ANDERSON DE REZENDE ROCHA

CAMALEÃO: UM SOFTWARE PARA SEGURANÇA DIGITAL UTILIZANDO ESTEGANOGRAFIA

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências para conclusão do curso de Ciência da Computação

Orientador
Prof. Heitor Augustus Xavier Costa

Co-Orientador
Prof. Lucas Monteiro Chaves

Lavras
Minas Gerais - Brasil
2003

ANDERSON DE REZENDE ROCHA

CAMALEÃO: UM SOFTWARE PARA SEGURANÇA DIGITAL UTILIZANDO ESTEGANOGRAFIA

Monografia apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências para conclusão do curso de Ciência da Computação

Aprovada em *10 de dezembro de 2003*

Prof. Mário Luiz Rodrigues de Oliveira

Prof. Heitor Augustus Xavier Costa
(Orientador)

Prof. Lucas Monteiro Chaves
(Co-Orientador)

Lavras
Minas Gerais - Brasil

Epígrafe

Não basta ensinar ao homem uma especialidade, porque se tornará assim uma máquina utilizável e não uma personalidade. É necessário que adquira um sentimento, um senso prático daquilo que vale a pena ser empreendido, daquilo que é belo, do que é moralmente correto.

(*Albert Einstein*)

Dedicatória

Dedico este trabalho à minha mãe por ter adiado a realização de muitos de seus sonhos em benefício da realização dos meus. À ela toda a felicidade do mundo. Dedico também à minha namorada Aninha, por me fazer a pessoa mais feliz do mundo.

Agradecimentos

Todo trabalho, por mais simples que seja, não é feito sem a ajuda de outras pessoas. Ao longo do desenvolvimento desta pesquisa tive muita ajuda. Desta forma, gostaria de agradecer aos meus (des)orientadores *Heitor* e *Lucas* por suas contribuições muito relevantes. Agradeço também aos *hackers* do grupo de discussão de criptografia *Sci-crypt* e, em especial, ao colega *Mok-Kong Shen* pelas excelentes contribuições. Também foram muito válidas as dicas de otimização do prof. *Mário Luiz*. Finalmente, gostaria de agradecer a todos os *meus colegas de classe* que me acompanharam durante os 4 anos de graduação. A eles desejo muito sucesso. Em especial, agradeço ao meu amigo e colega *Júlio Alves* pelas dicas sempre muito relevantes ao longo de minha graduação.

Camaleão: um Software para Segurança Digital Utilizando Esteganografia

A busca por novos meios eficientes e eficazes de proteção digital é um campo de pesquisas fundamentado nos mais variados campos da ciência. A *esteganografia* configura-se como um destes meios de proteção. Inclui um vasto conjunto de métodos para comunicações secretas tais como tintas "invisíveis", micro-pontos, arranjo de caracteres (*character arrangement*), assinaturas digitais, canais escondidos (*covert channels*), comunicações por espalhamento de espectro (*spread spectrum communications*) entre outras. Neste âmbito, o principal objetivo deste trabalho foi desenvolver um produto de *software* capaz de permitir a comunicação segura pela *internet* por fazer uso de técnicas *esteganográficas* em imagens digitais.

Camaleão: a Digital Security Software Using Steganography

Digital protection is a research area which needs efficient ways to make it possible. The *steganography* is configured as one of these eletronic protection way. It includes a set of methods for private communications such as *invisible inks, micro-dots, character arrangement, digital signatures, covert channels* and *spread spectrum communications*. So, main objective of work was to develop a software that allow security communication on the internet by using *steganographic* techniques in digital images.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A busca por novos meios eficientes e eficazes de proteção digital é um campo de pesquisa fundamentado nos mais variados campos da ciência. Basicamente, este campo de pesquisa se divide em duas ramificações. De um lado, estão aqueles que buscam técnicas para obter maior proteção digital. Do outro lado, estão aqueles que querem minar a proteção, isto é, querem ter acesso à informação sem autorização.

Uma das áreas que tem recebido muita atenção recentemente é a *esteganografia*. Esta é a arte de mascarar informações como uma forma de evitar a sua detecção. Segundo [Popa, 1998], *esteganografia* deriva do grego, donde *estegano = esconder, mascarar* e *grafia = escrita*. Logo, *esteganografia* é a arte da *escrita encoberta* ou, de forma mais abrangente, é a arte das comunicações encobertas.

A *esteganografia* inclui um vasto conjunto de métodos para comunicações secretas desenvolvidos ao longo da história. Dentre tais métodos estão: tintas "invisíveis", micro-pontos, arranjo de caracteres (*character arrangement*), assinaturas digitais, canais escondidos (*covert channels*), comunicações por espalhamento de espectro (*spread spectrum communications*), entre outras.

Atualmente, trabalha-se na estruturação e no desenvolvