas pesquisas na área, acredita-se que a aplicação de tal técnica não prejudique o diagnóstico do médico [EDDIE B. L. FILHO, 2005].

3.3.5 Certificação e Controle de Acesso

Esse tipo de técnica é mais uma forma utilizada para impedir a cópia de algum documento original, ou seja, é uma forma de identificar se um documento é uma fraude ou não. Essa técnica é mais utilizada em carteiras de identidade e transporte. A foto do documento possui informações que a identificar o documento. Normalmente, documentos falsificados não possuem essa informação ou a informação está equivocada.

3.3.6 Economia de Banda

Também pode ser utilizada para aproveitar a banda de comunicação, ou seja, além de utilizar um mesmo meio para transferir duas informações simultaneamente, estaria diminuindo o tempo de transferência desses arquivos. Um exemplo seria a transmissão de dados através da TV. Com isso, usuário poderia obter algum tipo de informação sobre o produto apenas com o clique do controle. Essa forma não comprometeria a qualidade da imagem fornecida pela emissora.

Outro exemplo dessa aplicação seria a utilização de legendas nos filmes. As legendas, poderiam vir embutidas nas fitas VCR, em DVDs e até mesmo nas imagens recebidas pelos televisores. Dessa forma, em um mesmo arquivo (nesse caso, um mesmo filme) poderia possuir outros arquivos (legendas) sem ocupar mais espaço para isso. Apenas seria necessário que os aparelhos correspondentes, pudessem decodificar esse tipo de conteúdo.

3.3.7 Autenticação

A utilização da esteganografia em alguns objetos pode servir para indicar a sua autenticação. As marcas digitais poderiam ser utilizadas para indicar se uma mídia sofreu algum tipo de alteração ou se é realmente original. Um exemplo disso seria a utilização dessa técnica em câmeras digitais. Com isso, as imagens poderiam ser comprovadas como originais podendo assim ser aceitas como provas, já que hoje, por não poder ser identificado sua autenticidade, essas não são aceitas como prova [GOIS, 2003].

3.3.8 Ética

Um importante tema relacionado com a ocultação de informação que gera discussão é a ética já que esse tipo de técnica propicia a privacidade e de certa forma, algum tipo de segurança. Dessa maneira, essa técnica pode cair em mãos erradas e acabar sendo utilizada de forma irregular (por grupos criminosos) como forma de chantagens, difamação ou até mesmo seqüestro.

3.4 Esteganálise

A tentativa de encontrar mensagens ocultas em algum tipo de objeto e torná-las inúteis é chamada de esteganoanálise. Ao utilizar algum tipo de objeto com o propósito de ocultar uma informação, faz com que esse objeto sofra algum tipo de modificação. De certa forma, essa modificação pode ser percebida através da comparação entre objetos com características semelhantes. Assim, o objetivo que possui esteganografia, se torna ineficiente.

Com isso, pode-se perceber que as técnicas de esteganografia podem detectar mensagens ocultas em um objeto de cobertura. O processo de "mascarar" uma informação pode ser simples e facilmente identificável ou robusto, fazendo com que a tentativa de identificar e recuperar uma mensagem se torne difícil.

Normalmente, apenas a descoberta da existência de uma mensagem em um objeto esteganográfico é o suficiente para um suposto agressor [GOIS, 2003]. Já que, as mensagens são delicadas e com isso, podem ser facilmente destruída, ou até mesmo, pode ocorrer a destruição do objeto de cobertura.

Devido ao fato de terroristas utilizarem métodos de esteganografia para comunicarem ataques, estão sendo desenvolvidos novas técnicas de esteganálise para a identificação de tal informação [KOBUSZEWSKI, 2004].

Os mais eficientes algoritmos de esteganoanálise não são capazes de identificar a informação presente já que, vários algoritmos de ocultamento de informação utilizam geradores aleatórios dificultando assim a sua identificação. Mesmo assim, esses algoritmos são capazes de detectar a sua presença.

Ainda não existe uma forma de garantir que um objeto esteganográfico possa burlar a esteganoanálise.

3.5 Tipos de Ataques

Há vários tipos de ataques para identificar a existência de esteganografia. Dentre os vários tipos existentes, destacam-se: Ataques Aurais, Ataques Estruturais e Ataques

Estatísticos [PETRI, 2004].

3.5.1 Ataques Aurais

São também conhecidos como ataques visuais. Segundo [ROCHA, 2003], "*Alguns ataques retiram as partes significativas da imagem como um meio de facilitar aos olhos humanos a busca por anomalias na imagem*". Esse tipo de ataque facilita a busca de anomalia nas imagens pelos olhos humanos já que, ao aplicar os métodos de esteganografia retiram partes significativas da imagem.

Um meio de identificar se existe uma mensagem em um objeto de cobertura consiste na retirada dos sete primeiros bits de um pixel, deixando apenas o bit menos significativo. Sabendo que as cores podem sofrer variações de 256 formas diferentes (0 até 255 em binário), já que cada pixel é formado por três bytes, o último bit forma uma imagem visível se comparada com a original. Com isso, pode-se comparar as imagens e perceber se existem possíveis alterações [GOIS, 2003].

3.5.2 Ataques Estruturais

Quando uma mensagem é inserida, a estrutura do arquivo de dados sofre algum tipo de alteração. Por muitas vezes, essa alteração, não é percebida pelo olho humano. Dessa forma, sistemas capazes de analisar padrões estruturais podem descobrir a existência de mensagens ocultas [EDUARDO P. JULIO, 2007].

Um método utilizado em ataques estruturais seria a compactação do arquivo já que dessa maneira, elimina os bits semelhantes, ou seja, a redundância e com isso altera a estrutura do arquivo, fazendo com que se torne transparente o que estava oculto. [GOIS, 2003].

3.5.3 Ataques Estatísticos

Segundo [ROCHA, 2003], "os padrões dos pixels e seus bits menos significativos frequentemente revelam a existência de uma mensagem secreta nos perfis estatísticos". A estatística possui o objetivo de classificar a probabilidade de um fenômeno ocorrer ou não [GOIS, 2003].

Esse ataque é utilizado para identificar se algum dado (ou objeto de cobertura) possui algum tipo de mensagem. O dado que possui algum tipo de mensagem, não possui o perfil esperado, e quando submetido por ataques estatísticos, na grande parte das vezes, os dados analisados são mais aleatórios do que aqueles que apresentam algum tipo de mensagem oculta.

Capítulo 4

Implementação do Programa

Este capítulo possui como objetivo apresentar as especificações e implementação do protótipo de esteganografia utilizando a técnica LSB proposto inicialmente.

4.1 Principais Requisitos do Modelo

O protótipo desenvolvido nesse projeto possui como finalidade utilizar a técnica de esteganografia LSB para ocultar informações no formato texto em uma imagem digital e em um áudio.

Esse software possui dois tipos de usuários bem distintos, o emissor e o receptor. O emissor possui como objetivo inserir um dado (uma mensagem no formato texto) em um recipiente (imagem digital ou áudio). Para que a inserção aconteça de maneira eficiente e correta, é necessário que se tenha uma imagem digital no formato *bitmap* (*PNG*) ou um áudio no formato *WAV*.

Bitmap são imagens de rastreio ou pintura, essas imagens são formadas por pontos denominados pixels, cada ponto é pintado com uma cor diferente fazendo com que dessa forma o olho humano visualize uma imagem [LOPES, 2004].

O único formato de *bitmap* aqui utilizado é o *PNG*, pois ao aplicar a técnica LSB em outros formatos de arquivo como *JPG* e *GIF*, por exemplo, o resultado não é satisfatório, já que ao inserir uma mensagem no bit menos significativo não é possível obter a mensagem inserida. Isso se dá pelo fato desses formatos de arquivos sofrerem compressão por perda, dessa forma as imagens sofrem transformações que fazem com que elas percam algum tipo de informação [LOPES, 2004].

Por esse motivo, as imagens do tipo *JPG* e *GIF* ocupam um espaço menor no disco rígido do que as *PNG*. Porém, as imagens que possuem como formato *PNG* não proporcionam perda de qualidade e dessa forma, possuem uma melhor fidelidade à imagem original [LOPES, 2004].

WAV possui áudio no formato pulso, também conhecido como PCM (*pulse code modulation*). Esse tipo de formato utiliza uma técnica de armazenamento de áudio não-comprimido, ou seja, não permite perda da informação [EVERSON CARLOS MAUDA, 2008]. Isso explica o porquê da grande parte dos outros formatos serem menores e consequentemente ocuparem menor espaço que o *WAV*.

Para que se possa realizar a inserção é necessário que se utilize uma estego-chave, que é responsável por controlar o segredo relacionado com a inserção e a detecção do dado embutido.

Após a inserção de uma informação pelo emitente, é gerado como produto um novo objeto esteganografado, ou seja, um estego-objeto.

O receptor possui como objetivo retirar a mensagem de um estego-objeto (da mesma forma que o emitente é necessário que se tenha uma imagem digital no formato *PNG* ou um áudio no formato *WAV*). Dessa forma será obtida como produto a mensagem o dado embutido no recipiente.

Com a utilização desse software é possível realizar uma comunicação segura, seja esta feita por uma rede de comunicação ou não, entre duas ou mais pessoas. A imagem ou o áudio pode ser enviado via e-mail pelo emitente para o receptor, e mesmo que esse estego-objeto seja desviado ou até mesmo visualizado por uma pessoa indesejada, não será percebido ou identificado.

4.2 Especificação do Protótipo

Esse tópico possui como objetivo a análise do projeto através da utilização do diagrama de caso de uso, diagrama de classe, diagrama de sequência e diagrama de Atividade.

4.2.1 Diagrama de caso de uso

Esse projeto possui apenas quatro tipos de caso de uso conforme a Figura 4.1: 'insere dado imagem', 'insere dado áudio', 'identifica mensagem imagem' e o 'identifica mensagem áudio'.

'Insere dado imagem' é designado por inserir o dado em uma imagem e 'insere dado áudio' é o designado por inserir o dado em um áudio e ambos serão identificados pelo emissor.

O caso de uso 'identifica mensagem imagem' é designado por retirar o dado embutido em uma imagem e o 'identifica mensagem áudio' é designado por retirar o dado embutido em um áudio e ambos serão identificados pelo receptor.

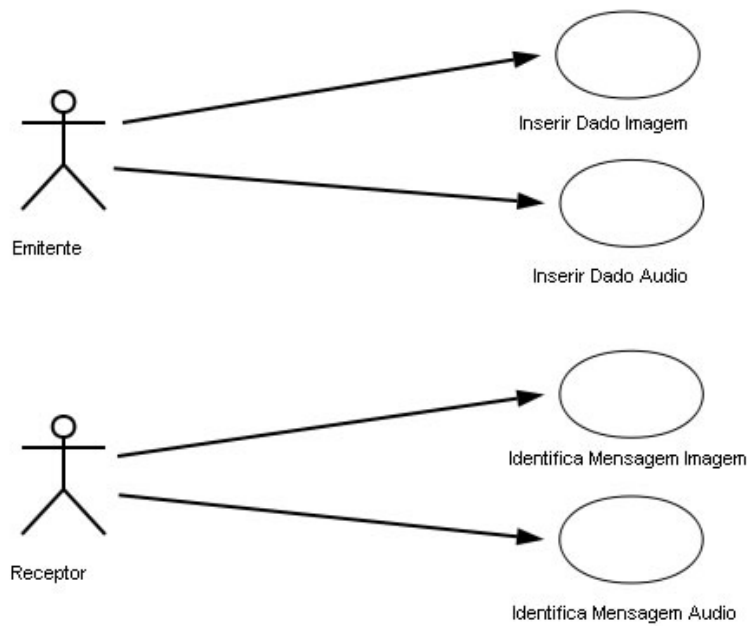


Figura 4.1: Diagrama de Caso de Uso

4.2.2 Diagrama de Classe

Esse tipo de diagrama possui a finalidade de ilustrar o comportamento de uma classe. Existem cinco classes envolvidas na construção do projeto. São elas, *JanelaPrincipal*, *InsercaoDaMensagem_Imagem*, *InsercaoDaMensagem_Audio*, *DeteccaoDaMensagem_Imagem* e *DeteccaoDaMensagem_Audio*.

Principal: Como o próprio nome já diz, é a principal classe do projeto. É a classe que fica responsável pela inicialização do sistema é nessa classe que é construído a interface gráfica e com isso todo o elo entre o sistema. Dessa forma, essa classe é responsável por carregar as imagens e os áudios (tanto de inserção como de remoção da mensagem), pelas informações dos dados das imagens e dos áudios carregadas, criação de um novo diretório, caso seja necessário, para comportar um estego-objeto, ou seja, um objeto esteganografado. O diagrama dessa classe pode ser visualizado na Figura 4.2.



Figura 4.2: Diagrama da Classe Principal

InsercaoDaMensagem_Imagem: Essa classe é responsável pela identificação do padrão RGB de cada pixel e dessa forma, inserir a mensagem digitada pelo usuário no recipiente (imagem digital). Além disso, também é responsável pela criação de uma cópia da imagem selecionada para um diretório pré-estabelecido. O diagrama dessa classe pode ser visualizado na Figura 4.3.

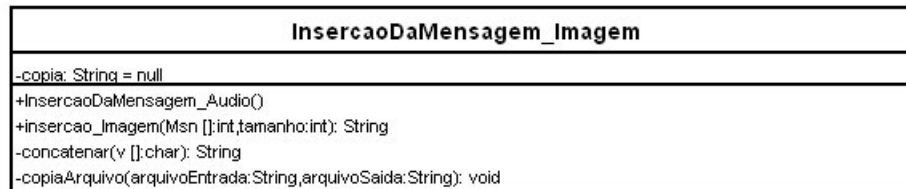


Figura 4.3: Diagrama da Classe InsercaoDaMensagem_Imagem

DeteccaoDaMensagem_Imagem: Essa classe é responsável por identificar se existe ou não uma mensagem esteganografada em uma determinada imagem além de possibilitar a remoção do dado embutido (mensagem oculta) através da manipulação do padrão RGB de cada pixel. O diagrama dessa classe pode ser visualizado na Figura 4.4.

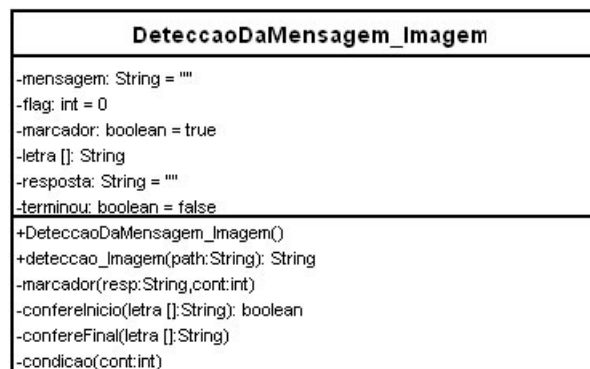


Figura 4.4: Diagrama da Classe DeteccaoDaMensagem_Imagem

InsercaoDaMensagem_Audio: Essa classe é responsável por percorrer o arquivo de áudio, separar o cabeçalho do setor de dados e inserir a mensagem digitada pelo usuário no recipiente (áudio). Da mesma forma que ocorre com a imagem, é criada uma cópia do áudio original para um diretório pré-estabelecido. O diagrama dessa classe pode ser visualizado na Figura 4.5.



Figura 4.5: Diagrama da Classe InsercaoDaMensagem_Audio

DeteccaoDaMensagem_Audio: Essa classe é responsável por identificar se existe ou não uma mensagem esteganografada em um determinado áudio além de possibilitar a remoção do dado embutido (mensagem oculta). Isso é possível analisando o setor de dados do áudio indicado pelo usuário. O diagrama dessa classe pode ser visualizado na Figura 4.6.

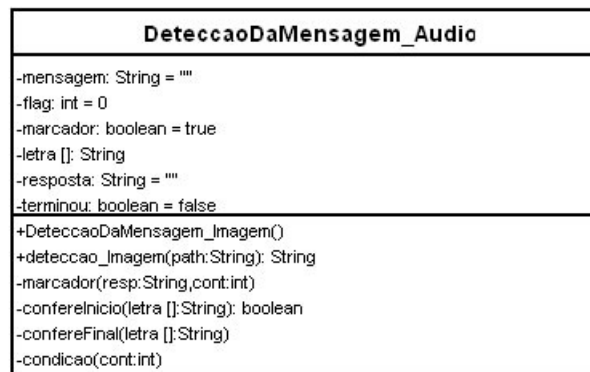


Figura 4.6: Diagrama da Classe DeteccaoDaMensagem_Audio

A Figura 4.7 apresenta o diagrama de classe completo, mostrando todas as interações, demonstrando assim o funcionamento em sua totalidade.

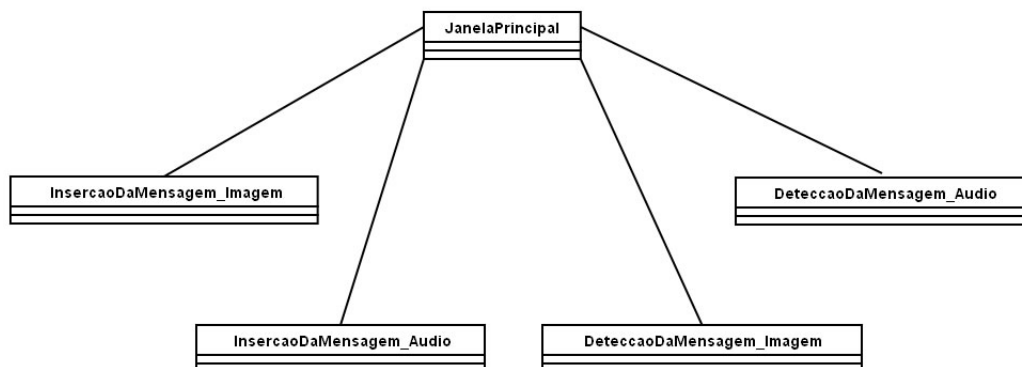


Figura 4.7: Diagrama de Classe

4.2.3 Diagrama de Seqüência

Diagrama de seqüência é a forma utilizada para descrever as interações entre grupos de objetos de uma classe que comunicam de maneira harmoniosa durante um intervalo de tempo. A sua criação toma por base o diagrama de caso de uso já que, é necessário, inicialmente, identificar a seqüência de eventos do sistema para depois definir o comportamento de cada evento individualmente [PFLEEGER, 2003]. São principalmente utilizados para indicar os processos e os possíveis acionamentos simultâneos além de apresentar a seqüência geral do fluxo de controle.

Nas sessões a seguir serão detalhados os diagramas de seqüências possíveis do sistema.

Inserção da Mensagem em Imagem

Este diagrama é utilizado no momento pelo qual o usuário insere uma mensagem em uma imagem, considerando sempre que essa imagem esteja no formato Bitmap. Para isso é instanciado um objeto da classe `InsercaoDaMensagem_Imagem`. Após o objeto dessa classe ser instanciado, é utilizando o método "`copiaArquivo()`" que faz uma cópia do arquivo original para uma pasta criada pelo programa denominada 'Esteganografia'.

Logo após, o método "`inserção_Imagem()`" recebe a informação na forma de vetor que será inserida na cópia da imagem original na forma de bits. Para que essa operação seja realizada com sucesso, é utilizado o método "`concatenar()`" que como o próprio nome sugere, tem a finalidade de retornar e concatenar o mapa de bits situado no vetor com mensagem que será inserida. Essas operações podem ser visualizadas se for analisado o diagrama de seqüência da Figura 4.8.

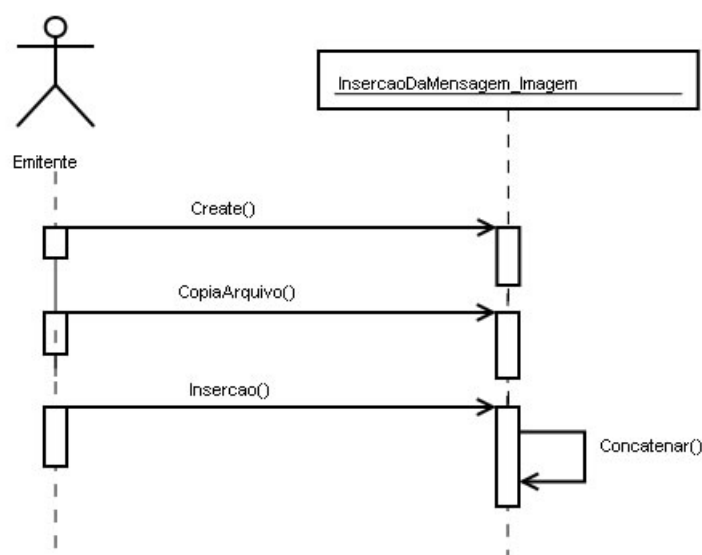


Figura 4.8: Diagrama de Seqüência: Inserção da Mensagem em Imagem

Remoção da Mensagem em Imagem

Esse diagrama é executado com a finalidade de verificar se existe uma mensagem em uma determinada imagem e caso exista, retirá-la com o propósito de fornecer ao usuário o seu conteúdo. É instanciado um objeto da classe `DeteccaoDaMensagem_Imagem` e utilizando em seguida o método `"detecção_Imagem()"` que percorre toda a imagem para identificar se esta possui alguma mensagem oculta.

Isso é possível graças ao método `"marcador()"` que faz uso dos métodos `"confereInicio()"` e `"confereFinal()"` com a finalidade de verificar se existe ou não uma mensagem oculta. Dessa forma, o programa informa para o usuário a condição da imagem (respondendo positivamente se existir alguma mensagem e negativamente caso não exista uma mensagem naquele recipiente), caso exista alguma mensagem oculta, esta será retornada para a classe Principal onde será apresentado para o usuário. Esse processo será mais bem abordado na sessão 6.3.3 referente ao Funcionamento do Programa.

Essas operações podem ser melhor visualizadas se for analisado o diagrama de sequência da Figura 4.9.

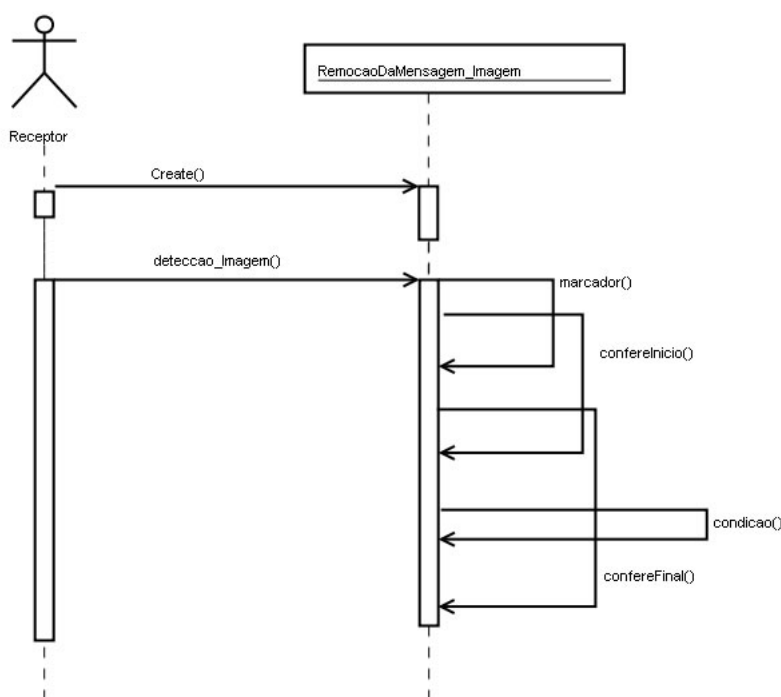


Figura 4.9: Diagrama de Sequência: Remoção da Mensagem em Imagem

Inserção da Mensagem em Áudio

Este diagrama é utilizado no momento pelo qual o usuário insere uma mensagem em um áudio, considerando sempre que esse áudio esteja no formato *WAV*. Para isso é

instanciado um objeto da classe `InsercaoDaMensagem_Audio` já que é nesta classe que a mensagem é inserida no áudio. Após a instância desse objeto é chamado o método `"inserção_Audio()"`.

Esse método recebe um vetor contendo a mensagem que será inserida no áudio na forma de bits além do diretório onde o áudio esteganografado será armazenado. Para que essa operação seja realizada com sucesso, é utilizado método `"concatenar"` que como o próprio nome sugere, tem a finalidade de retornar e concatenar o mapa de bits situado no vetor com mensagem que será inserida. Essas operações podem ser mais bem visualizadas se for analisado o diagrama de seqüência da Figura 4.10.

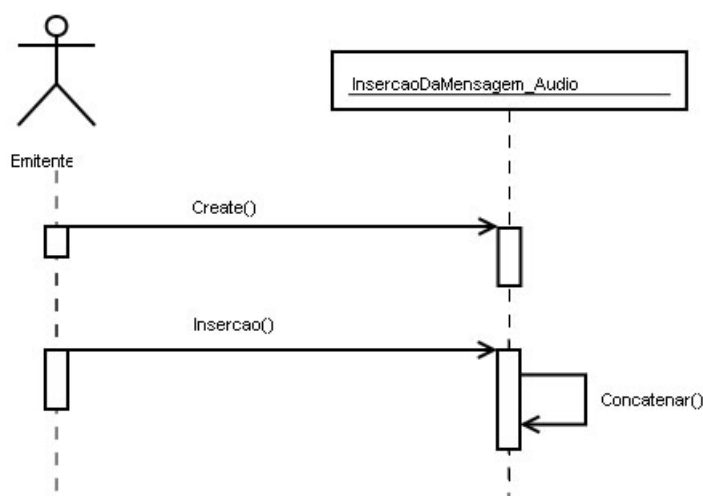


Figura 4.10: Diagrama de Seqüência: Inserção da Mensagem em Áudio

Remoção da Mensagem em Áudio

Esse diagrama possui as mesmas propriedades do diagrama apresentado na Remoção da Mensagem em Imagem, porém, a forma de obtenção da informação oculta se faz de forma diferente. Dessa forma para verificar existência de uma mensagem em um determinado áudio é instanciado um objeto da classe `DeteccaoDaMensagem_Imagem`, utilizando assim o método `"detecção_Audio()"` que percorre o áudio para identificar a existência de mensagem oculta.

Isso é possível graças ao método `"marcador()"` que faz uso dos métodos `"confereInicio()"` e `"confereFinal()"` com a finalidade de verificar se existe ou não uma mensagem oculta. Dessa forma, o programa informa para o usuário a condição do áudio (respondendo positivamente se existir alguma mensagem e negativamente caso não exista uma mensagem naquele recipiente), caso exista alguma mensagem oculta, esta será retornada para a classe Principal onde será apresentado para o usuário. Esse processo será detalhado

na sessão 4.3.3 referente ao Funcionamento do Programa.

Essas operações podem ser mais bem visualizadas se for analisado o diagrama de seqüência da Figura 4.11.

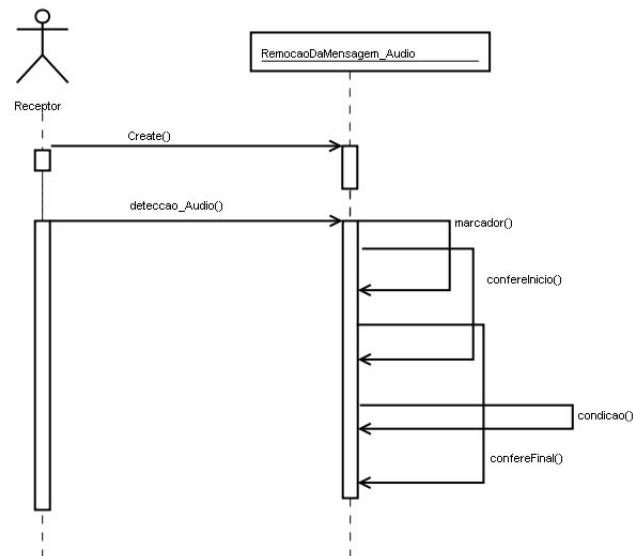


Figura 4.11: Diagrama de Seqüência: Remoção da Mensagem em Áudio

4.2.4 Diagrama de Atividade

O diagrama de atividade serve para demonstrar o fluxo de procedimentos ou atividades que ocorrem em uma determinada classe [PFLEEGER, 2003]. As Figuras 4.12, 4.13, 4.14, 4.15 demonstram o funcionamento desse diagrama em questão. O diagrama de atividade foi dividido em quatro figuras para permitir uma melhor visualização.

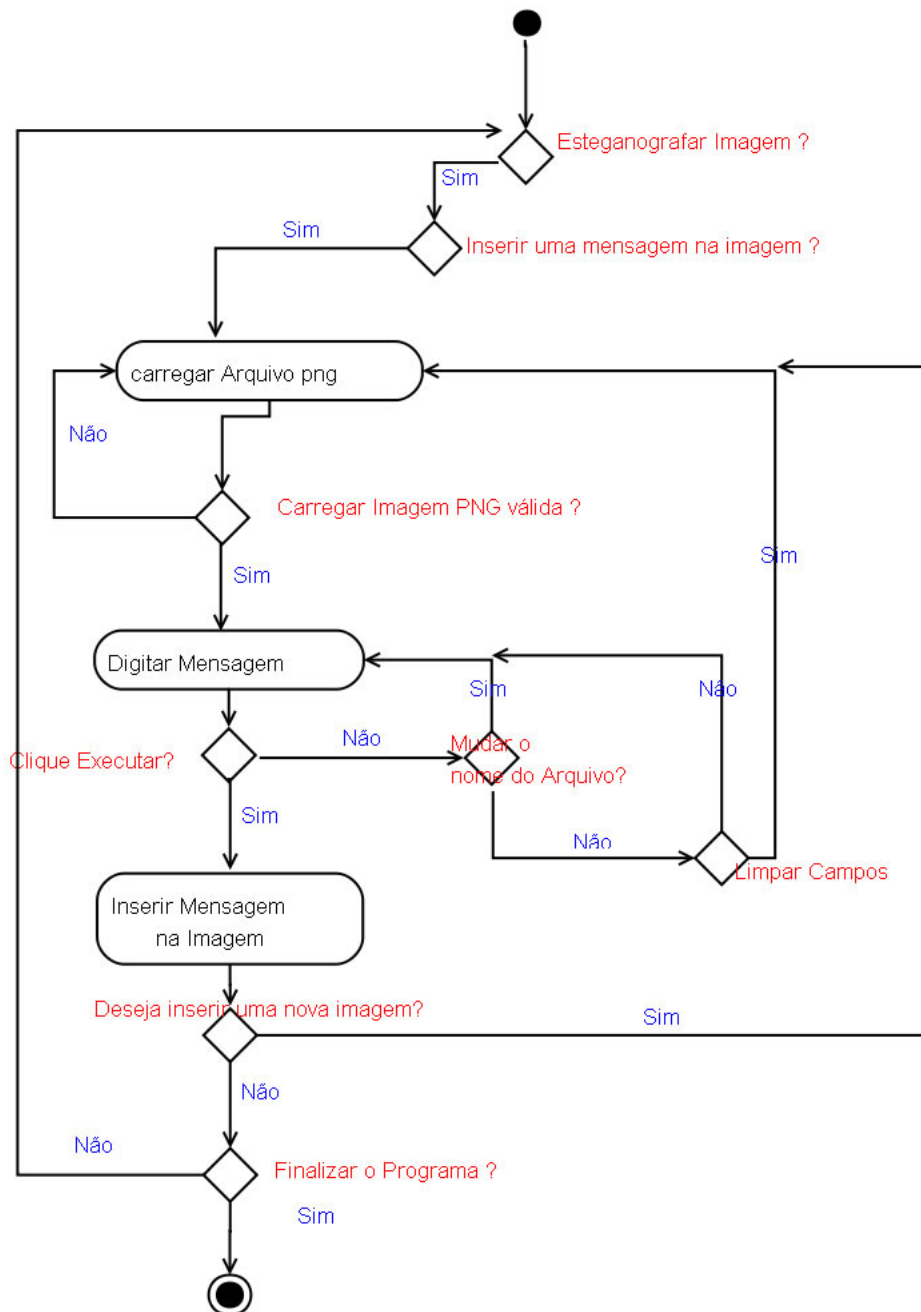


Figura 4.12: Diagrama de Atividade: Inserção da Mensagem em Imagem

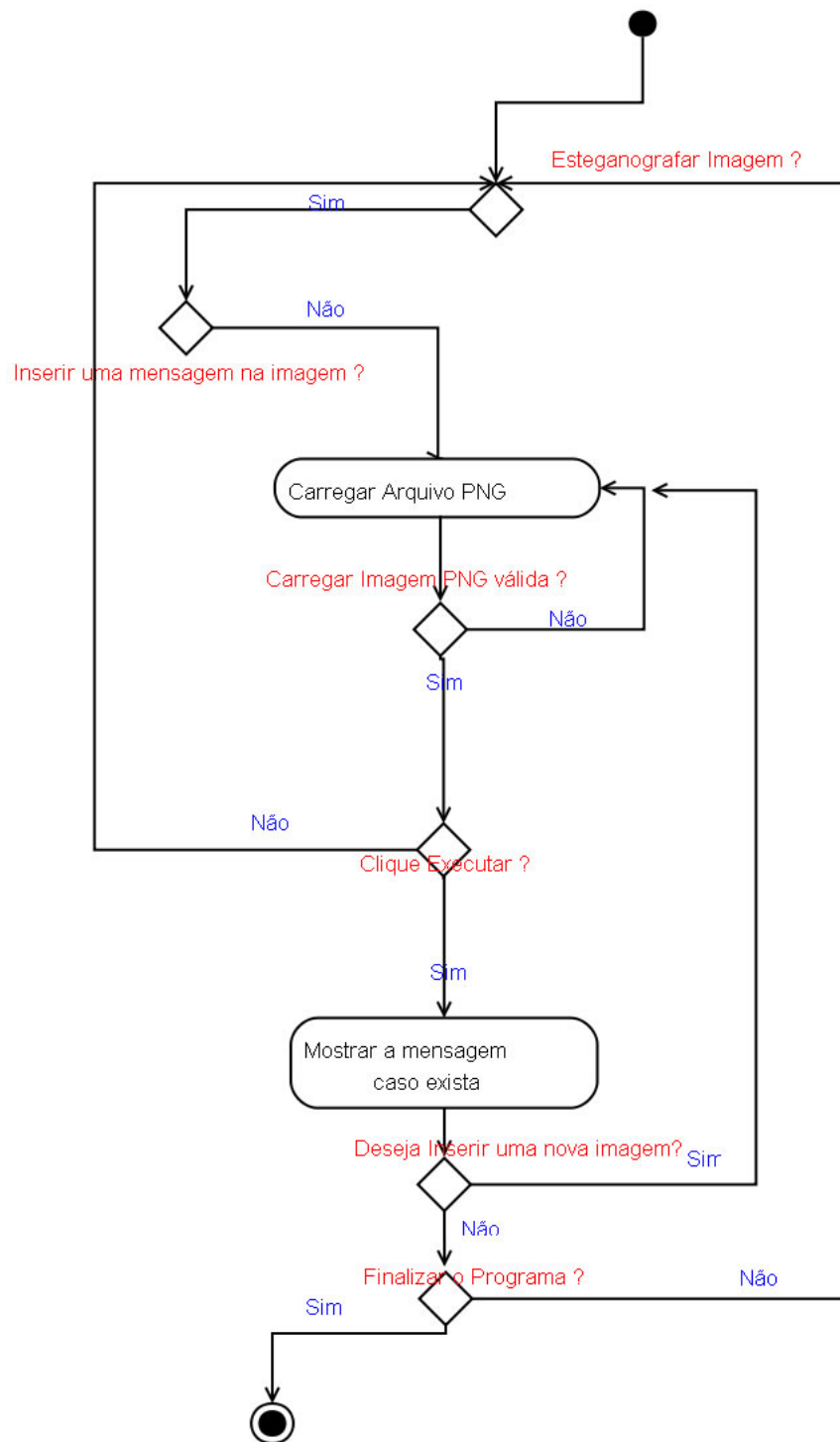


Figura 4.13: Diagrama de Atividade: Remoção da Mensagem em Imagem

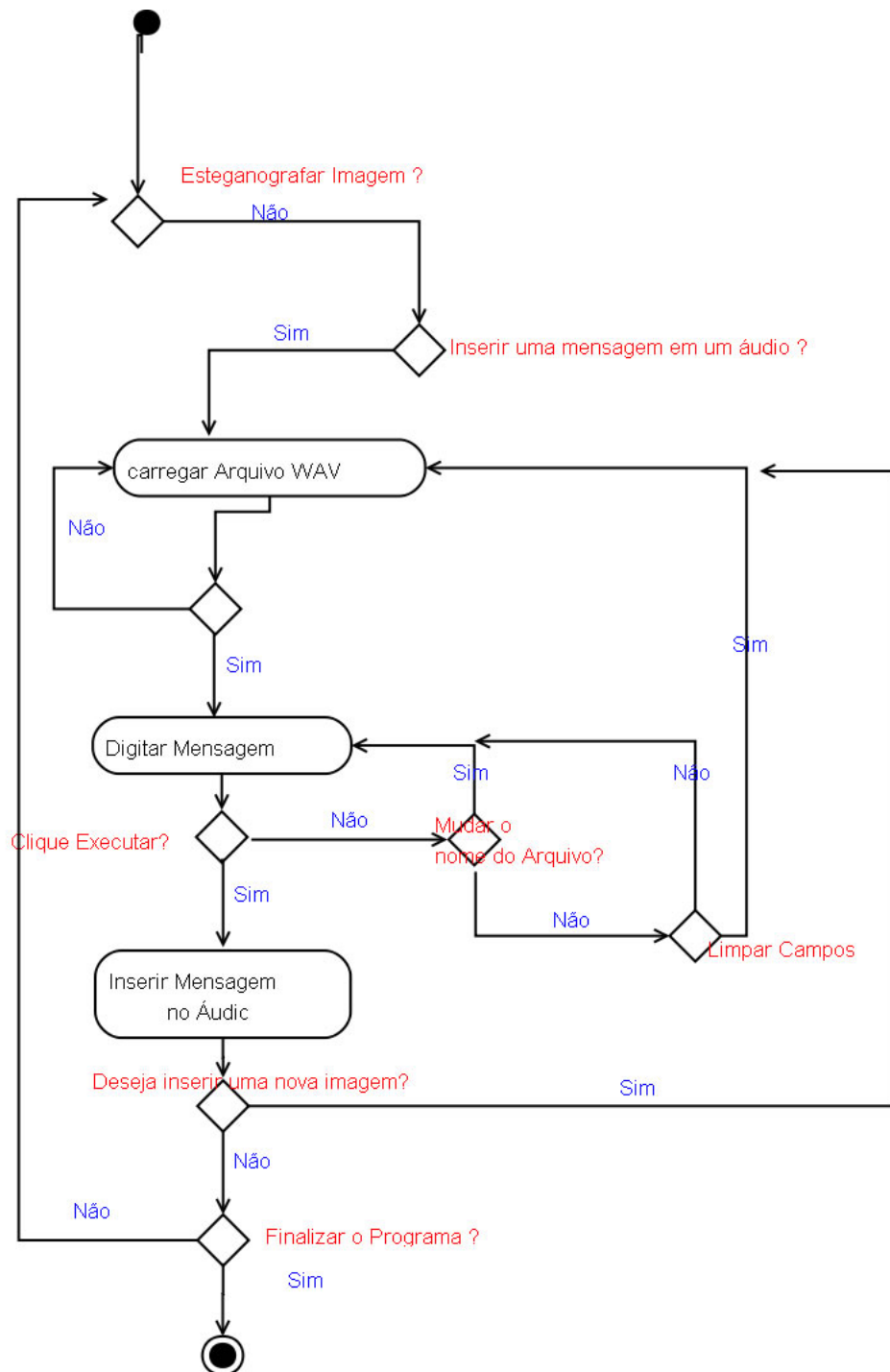


Figura 4.14: Diagrama de Atividade: Inserção da Mensagem em Áudio

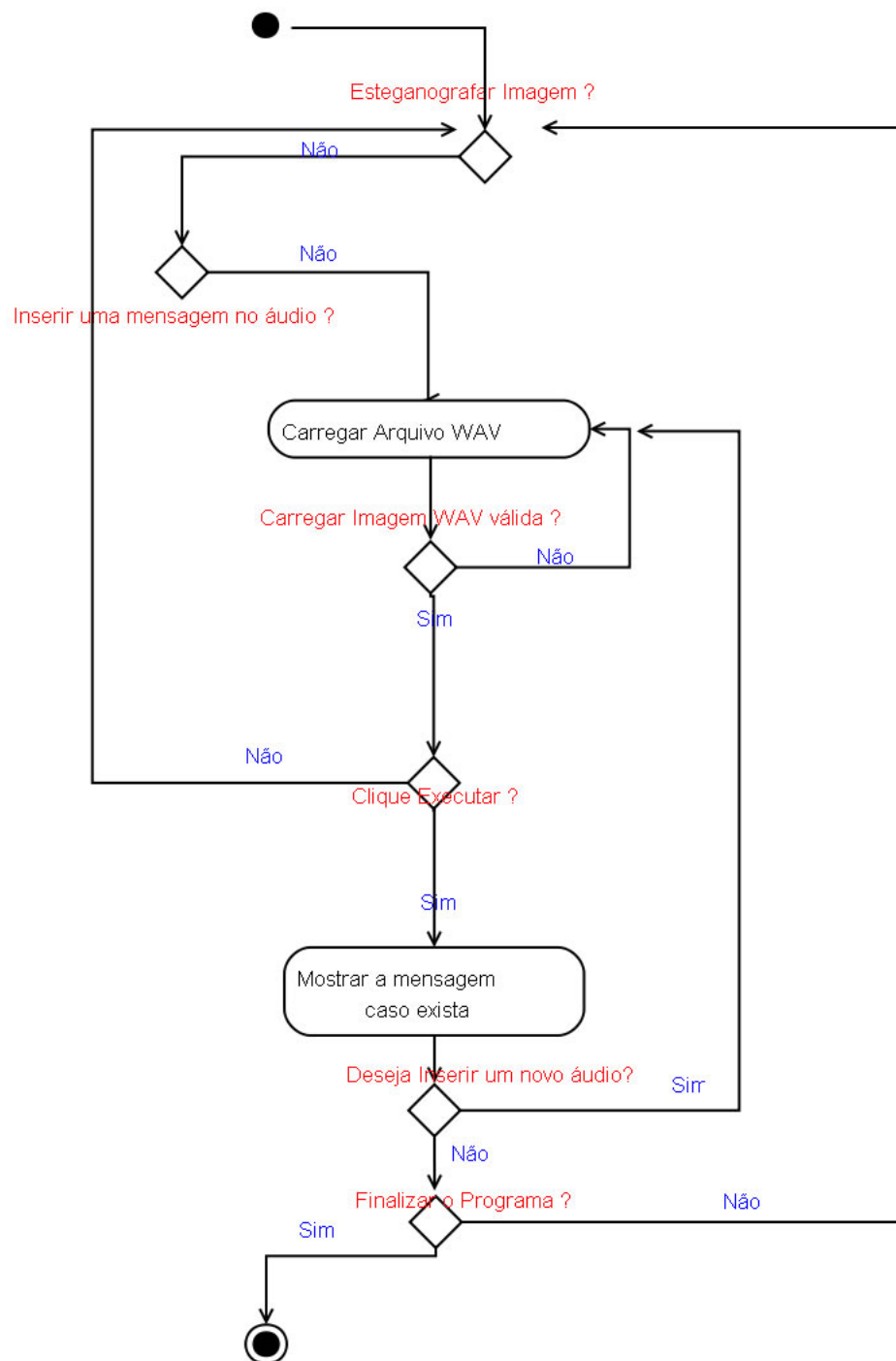


Figura 4.15: Diagrama de Atividade: Remoção da Mensagem em Áudio

4.3 Implementação

Para que o programa fosse desenvolvido com sucesso foi utilizada a linguagem Java, sendo esta uma plataforma gratuita. Para que o programa possa executar é necessário apenas o JVM (Java Virtual Machine) que é gratuita. Para auxiliar na programação foi utilizada NetBeans 6.5.1 que é uma IDE gratuita. Além disso, Java é uma linguagem que funciona em diversos sistemas operacionais, arquiteturas além de ser utilizada tanto em desktop como também na web [SCHIAVONI, 2008].

Java é uma linguagem orientada a objeto, o que facilita a modulação do programa e a sua visualização em um diagrama UML. O que permite manter uma documentação sobre o programa permitindo assim futuras alterações.

Propicia também, uma vasta quantidade e qualidade de APIs além de ser documentada facilitando ao programador a visualização e o entendimento de funcionalidades desconhecidas [SCHIAVONI, 2008].

4.3.1 Interface

Ponto fundamental no funcionamento de um programa é a utilização da interface, já que sem o seu conhecimento prévio não é possível obter êxito em seu funcionamento. A interface do programa em questão possui dois momentos distintos. Primeiramente, o usuário deve decidir se deseja manusear imagem ou áudio. Após essa decisão o usuário deve decidir entre inserir um texto (em uma imagem ou áudio) ou detectar (e retirar) um conteúdo oculto (em uma imagem ou áudio).

A interface foi dividida em abas para melhor dividir o funcionamento do sistema.

As Figuras 4.16 e 4.17 mostram a interface do sistema referente à inserção e detecção de uma mensagem em uma imagem, já as Figuras 4.18 e 4.19 mostram a interface do sistema referente à inserção e detecção de uma mensagem em um áudio.

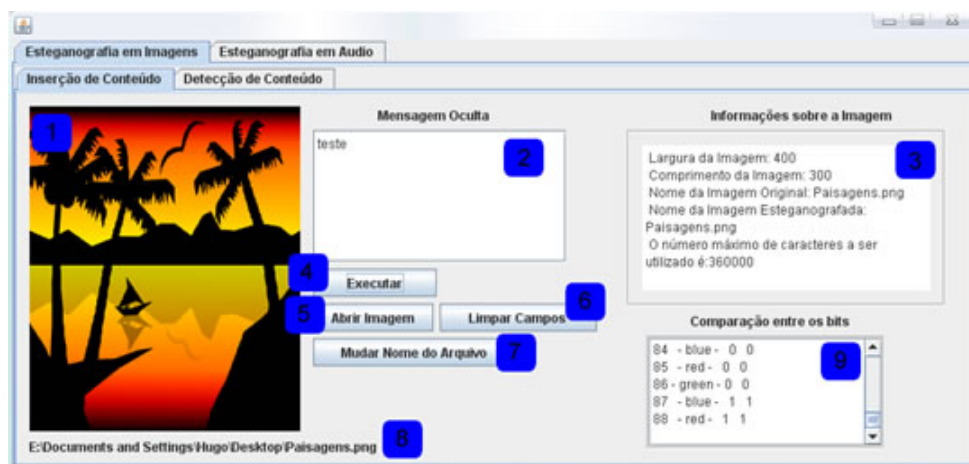


Figura 4.16: Interface: Inserção de Conteúdo em Imagem

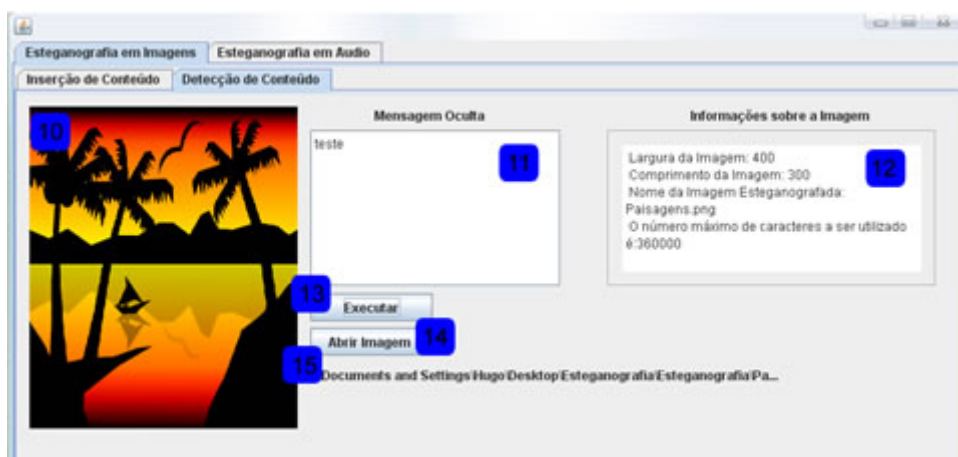


Figura 4.17: Interface: Detecção de Conteúdo em Imagem

Abaixo será explicada a finalidade de cada campo das Figuras 4.16 e 4.17. Os números de um a nove fazem referência à aba 'Inserção de Conteúdo de Imagem' enquanto os números de dez a quinze fazem referência à aba 'Detecção de Conteúdo em Imagem'.

- Os números um e dez refere-se à imagem que foi aberta pelo usuário. As dimensões dessa imagem são: 250 (altura) X 300 (largura). Independentemente do tamanho da imagem original, ela aparecerá com essas dimensões, já que a finalidade é de mostrar para o usuário a imagem aberta por ele;
- O número dois refere-se ao local onde o texto deve ser digitado para que possa ser oculto na imagem;
- Os números três e doze referem-se ao local onde irá aparecer informações referentes à imagem aberta pelo usuário;
- Os números quatro e treze referem-se ao botão 'Executar'. Quando solicitado ele poderá inserir o texto digitado pelo usuário (caso esteja na aba 'Inserção de Conteúdo', número quatro) ou apresentar para o usuário a mensagem oculta (caso esteja na aba 'Detecção de Conteúdo', número treze);
- Os números cinco e quatorze referem-se ao botão 'Abrir Arquivo'. Quando solicitado tem a finalidade de abrir a imagem determinada pelo usuário;
- O número seis refere-se ao botão 'Limpar Campos'. Quando solicitado tem a finalidade de limpar todos os campos da interface;
- O número sete refere-se ao botão 'Mudar Nome do Arquivo'. Quando solicitado, dará a possibilidade de mudar o nome da imagem que será esteganografada;

- Os números oito e quinze referem-se ao local onde irá aparecer o diretório onde a imagem aberta pelo usuário está;
- O número nove é a parte do programa que tem o intuito de mostrar a comparação entre o bit original e o bit modificado;
- O número dez refere-se ao local onde irá aparecer o texto (caso exista) que permanecia oculto na imagem;
- O número onze refere-se ao local onde irá aparecer o texto (caso exista) que permanecia oculto na imagem.

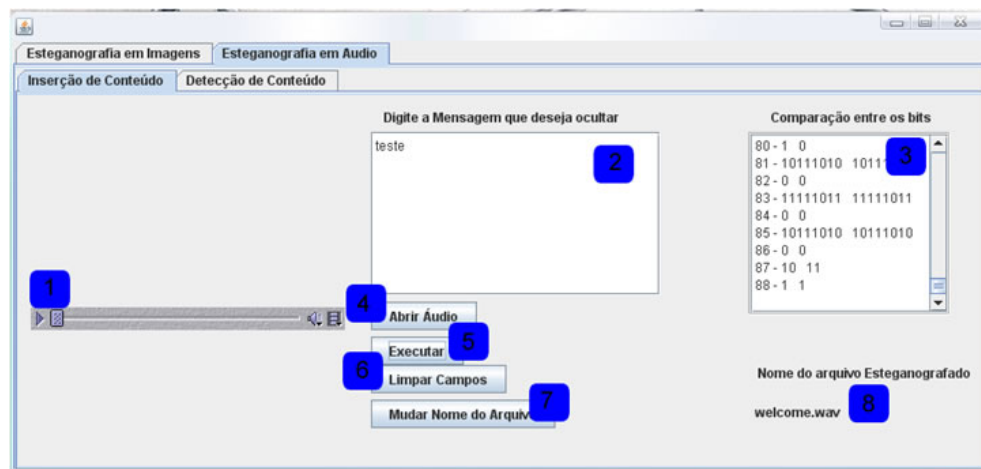


Figura 4.18: Interface: Inserção de Conteúdo em Áudio

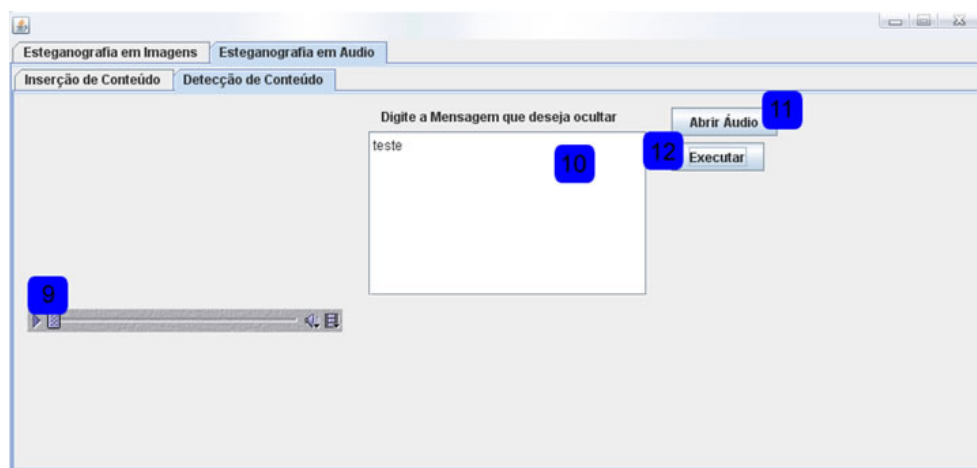


Figura 4.19: Interface: Detecção de Conteúdo em Áudio

Abaixo será explicada a finalidade de cada campo das Figuras 4.18 e 4.19. Os números de um a oito fazem referência à aba 'Inserção de Conteúdo de Áudio' enquanto os números de nove a doze fazem referência a aba 'Detecção de Conteúdo em Áudio'.

- Os números um e nove referem-se à barra responsável por controlar e mostrar a execução do áudio, além de apresentar as informações sobre o áudio;
- O número dois refere-se ao local onde o texto deve ser digitado para que possa ser oculto no áudio;
- O número três é a parte do programa que tem o intuito de mostrar a comparação entre o bit original e o bit modificado.
- Os números quatro e onze referem-se ao botão 'Abrir Áudio'. Quando solicitado tem a finalidade de abrir a imagem determinada pelo usuário;
- Os números cinco e doze referem-se ao botão 'Executar'. Quando solicitado ele poderá inserir o texto digitado pelo usuário (caso esteja na aba 'Inserção de Conteúdo', número cinco) ou apresentar para o usuário a mensagem oculta (caso esteja na aba 'Detecção de Conteúdo', número doze);
- O número seis refere-se ao botão 'Limpar Campos'. Quando solicitado tem a finalidade de limpar todos os campos da interface;
- O número sete refere-se ao botão 'Mudar Nome do Arquivo'. Quando solicitado, dará a possibilidade de mudar o nome do áudio que será esteganografado;
- O número oito refere-se ao nome do arquivo que foi esteganografado com sucesso e que está presente no diretório 'Esteganografia' criado pelo programa.

- O número dez refere-se ao local onde irá aparecer o texto (caso exista) que permanecia oculto no áudio.

4.3.2 Implementação Esteganografia em Imagem

Para que esse programa funcione de maneira correta é necessário que a imagem esteja no formato *PNG* do contrário não funcionará. Tomando como base esse requisito, é necessário identificar a capacidade de armazenamento que a imagem pode suportar. A capacidade de armazenamento de uma informação pode variar muito de uma imagem para a outra. Isso se dá graças às características particulares de cada imagem como largura e comprimento.

Então para identificar a capacidade é necessário que se tenham as informações de altura e largura. Após essa identificação é feito um cálculo para encontrar a capacidade de armazenamento. Deve-se multiplicar a altura pela largura para identificar a quantidade de *pixels* que a imagem possui e logo após multiplicar por três (3) já que cada *pixel* é dividido pelo padrão RGB em três (vermelho, verde, azul) e divide o resultado por oito já que cada caractere é formado por um conjunto de oito bits.

O programa identifica a capacidade de cada imagem no momento em que a imagem é aberta. Essa ação ocorre quando o usuário aperta o botão "Abrir Imagem". Dessa forma, após o usuário selecionar uma imagem ele calcula a capacidade da imagem e a informa ao usuário.

Após essa ação, o programa disponibiliza para o usuário o diretório onde imagem foi selecionada. Disponibiliza a ação de inserir a mensagem texto em uma imagem através do clique no botão "Executar" além de habilitar os botões "Limpar Campos", o que reinicia todos os campos do programa, e "Mudar Nome do Arquivo", que modifica, caso seja necessário, o nome do arquivo.

As ações (exceto 'mudar o nome do arquivo' que ocorre apenas na aba 'Inserção de Conteúdo'), ocorrem em ambas as abas do programa.

Caso de Uso "Inserção da Mensagem"

Esse caso de uso ocorre quando um usuário (emissor) decide ocultar uma mensagem em uma imagem com o intuito de permitir que apenas pessoal autorizado (receptor) tenha acesso a tal conteúdo. Para que a mensagem seja oculta na imagem é necessário que o usuário digite a mensagem no campo disponível para esse fim e abra a imagem que ocultará a mensagem.

O evento responsável pela inserção de uma mensagem ocorre quando o usuário lança um evento (clique) sobre o botão 'Executar' que está presente na aba 'Inserção de Conteúdo'. Esse evento faz com que o programa capture o texto digitado pelo usuário (caso não

tenha digitado não será inserido nada), faz uma cópia de segurança para o diretório 'Esteganografia'.

Ao capturar a mensagem digitada pelo usuário, é acrescentada ao texto, uma marcação para identificar onde está localizado o início da mensagem e o seu fim. Isso é necessário para que o programa possa identificar a mensagem quando existir a necessidade de retirá-la da imagem.

A marcação é composta pelas strings #I# e #F#. A primeira é responsável por demarcar o início da mensagem e a segunda o seu fim. Após isso, toda mensagem (juntamente com a marcação), é transformada para o seu respectivo número em binário. Cada número ou letra pode ser transformado em binário sendo que estes são compostos por 8 bits (ou 1 byte) cada.

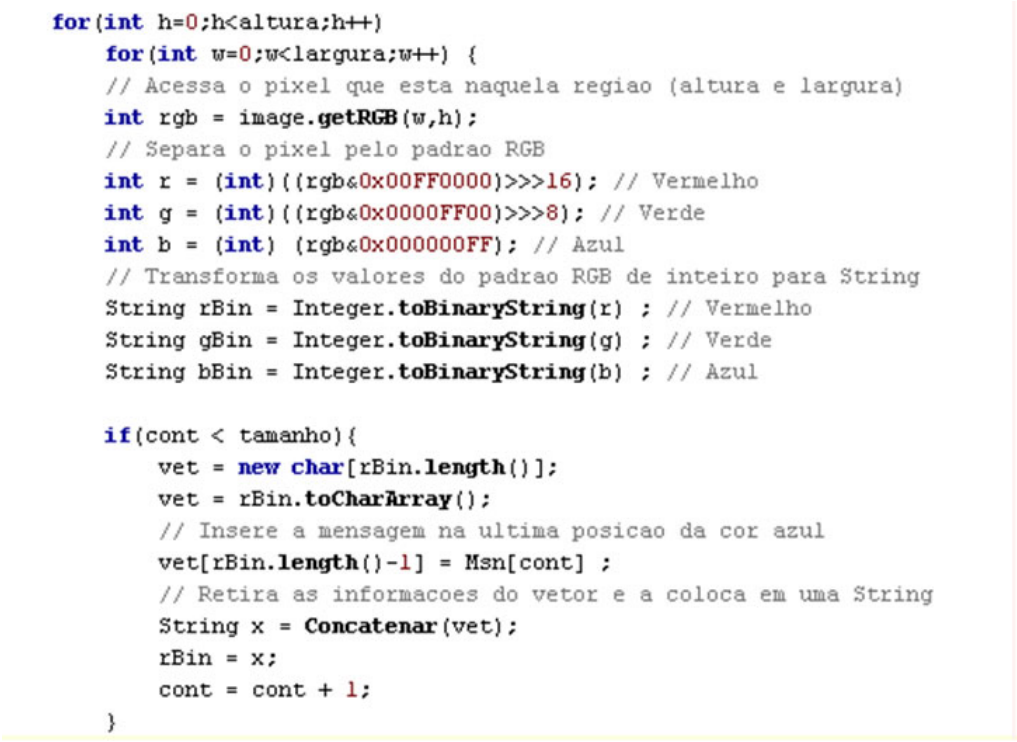
A Figura 4.20 serve como exemplo de visualização da inserção da palavra teste em uma determinada imagem. Existem 3 colunas principais: A primeira representa a coluna do padrão RGB (Red, Green, Blue) que determina a procedência daquele byte. A segunda coluna representa os bytes da imagem principal e a terceira coluna representa os bytes da imagem esteganografada, pode-se perceber que não ocorre mudança em todos os últimos bits significativo da nova imagem formada. Diminuindo a quantidade de alteração da imagem.

1 - red - 110100 110100	31 - red - 110100 110100	61 - red - 110100 110100
2 - green - 111000 111000	32 - green - 111000 111000	62 - green - 111000 111001
3 - blue - 110101 110101	33 - blue - 110101 110100	63 - blue - 110101 110100
4 - red - 110100 110100	34 - red - 110100 110101	64 - red - 110100 110101
5 - green - 111000 111000	35 - green - 111000 111001	65 - green - 111000 111000
6 - blue - 110101 110100	36 - blue - 110101 110100	66 - blue - 110101 110100
7 - red - 110100 110101	37 - red - 110100 110100	67 - red - 110100 110101
8 - green - 111000 111001	38 - green - 111000 111001	68 - green - 111000 111000
9 - blue - 110101 110100	39 - blue - 110101 110100	69 - blue - 110101 110100
10 - red - 110100 110101	40 - red - 110100 110101	70 - red - 110100 110100
11 - green - 111000 111000	41 - green - 111000 111000	71 - green - 111000 111001
12 - blue - 110101 110100	42 - blue - 110101 110101	72 - blue - 110101 110101
13 - red - 110100 110101	43 - red - 110100 110101	73 - red - 110100 110100
14 - green - 111000 111000	44 - green - 111000 111001	74 - green - 111000 111001
15 - blue - 110101 110100	45 - blue - 110101 110100	75 - blue - 110101 110100
16 - red - 110100 110101	46 - red - 110100 110100	76 - red - 110100 110100
17 - green - 111000 111000	47 - green - 111000 111001	77 - green - 111000 111000
18 - blue - 110101 110100	48 - blue - 110101 110101	78 - blue - 110101 110101
19 - red - 110100 110101	49 - red - 110100 110100	79 - red - 110100 110101
20 - green - 111000 111000	50 - green - 111000 111001	80 - green - 111000 111000
21 - blue - 110101 110100	51 - blue - 110101 110101	81 - blue - 110101 110100
22 - red - 110100 110100	52 - red - 110100 110101	82 - red - 110100 110100
23 - green - 111000 111001	53 - green - 111000 111000	83 - green - 111000 111001
24 - blue - 110101 110101	54 - blue - 110101 110101	84 - blue - 110101 110100
25 - red - 110100 110100	55 - red - 110100 110100	85 - red - 110100 110100
26 - green - 111000 111001	56 - green - 111000 111000	86 - green - 111000 111000
27 - blue - 110101 110101	57 - blue - 110101 110100	87 - blue - 110101 110101
28 - red - 110100 110101	58 - red - 110100 110101	88 - red - 110100 110101
29 - green - 111000 111000	59 - green - 111000 111001	
30 - blue - 110101 110101	60 - blue - 110101 110100	

Figura 4.20: Comparação entre bits de uma Imagem

Após esses passos, o método 'insercao_Imagem()' da classe 'InsercaoDaMensagem_Imagem' é acionado. Esse método é responsável por inserir a mensagem na imagem. Para isso todos os pixels da imagem são colocados em um vetor para serem percorridos, dessa forma,

o programa acessa cada pixel individualmente e acessa o seu padrão RGB, já que cada pixel é formado pela união das três cores primárias (vermelho, verde, azul).

Com isso, cada bit, que compõe a mensagem, é colocado no último bit do padrão RGB devido à união das três cores, ou seja, cada pixel é capaz de armazenar três bits da informação. A Figura 4.21 e 4.22 apresenta parte das linhas de código que compõe o método 'insercao_Imagem()'.


```
for(int h=0;h<altura;h++)
    for(int w=0;w<largura;w++) {
        // Acessa o pixel que esta naquela regioao (altura e largura)
        int rgb = image.getRGB(w,h);
        // Separa o pixel pelo padrao RGB
        int r = (int)((rgb&0x00FF0000)>>>16); // Vermelho
        int g = (int)((rgb&0x0000FF00)>>>8); // Verde
        int b = (int) (rgb&0x000000FF); // Azul
        // Transforma os valores do padrao RGB de inteiro para String
        String rBin = Integer.toBinaryString(r) ; // Vermelho
        String gBin = Integer.toBinaryString(g) ; // Verde
        String bBin = Integer.toBinaryString(b) ; // Azul

        if(cont < tamanho){
            vet = new char[rBin.length()];
            vet = rBin.toCharArray();
            // Insere a mensagem na ultima posicao da cor azul
            vet[rBin.length()-1] = Msn[cont] ;
            // Retira as informacoes do vetor e a coloca em uma String
            String x = Concatenar(vet);
            rBin = x;
            cont = cont + 1;
        }
    }
}
```

Figura 4.21: Código de inserção de mensagem em uma imagem 1

```

if(cont < tamanho){
    vet = new char[gBin.length()];
    vet = gBin.toCharArray();
    // Insere a mensagem na ultima posicao da cor azul
    vet[gBin.length()-1] = Msn[cont];
    // Retira as informacoes do vetor e a coloca em uma String
    String x = Concatenar(vet);
    gBin= x;
    cont = cont + 1;
}
if(cont < tamanho){
    vet = new char[bBin.length()];
    vet = bBin.toCharArray();
    // Insere a mensagem na ultima posicao da cor azul
    vet[bBin.length()-1] = Msn[cont] ;
    // Retira as informacoes do vetor e a coloca em uma String
    String x = Concatenar(vet);
    bBin= x;
    cont = cont + 1;
}
// Forma o 'novo' pixel com as devidas atualizacoes
pixels[largura * h + w] = new Color(Integer.parseInt(rBin, 2),Integer.parseInt(gBin, 2)
    Integer.parseInt(bBin, 2) ).getRGB();
}
// Forma a imagem como um todo
image.setRGB(0, 0, largura, altura, pixels, 0, largura);
// Substitui a imagem anterior pela nova
ImageIO.write(image, "PNG", new File(copia));

```

Figura 4.22: Código de inserção de mensagem em uma imagem 2

Após inserção dos bits da mensagem nos bits menos significativos da imagem, é criado uma nova imagem que substitui a cópia feita no diretório 'Esteganografia', retornando assim uma mensagem informando algum problema ou sucesso no processo encerrando assim a execução desse processo.

Caso de Uso "Detecção da Mensagem"

O evento responsável pela detecção de uma mensagem ocorre quando o usuário lança um evento (clique) sobre o botão 'Executar' que está presente na aba 'Detecção de Conteúdo'. Esse evento faz com que o programa ative o método 'deteccao_Imagem()' da classe 'DeteccaoDaMensagem_Imagem' com o intuito de capturar a mensagem oculta e apresentá-la ao usuário.

Para capturar a mensagem oculta, é necessário que os pixels da imagem sejam colocados em um vetor para serem percorridos, dessa forma, o programa acessa cada pixel individualmente e visualiza o seu padrão RGB, já que cada pixel é formado pela união das três cores primárias (vermelho, verde, azul).

Com isso, o programa retira os últimos bits do padrão RGB de cada pixel e os analisa em trincas de oito bits cada, já que, cada caractere é formado pela união de oito bits e a

condição de início e de fim é determinado pela trinca de caracteres #I# e #F#.

As condições de início e fim são analisadas pelos métodos 'confereInicio()' e 'confereFinal()'. O primeiro corresponde por identificar a condição de início da mensagem oculta, caso não existe essa condição inicial (#I#) o usuário é informado que aquela imagem não possui mensagem esteganografada e o segundo é responsável por identificar a condição de parada da mesma (#F#), já que dessa forma pode-se identificar a mensagem oculta por inteiro. Os dois métodos estão discriminados na Figura 6.23.

```
private boolean ConfereInicio(String[] letra) {
    boolean a ;
    // Determina se na primeira trinca de caracteres EXISTE a condicao de inicio #I#
    if(((char)Integer.parseInt(letra[0], 2) == '#') && ((char)Integer.parseInt(letra[1], 2) == 'I')
        && ((char)Integer.parseInt(letra[2], 2) == '#')){
        a = true;
    }else{
        a = false;
    }
    return a;
}
private boolean ConfereFinal(String[] letra) {
    boolean a ;
    // Determina se em uma determinada situacao EXISTE uma trinca de caracteres que caracteriza uma condicao de fim #F#
    if(((char)Integer.parseInt(letra[0], 2) == '#') && ((char)Integer.parseInt(letra[1], 2) == 'F')
        && ((char)Integer.parseInt(letra[2], 2) == '#')){
        a = true;
    }else{
        // Caso NAO EXISTA uma condicao de parada e seja diferente de #I# entao, corresponde a mensagem oculta
        if (!( ((char)Integer.parseInt(letra[0], 2) == '#') || ((char)Integer.parseInt(letra[0], 2) == 'I'))){
            mensagem = mensagem + (char)Integer.parseInt(letra[0], 2);
        }
        a = false;
    }
    return a;
}
```

Figura 4.23: Verifica se existe alguma mensagem oculta ou condição de parada

Quando o programa identifica a posição de parada, o método 'deteccao_Imagem()' retorna a mensagem que estava oculta na imagem. Logo após essa ação a mensagem é apresentada para o usuário.

4.3.3 Implementação Esteganografia em Áudio

Um pré-requisito para que o programa possa funcionar é a necessidade que a extensão do áudio esteja no formato WAV caso contrário, o programa informará uma mensagem informando ao usuário a invalidação de tal áudio. Tomando como base esse requisito, da mesma forma que ocorre com a imagem, é necessária a identificação da capacidade que o áudio suporta já que, varia de acordo com o tamanho da imagem.

Para se determinar a capacidade do áudio é necessário separar o cabeçalho do setor de dados do áudio. O cabeçalho é o local responsável por identificar como o som foi

digitalizado e a qualidade dos canais e suas frequências. Já o setor de dados é o local que encontra as amostras de dados em bits.

Os primeiros 60 bytes do arquivo de áudio com extensão WAV são identificados como sendo o cabeçalho do arquivo. Qualquer alteração nesses bits pode prejudicar ou até mesmo impossibilitar a reprodução do áudio.

O programa identifica a capacidade do áudio no momento em que o usuário clica no botão "Abrir Áudio". Após a escolha do áudio, o programa proporciona para o usuário os botões "Executar" inserir ou retirar uma mensagem do áudio, "Limpar Campos" que reinicia todos os campos do programa, e "Mudar Nome do Arquivo", que modifica, caso seja necessário, o nome do arquivo.

As ações (exceto 'mudar o nome do arquivo' que ocorre apenas na aba 'Inserção de Conteúdo') ocorrem em ambas às abas do programa.

Caso de Uso "Inserção da Mensagem"

Esse caso de uso realiza procedimentos bem parecidos com a "Inserção da Mensagem" em imagem. A principal diferença ocorre no modo como as informações serão ocultas no arquivo. Dessa forma, esse caso de uso ocorre quando um usuário (emissor) decide ocultar uma mensagem em um arquivo de áudio com o intuito de propiciar que apenas pessoal autorizado tenha acesso ao conteúdo.

Com isso, o evento responsável pela inserção de uma mensagem ocorre quando o usuário lança um evento (clique) sobre o botão 'Executar' que está presente na aba 'Inserção de Conteúdo'. Esse evento captura o texto digitado pelo usuário (caso não tenha digitado nenhum caractere será emitido uma mensagem informando ao usuário a sua necessidade) caso o número de caracteres digitado seja maior que a capacidade total do arquivo de áudio o usuário é informado com uma mensagem.

Após capturar a mensagem digitada pelo usuário, é acrescentada ao texto, uma marcação para identificar onde está localizado o início da mensagem e o seu fim. Isso é necessário para que o programa possa identificar a mensagem quando existir a necessidade de retirá-la da imagem.

A marcação é composta, da mesma forma que foi utilizada na imagem, strings #I# e #F#. A primeira é responsável por demarcar o início da mensagem e a segunda o seu fim. Após isso, toda mensagem (juntamente com a marcação), é transformada para o seu respectivo número em binário. Cada caractere possui o seu respectivo valor em binário sendo que estes são compostos por 8 bits (ou 1 byte) cada.

Após essa transformação, essa informação, escrita em bits, é concatenada se transformando assim em uma grande String em seguida, cada caractere dessa String ocupa uma posição em um vetor para ser inserida no áudio. Depois desses passos, o método

'insercao_Audio()' da classe 'InsercaoDaMensagem_Audio' é acionado. Esse método é responsável por inserir a mensagem escrita pelo usuário no áudio.

A Figura 4.24 serve como exemplo de visualização da inserção da palavra teste em um determinado áudio. A primeira coluna representa os bits do áudio original e a segunda coluna os bits do áudio esteganografado. Dessa forma, se pode notar que não ocorre mudança em todos os últimos bits significativo da nova imagem formada, diminuindo assim a quantidade de alterações do áudio.

1 - 11001011 11001010	31 - 11010 11010	61 - 11001000 11001000
2 - 0 0	32 - 1 0	62 - 0 1
3 - 11110 11111	33 - 11010001 11010000	63 - 1011 1010
4 - 1 0	34 - 0 1	64 - 1 1
5 - 11010011 11010010	35 - 11011 11011	65 - 11000111 11000110
6 - 0 0	36 - 1 0	66 - 0 0
7 - 1101 1101	37 - 11010100 11010100	67 - 1101 1101
8 - 1 1	38 - 0 1	68 - 1 0
9 - 11000100 11000100	39 - 11000 11000	69 - 11000001 11000000
10 - 0 1	40 - 1 1	70 - 0 0
11 - 1101 1100	41 - 11010010 11010010	71 - 111 111
12 - 1 0	42 - 0 1	72 - 1 1
13 - 11001101 11001101	43 - 1000 1001	73 - 11001000 11001000
14 - 0 0	44 - 1 1	74 - 0 1
15 - 10011 10010	45 - 11001110 11001110	75 - 111 110
16 - 1 1	46 - 0 0	76 - 1 0
17 - 11010001 11010000	47 - 1010 1011	77 - 11000000 11000000
18 - 0 0	48 - 1 1	78 - 0 1
19 - 1011 1011	49 - 11001011 11001010	79 - 0 1
20 - 1 0	50 - 0 1	80 - 1 0
21 - 11000010 11000010	51 - 110 111	81 - 10111010 10111010
22 - 0 0	52 - 1 1	82 - 0 0
23 - 11001 11001	53 - 11000110 11000110	83 - 11111011 11111011
24 - 1 1	54 - 0 1	84 - 0 0
25 - 11010111 11010110	55 - 111 110	85 - 10111010 10111010
26 - 0 1	56 - 1 0	86 - 0 0
27 - 11010 11011	57 - 11000111 11000110	87 - 10 11
28 - 1 1	58 - 0 1	88 - 1 1
29 - 11010010 11010010	59 - 10 11	
30 - 0 1	60 - 1 0	

Figura 4.24: Comparação entre bits do arquivo de áudio

Para que isso seja possível, o setor de dados do áudio é percorrido byte a byte sendo substituído o seu ultimo bit por aquele que está disposto dentro do vetor até terminar as informações existentes nele. A Figura 4.25 apresenta parte das linhas de código que compõe o método 'insercao_Audio()'.

```

DataInputStream fis = new DataInputStream(new BufferedInputStream(
    new FileInputStream(arquivoEntrada)));
DataOutputStream fos = new DataOutputStream(new BufferedOutputStream(
    new FileOutputStream(arquivoSaida)));
while ((temp = fis.read()) != -1) {

    if (cont < inicio) {
        fos.write(temp);

    } else {
        if (tamanho > contBit) {
            //Transforma o byte em String
            String audioBin = Integer.toBinaryString(temp);
            //Separa a String em um vetor de caracteres
            vet = new char[audioBin.length()];
            vet = audioBin.toCharArray();
            // Insere a mensagem na ultima posicao do byte
            vet[audioBin.length() - 1] = Msn[contBit];
            // Retira as informacoes do vetor e a coloca em uma String
            String x = concatenar(vet);
            //Forma o novo arquivo de audio
            fos.write(Integer.parseInt(x));
            contBit = contBit + 1;
            bit = bit + contBit + " - " + audioBin + " " + x + "\n";
        } else {
            fos.write(temp);
        }
    }
}

```

Figura 4.25: Código de inserção de mensagem em áudio

Caso de Uso "Detecção da Mensagem"

Esse caso de uso possui algumas características em comum com a detecção da mensagem em imagem. O que diferencia é a forma como arquivos são acessados para a obtenção da mensagem. Para que isso ocorra, é necessário que o usuário lance um evento (clique) sobre o botão 'Executar' que está na aba 'Detecção de Conteúdo' após este ter selecionado um áudio com extensão (*WAP*).

Esse evento faz com que o programa ative o método 'deteccao_Audio()' da classe 'DeteccaoDaMensagem_Audio' com o intuito de capturar a mensagem oculta e apresentá-la ao usuário.

Para captura a mensagem oculta, é necessário que os bytes do setor de dados sejam percorridos (a partir do byte 60), analisando assim os seus últimos bits com o intuito de identificar a existência de uma mensagem oculta.

Dessa forma, os bits são concatenados em trincas de oito bits (formando assim um byte) para analisar a condição de início (#I#) e a condição de parada da mensagem (#F#) caso existam. Caso não existam o usuário é informado com uma mensagem de que aquele arquivo não possui mensagem oculta.

Os métodos 'confereInicio()' e 'confereFinal()' são responsáveis por de terminar a existência ou não da mensagem oculta. O primeiro é responsável por identificar a existência ou não de mensagem oculta (#I#) já o segundo é responsável por identificar a condição de termino da mensagem oculta (#F#).

Capítulo 5

Resultado e Considerações Finais

Para realizar uma análise do programa será realizado dois exemplos com o propósito de demonstrar os resultados obtidos através de comparações entre o antes e o depois da inserção da mensagem em imagens e em áudios.

Como primeiro exemplo, será inserido a palavra teste em uma imagem e em áudio. Para que essa palavra possa ser inserida, tanto na imagem como em um áudio, ela recebe dois identificadores um para identificar o início da palavra #I# e outro para identificar o fim dessa palavra #F#. A mensagem mais os identificadores ficam da seguinte forma: #I#teste#F# e sua representação em binário é representada por: 00100011 01001001 00100011 01110100 01100101 01110011 01110100 01100101 00100011 01000110 00100011.

Para que essa mensagem possa ser inserida (tanto em um áudio como em uma imagem) será necessário utilizar 88 bytes do arquivo recipiente.

Será utilizado um histograma (representa a frequência com que as cores aparecem em uma imagem) para comparar a imagem original da esteganografada e será retirada uma amostra de onda de áudio através de um editor de áudio para comparar uma amostra de onda original com uma amostra de onda esteganografada.

As Figuras 5.1 e 5.2 compreendem ao histograma e indica a de comparação entre uma imagem original e outra esteganografada.

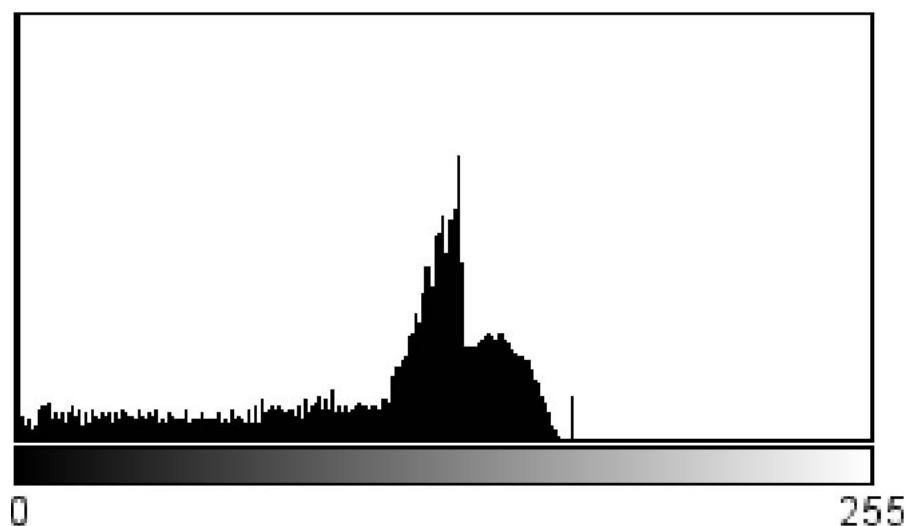


Figura 5.1: Histograma da Imagem Original

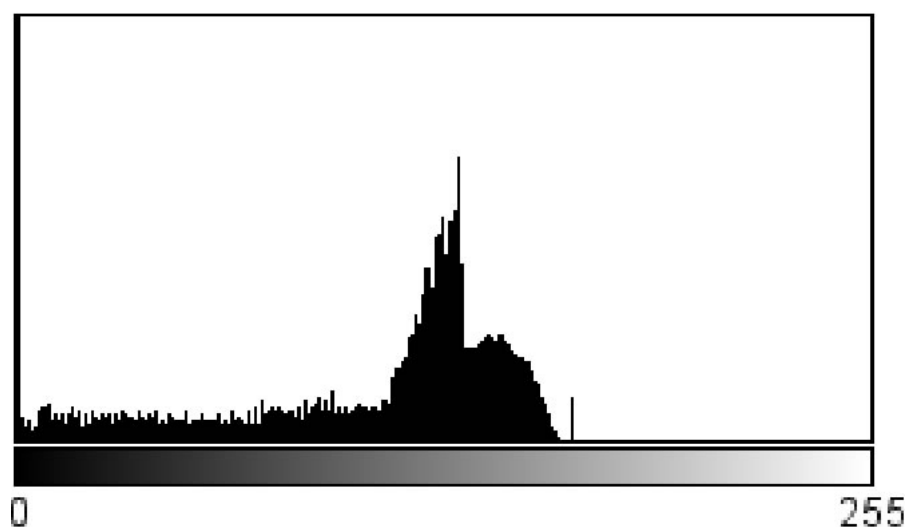


Figura 5.2: Histograma da Imagem Esteganografada

As Figuras 5.3 e 5.4 foram geradas a partir de um editor de áudio e dessa forma pode-se comparar uma amostra de uma onda de áudio original com uma amostra de onda de áudio esteganografada e com isso identificar onde e o que foi alterado. A região circulada representa o local onde ocorreu a inserção da palavra teste no áudio.

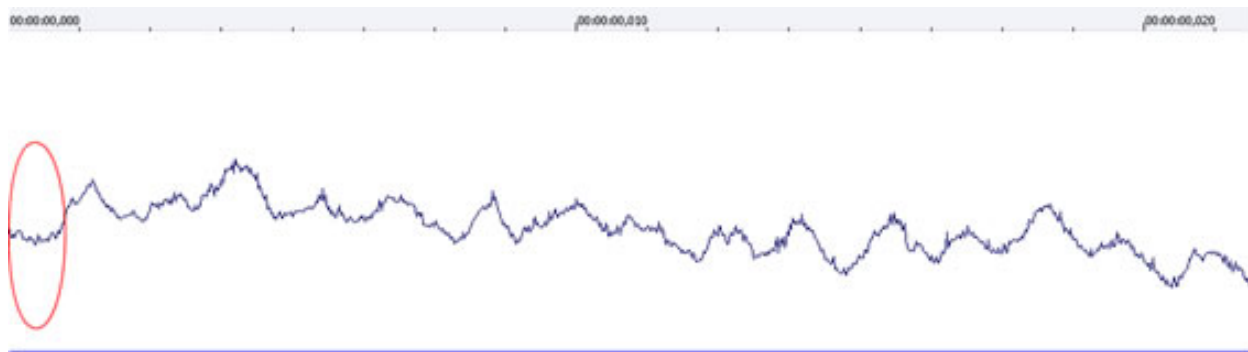


Figura 5.3: Áudio Original

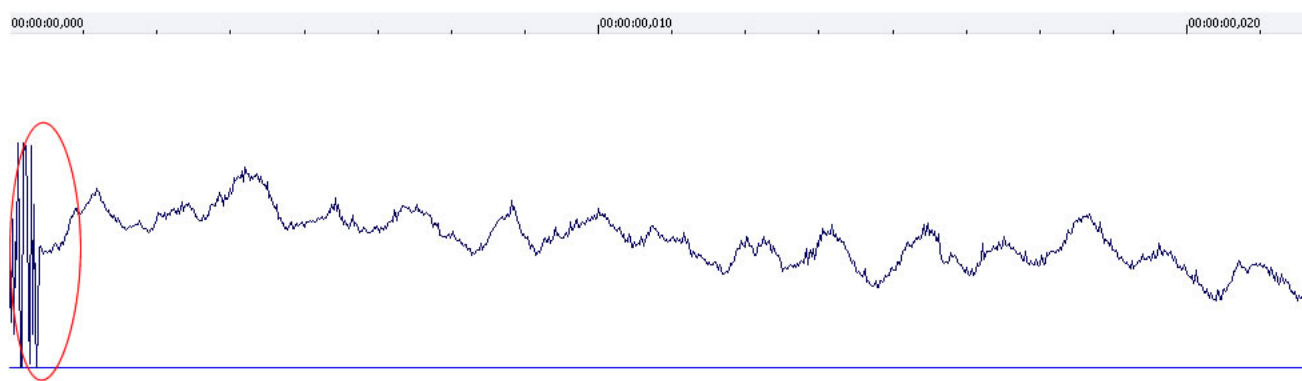


Figura 5.4: Áudio Esteganografado

No Segundo exemplo será inserido um trecho de uma carta escrito por Osama Bin Laden [CARVALHO, 2005]. Essa carta foi esteganografada em uma imagem e foi enviado por e-mail para seus seguidores. O trecho da carta que será inserido em uma imagem e em áudio é: *"Tenho também importante mensagem para nossos jovens, nesse período difícil que atravessamos. Vocês devem levantar bem alto a bandeira da jihad contra os sionistas. Vocês são os legítimos sucessores de nossos valentes ancestrais. A propósito de nossos jovens, é bom saber que eles acreditam no paraíso após a morte. Eles sabem que tomar parte na luta contra os infiéis não abreviará seus dias. E que estar ausente dela não tornará seus dias mais longos. Nossos jovens sabem muito bem o significado dos versos: Se a morte é certa, então é uma vergonha morrer covarde mente. Quem não morrer pela espada morrerá por outra razão ..."*. Esse trecho foi colocado com o intuito de mostrar que a esteganografia esta sendo utilizada atualmente e, além disso, pode ser utilizada por pessoas má intencionadas.

Da mesma forma que no primeiro exemplo será inserido os identificadores #I# no

começo e o #F# no final da mensagem.

Para que essa mensagem possa ser inserida (tanto em um áudio como em uma imagem) será necessário utilizar 5064 bytes do arquivo recipiente.

As Figuras 5.5 e 5.6 compreendem ao histograma e indica a de comparação entre uma imagem original e outra esteganografada. A região circulada representa o local onde ocorreu a inserção do texto na imagem.

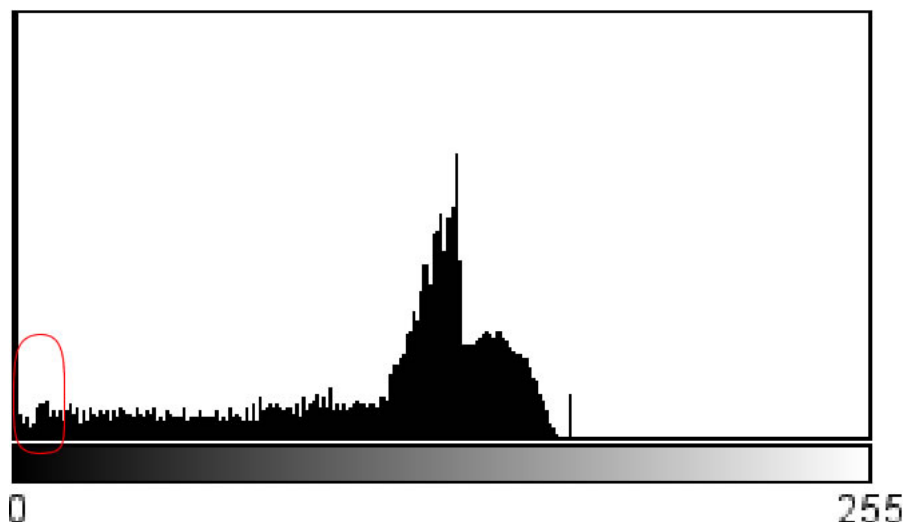


Figura 5.5: Histograma da Imagem Original

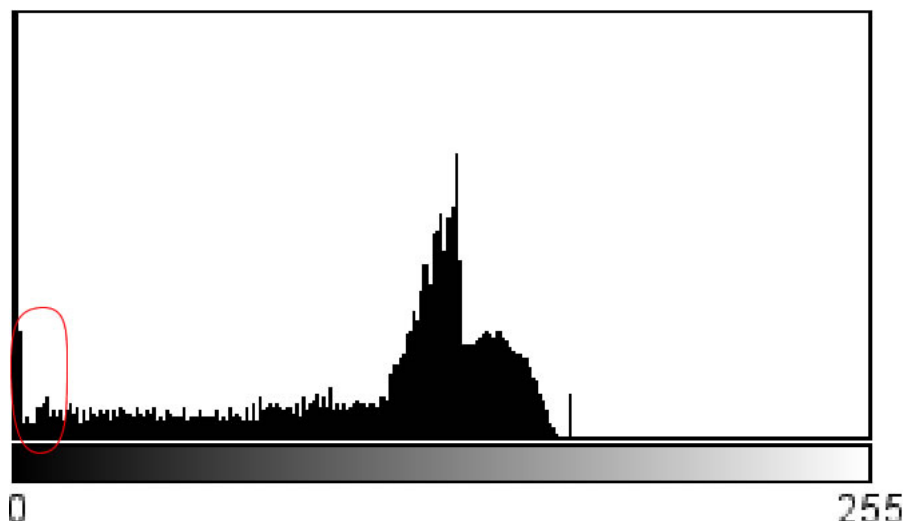


Figura 5.6: Histograma da Imagem Esteganografada

As Figuras 5.7 e 5.8 foram geradas a partir de um editor de áudio e dessa forma pode-se comparar uma amostra de uma onda de áudio original com uma amostra de onda de áudio esteganografada e com isso identificar onde e o que foi alterado. A região circulada representa o local onde ocorreu a inserção do texto no áudio.

Analisando apenas as figuras relacionadas com a imagem, pode se perceber que ao

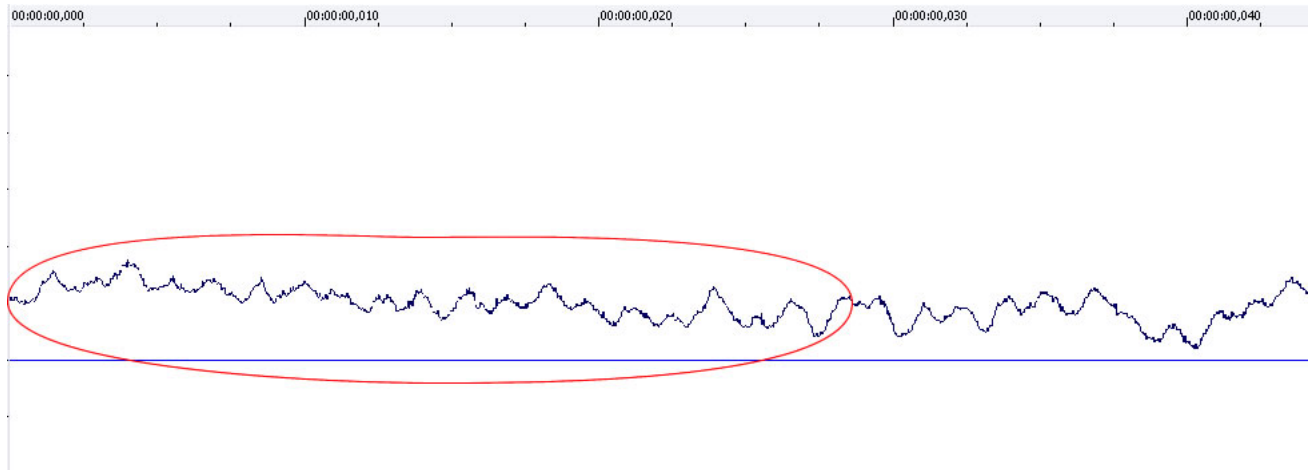


Figura 5.7: Áudio Original

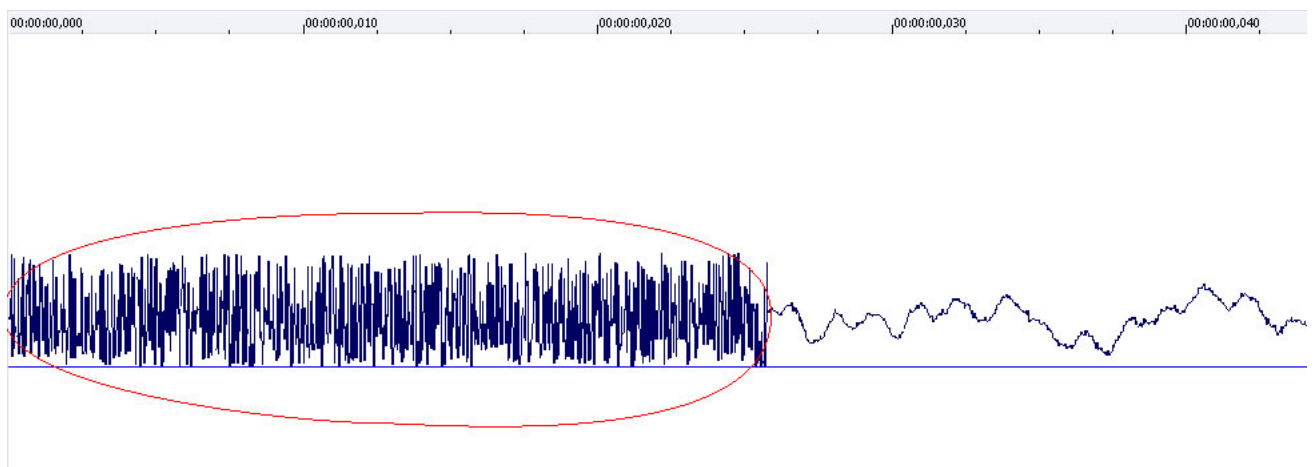


Figura 5.8: Áudio Esteganografado

inserir a palavra teste em uma imagem o histograma não é capaz de sinalizar uma modificação significativa. Isso não significa que a imagem esteganografada não sofreu modificação, mas sim que a modificação não foi suficiente para afetar substancialmente o histograma.

Porém com o aumento dos caracteres inseridos na imagem (o trecho da carta escrito por Osama Bin Laden), pode-se notar que em certo local ocorreu uma diferença significativa. Isso não significa que as alterações ocorreram somente no local demarcado, mas sim que a modificação naquela região foi suficiente para aparecer no histograma.

A complexidade do algoritmo de imagem é quadrática, pois é percorrido pixel a pixel da imagem e para que essa operação seja realizada é necessário processar os itens em pares (altura e largura). Dessa forma, o tempo de inserção da mensagem irá depender do tamanho da altura e da largura da imagem e o tempo de remoção da mensagem irá depender da altura e da largura da imagem referentes a posição que possui a mensagem escondida.

Quando se trata de áudio, ouvido humano não consegue identificar as pequenas modificações desse arquivo já que as suas alterações estão localizadas nos primeiros milésimos de segundos do áudio. Dessa forma, por haver poucas modificações e o intervalo de tempo ser muito curto não é possível identificar tal alteração.

A complexidade do algoritmo de áudio é linear, pois é percorrido byte a byte do setor de dados da imagem. Dessa forma, o tempo de inserção da mensagem irá depender do tamanho do setor de dados do áudio e o tempo de remoção da mensagem irá depender unicamente do número de caracteres que a mensagem possui, ou seja, quanto maior for o número de caracteres maior será o tempo de processamento do programa além da possibilidade de aumentar a chance do ouvido humano identifique tal alteração. Com isso pode-se concluir que para se ter um programa eficiente, é indicado a inserção de pouca quantidade de caracteres.

Ao comparar o arquivo de áudio com o de imagem, pode-se perceber que o arquivo de áudio comporta um volume maior de dados, porém pode permitir que o ouvido humano perceba alguma alteração significativa, ou seja, escute algum tipo de ruído. Ao inserir uma quantidade pequena de informação em ambos os arquivos, o arquivo de áudio sofre uma maior alteração em seu conteúdo do que o arquivo de imagem.

Capítulo 6

Conclusão

6.1 Conclusão

A segurança da informação representa um grande desafio em virtude do grande número de técnicas capazes de violar a privacidade. Para evitar que pessoas não autorizadas tenham acesso a uma determinada informação, várias técnicas estão sendo utilizadas, dentre elas se inclui a esteganografia.

A esteganografia pode ser considerada uma proteção eficiente quando o assunto é proteção digital. Já que permite a privacidade entre as comunicações pela rede de computadores ocultando a informação. Contudo pode ocorrer que tal privacidade fique a disposição de pessoas más intencionadas.

Nesse trabalho de conclusão de curso, foi oferecido e avaliado o método de esteganografia LSB aplicado em imagem e áudio. O aproveitamento dessa técnica se mostrou de grande valia além de ser bastante eficiente principalmente quanto à inserção de uma pequena quantidade de informação, pois com o aumento dos caracteres inseridos nos arquivos (imagem e áudio) aumenta a probabilidade do ser humano identificar alguma imperfeição.

Quanto maior o número de caracteres a serem inseridos, maior será o grau de processamento do software. Dessa forma, quanto maior a mensagem, maior será o tempo necessário para inseri-la no arquivo recipiente e como consequência disso, maior será o tempo necessário para obter a mensagem oculta.

Pode-se perceber também que é possível realizar comunicação segura utilizando a técnica esteganográfica LSB na rede de computadores (*Internet*), dificultando a identificação do conteúdo oculto por pessoas não autorizadas.

Foi encontrada durante o desenvolvimento do software uma restrição a respeito das extensões dos arquivos. Só podem ser utilizados arquivos *WAV* nos áudios e *PNG* com relação às imagens já que, em ambas as extensões, não são utilizadas compressão com perde de informação. Facilitando assim a inserção e obtenção da mensagem oculta. As

demais extensões não são indicadas já que utilizam compressão por perda não garantindo assim os objetivos da técnica abordada.

Dessa forma, em decorrência dos resultados oferecidos pode-se concluir que a utilização da técnica esteganográfica LSB em áudio e imagem é viável uma vez que, permite uma comunicação segura entre pessoas independente do local onde estejam. Conservando assim a privacidade e mantendo o conteúdo da mensagem em sigiloso.

6.2 Trabalhos Futuros

Como trabalho futuro, pode-se implementar a compactação de Huffman para diminuir o tamanho da mensagem no formato texto além de adicionar a criptografia como forma de tornar a informação mais protegida.

Também pode ser desenvolvida em trabalhos futuros a utilização de outros formatos de arquivo tanto de imagem como de áudio, principalmente os arquivos que sofrem compressão por perdas como JPEG e MP3.

Além disso, pode-se implementar um programa que possa utilizar a técnica LSB da esteganografia que possa inserir mensagens no formato texto em vídeos ou no áudio de um vídeo podendo dessa forma, esconder um número maior de informação ao se comparar com uma imagem ou um áudio.

Referências

- CAMEPELLO, R. (2001). Workshop em segurança de sistemas computacionais. In *Anais do WSEG'2001: SBC*.
- CARVALHO, D. F. D. (2005). Exploração tecnológica para esteganografia em vídeos digitais. São Paulo, SP, Brasil. Universidade de São Paulo, USP. Graduação em Bacharelado Em Informática.
- DIAS, C. (2000). *Segurança e Auditoria da Tecnologia da Informação*. Axcel Books, Rio de Janeiro, RJ, Brazil.
- EDDIE B. L. FILHO, EDUARDO A. B. DA SILVA, M. B. D. C. W. S. S. J. J. K. (2005). Electrocardiographic signal compression using multiscale recurrent patterns. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*.
- EDUARDO P. JULIO, WAGNER G. BRAZIL, C. V. N. A. (2007). Esteganografia e suas aplicações. Rio de Janeiro, RJ, Brasil. VII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais.
- EVERSON CARLOS MAUDA, GILBERTO BORDIGNIN CONOR, J. R. K. H. P. (2008). Identificação de notas musicais em arquivos wave monofônicos. *Workshop Científico de Computação*, III.
- FLACKE, M.-L. (1998). Ru10: Technical communication and encryption.
- GOIS, M. C. A. (2003). Mascaramento de informações: histórico, definições e aplicações. *Revista Eletrônica de Iniciação Científica*, III.
- GONZALEZ, R. C. e WOODS, R. E. (1998). *Digital Image Processing*. Prentice-Hall, Boston, MA, USA.
- IGUCHI, F. (2007). Inserção de marcas d'Água digitais usando recorrência de padrões multiescalas. Universidade Federal do Rio de Janeiro.
- JASCONE, F. L. T. (2003). Protótipo de software para ocultar texto criptografado em imagens digitais. In *Trabalho de Conclusão de Curso*, page 64, Santa Catarina, SC, Brasil. Universidade Federal de Blumenau.
- JULIO, E. P. (2007). Uma arquitetura de sistemas de detecção de intrusão em redes ad hoc sem fio usando esteganografia e mecanismos de reputação. In <http://www.ic.uff.br/PosGraduacao/Dissertacoes/369.pdf>. Universidade Federal Fluminense.

- KOBUSZEWSKI, A. (2004). Protótipo de software para ocultar textos compactados em arquivos de Áudio utilizando esteganografia. In *Trabalho de Conclusão de Curso (Graduação em Bacharelado em Ciências da Computação)*, pages 10–18, Santa Catarina, SC, Brasil. Universidade Regional de Blumenau.
- LOPES, A. B. (2004). Documento bitmap: Uma praga que se alastra. *Revista Profissional Publish*, pages 28-31, julho/agosto de 2004.
- MEDEIROS, C. D. R. (2001). Segurança da informação: implantação de medidas e ferramentas de segurança da informação. disponível: http://www.linuxsecurity.com.br/info/general/tce_seguranca_da_informacao.pdf.
- MISAGHI, M. (2001). Avaliação de modificações do cifrador caótico de roski. Master's thesis, Universidade Federal de Santa Catarina.
- PETRI, M. (2004). Esteganografia. Santa Catarina, SC, Brasil, http://www.mlaureano.org/aulas_material/orientacoes2/ist_2004_petri_esteganografia.pdf. Sociedade Educacional de Santa Catarina (SOCIESC).
- PFLEEGER, S. L. (2003). *Engenharia de Software - Teoria e Prática*. Prentice-Hall.
- POLLON, V. (2007). Esteganografia: a arte de ocultar arquivos dentro de arquivos. arXiv:quant-ph/0211100.
- POPA, R. (1998). An analysis of steganography technique. Master's thesis, Department of Computer Science and Software Engineering of The Polytechnic. Universidade de Timisora (Romania).
- ROCHA, A. D. R. (2003). Camaleão: Um software para segurança digital utilizando esteganografia. Minas Gerais, MG, Brasil. Departamento de Ciências da Computação. Universidade Federal de Lavras. Trabalho de Conclusão de Curso. <http://www.ic.unicamp.br/rocha/sci/stego/src/monografia.pdf>.
- SCHIAVONI, F. L. (2008). 10 razões por que usar java. disponível: <http://flavioschiavoni.blogspot.com/2008/09/10-razes-por-que-usar-java.html>.

Apêndice A

Código Fonte

A.1 Classe Principal

```
/*Mostra os principais métodos da classe Principal*/

public class Principal extends javax.swing.JFrame {

private void B_Executar_Insercao_ImagemMouseClicked(java.awt.event.MouseEvent evt) {

    InsercaoDaMensagem_Imagem insercao = null;
    String Msn = , bit = ;
    char[] mens;
    mensagem = T_Mensagem_Insercao_Imagem.getText();

    if (mensagem.length() != capacidade_De_armazenamento_Imagem) {
        if (!mensagem.equals("")) {
            System.out.print(mensagem.length());
            try {
                // Captura a mensagem digitada pelo usuario.
                Msn = mensagem_Final(true);
                // Determina o tamanho da imagem.
                mens = new char[Msn.length()];
                mens = Msn.toCharArray();
                // Cria o diretório 'Esteganografia' caso este ainda não exista.
                criaArquivo();
                String copia = "Esteganografia/" + nome_imagem_insercao;
                insercao = new InsercaoDaMensagem_Imagem();
                // Faz uma cópia do arquivo original para dentro do diretório 'Esteganografia'.
                insercao.copiaArquivo(caminho_insercao, copia);
                // Insere a mensagem na imagem.
                bit = insercao.insercao_Imagem(mens, Msn.length());
                // Caso a insercao ocorra com sucesso.
                JOptionPane.showMessageDialog(null, "A mensagem foi inserida com sucesso.");
            }
        }
    }
}
```

```

        } catch (IOException ex)
            ex.printStackTrace();
    }
}
} else {
    // Caso a imagem nao suporte o tamanho do texto.
    JOptionPane.showMessageDialog(null, "O texto digitado é inválido ou maior do que o requerido.");
}
}

private void B_Executar_Deteccao_ImagemMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        DeteccaoDaMensagem_Imagem rm = new DeteccaoDaMensagem_Imagem();
        String m = rm.deteccao_Imagem(caminho_deteccao);
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}

private void JBExecutarAudio_InserirMouseClicked(java.awt.event.MouseEvent evt) {
    InsercaoDaMensagem_Audio in_audio = null;
    String Msn = , bit = ;
    char[] mens;
    int i = 60;

    // Captura a mensagem digitada pelo usuario.
    Msn = mensagem_Final(false);
    if (tamanho_mensagem == 0) {
        JOptionPane.showMessageDialog(null, "Digite a mensagem que será Esteganografada.");
    } else {
        // Analisa se a imagem suporta o texto digitado.
        if (tamanho_mensagem != (capacidade_De_armazenamento_Midia - i)) {
            try {
                // Determina o tamanho do audio.
                mens = new char[Msn.length()];
                mens = Msn.toCharArray();
                // Cria o diretório 'Esteganografia' caso este ainda não tenha sido criado.
                criaArquivo();
                String copia = "Esteganografia/" + nome_arquivo_insercao;
                in_audio = new InsercaoDaMensagem_Audio();
                // Insere a mensagem no audio.
                bit = in_audio.insercao_Audio(mens, tamanho_mensagem, caminho_insercao, copia);
                // Caso a insercao ocorra com sucesso.
                JOptionPane.showMessageDialog(null, "A mensagem foi inserida com sucesso.");
            } catch (IOException ex) {

```

```

        ex.printStackTrace();
    }
} else {
    // Caso o audio nao suporte o tamanho do texto.
    JOptionPane.showMessageDialog(null, "O texto digitado Ã© maior do que o requerido.");
}
}
}

private void JBExecutarAudio_DetectarMouseReleased(java.awt.event.MouseEvent evt) {
    try {
        DeteccaoDaMensagem_Audio rm = new DeteccaoDaMensagem_Audio();
        String m = rm.deteccao_Audio(caminho_deteccao);
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}
}
}

```

A.2 Classe InsercaoDaMensagem_Imagem

```

/*Mostra o principal método da classe InsercaoDaMensagem_Imagem*/

public class InsercaoDaMensagem_Imagem {

    // Recebe o vetor de caracteres com a mensagem e o tamanho da mensagem
    public String insercao_Imagem(char[] Msn, int tamanho)throws IOException{
        int cont = 0, contador = 0;
        String bit = ;
        char[] vet;
        // Acessa a imagem que esta no diretorio 'Esteganografia'
        javax.swing.JFileChooser j = new javax.swing.JFileChooser() ;
        BufferedImage image = ImageIO.read(new File(copia));
        // Dimesoes da Imagem
        int largura = image.getWidth();
        int altura = image.getHeight();
        // Identifica todas as bandas da imagem
        int nbands = image.getSampleModel().getNumBands();
        // Identifica todos os pixels da imagem
        int[] pixels = new int[nbands*largura*altura];
        for(int h=0;h<altura;h++)
            for(int w=0;w<largura;w++) {
                contador = contador + 1;
            }
        // Acessa o pixel que esta naquela regioao (altura e largura)
    }
}

```

```

int rgb = image.getRGB(w,h);
// Separa o pixel pelo padrao RGB
int r = (int)((rgb&0x00FF0000)>>16); // Vermelho
int g = (int)((rgb&0x0000FF00)>>8); // Verde
int b = (int) (rgb&0x000000FF); // Azul
// Transforma os valores do padrao RGB de inteiro para String
String rBin = Integer.toBinaryString(r) ; // Vermelho
String gBin = Integer.toBinaryString(g) ; // Verde
String bBin = Integer.toBinaryString(b) ; // Azul
if(cont < tamanho){
    vet = new char[rBin.length()];
    vet = rBin.toCharArray();
    // Insere a mensagem na ultima posicao da cor VERMELHO
    vet[rBin.length()-1] = Msn[cont] ;
    // Retira as informacoes do vetor e a coloca em uma String
    String x = concatenar(vet);
    rBin = x;
    cont = cont + 1;
}
contador = contador + 1;
if(cont < tamanho){
    vet = new char[gBin.length()];
    vet = gBin.toCharArray();
    // Insere a mensagem na ultima posicao da cor VERDE
    vet[gBin.length()-1] = Msn[cont];
    // Retira as informacoes do vetor e a coloca em uma String
    String x = concatenar(vet);
    gBin= x;
    cont = cont + 1;
}
contador = contador + 1;
if(cont < tamanho){
    vet = new char[bBin.length()];
    vet = bBin.toCharArray();
    // Insere a mensagem na ultima posicao da cor AZUL
    vet[bBin.length()-1] = Msn[cont] ;
    // Retira as informacoes do vetor e a coloca em uma String
    String x = concatenar(vet);
    bBin= x;
    cont = cont + 1;
}
// Forma o 'novo' pixel com as devidas atualizacoes
pixels[largura * h + w] = new Color(Integer.parseInt(rBin, 2),Integer.parseInt(gBin, 2) ,Integer.parseInt(bBin, 2) ).getRGB();
}

```

```

// Forma a imagem como um todo
image.setRGB(0, 0, largura, altura, pixels, 0, largura);
// Substitui a imagem anterior pela nova
ImageIO.write(image, "PNG", new File(copia));
}
}

```

A.3 Classe DeteccaoDaMensagem_Imagem

```

/*Mostra o principal método da classe InsercaoDaMensagem_Imagem*/

public class DeteccaoDaMensagem_Imagem {

public String deteccao_Imagem(String path) throws IOException{
    char x ;
    int cont = 0;
    // Acessa a imagem que esta no diretorio 'Esteganografia'
    javax.swing.JFileChooser j = new javax.swing.JFileChooser() ;
    BufferedImage image = ImageIO.read(new File(path));
    // Dimesoes da Imagem
    int width = image.getWidth();
    int height = image.getHeight();
    for(int h=0;h<height;h++){
        for(int w=0;w<width;w++) {
            // Acessa o pixel que esta naquela regio (altura e largura)
            int rgb = image.getRGB(w,h);
            // Separa o pixel pelo padrao rgb
            int r = (int)((rgb&0x00FF0000)>>>16); // Vermelho
            int g = (int)((rgb&0x0000FF00)>>>8); // Verde
            int b = (int) (rgb&0x000000FF); // Azul
            // Transforma os valores do padrao RGB de inteiro para String
            String rBin = Integer.toBinaryString(r) ; // Vermelho
            String gBin = Integer.toBinaryString(g) ; // Verde
            String bBin = Integer.toBinaryString(b) ; // Azul
            // Retira o ultimo bit do byte retirado do padrao RGB VERMELHO
            x = rBin.charAt(rBin.length()-1);
            resposta = resposta + x;
            cont = cont +1;
            // Identifica se existe mensagem e as condicao de inicio #I# e fim #F# marcador(resposta,cont);
            // Identifica se formacao de cada caracter
            cont = condicao(cont);
            // Retira o ultimo bit do byte retirado do padrao RGB VERDE
            x = gBin.charAt(gBin.length()-1);

```

```

        resposta = resposta + x;
        cont = cont +1;
        // Identifica se existe mensagem e as condicao de inicio #I# e fim #F#
        marcador(resposta,cont);
        // Identifica se formacao de cada caracter
        cont = condicao(cont);
        // Retira o ultimo bit do byte retirado do padrao RGB AZUL
        x = bBin.charAt(bBin.length()-1);
        resposta = resposta + x;
        cont = cont +1;
        // Identifica se existe mensagem e as condicao de inicio #I# e fim #F#
        marcador(resposta,cont);
        // Identifica se formacao de cada caracter
        cont = condicao(cont);
        if(terminou)
            // Retorna a mensagem oculta
            return mensagem;
    }
}
return ;
}
}

```

A.4 Classe InsercaoDaMensagem_Audio

/*Mostra o principal método da classe InsercaoDaMensagem_Audio*/

```

public class InsercaoDaMensagem_Audio {

    public String insercao_Audio(char[] Msn, int tamanho, String arquivoEntrada, String arquivoSaida)
    throws IOException {
        char[] vet;
        int cont = 0, temp;
        int contBit = 0;
        int inicio = 60;
        String bit = ;

        DataInputStream fis = new DataInputStream(new BufferedInputStream(
            new FileInputStream(arquivoEntrada)));
        DataOutputStream fos = new DataOutputStream(new
            BufferedOutputStream( new FileOutputStream(arquivoSaida)));
        while ((temp = fis.read()) != -1) {

```

```

    if (cont != inicio) {
        fos.write(temp);
    } else {
        if (tamanho != contBit) {
            //Transforma o byte em String
            String audioBin = Integer.toBinaryString(temp);
            //Separa a String em um vetor de caracteres
            vet = new char[audioBin.length()];
            vet = audioBin.toCharArray();
            // Insere a mensagem na ultima posicao do byte
            vet[audioBin.length() - 1] = Msn[contBit];
            // Retira as informacoes do vetor e a coloca em uma String
            String x = concatenar(vet);
            //Forma o novo arquivo de audio
            fos.write(Integer.parseInt(x));
            contBit = contBit + 1;
        } else {
            fos.write(temp);
        }
    }
    cont = cont + 1;
}
fis.close();
fos.close();
}
}

```

A.5 Classe DeteccaoDaMensagem_Audio

/*Mostra o principal método da classe DeteccaoDaMensagem_Audio*/

```

public class DeteccaoDaMensagem_Audio {

    public String deteccao_Audio(String path) throws IOException
    {
        char ultimobit;
        int contar = 0, temp, contador = 0;
        int inicio = 60;
        DataInputStream fis = new DataInputStream(new BufferedInputStream(
            new FileInputStream(path)));
        while((temp = fis.read()) != -1) {
            if(contador != inicio){
                String x = Integer.toBinaryString(temp) ;
                ultimobit = x.charAt(x.length()-1);
            }
        }
    }
}

```



```

    resposta = resposta + ultimobit;
    contar = contar +1;
    // Identifica se existe mensagem e as condicao de inicio #I# e fim #F#
    marcador(resposta,contar);
    // Identifica se formacao de cada caracter
    contar = condicao(contar);
}
contador = contador + 1;
}
fis.close();
return mensagem;
}
}

```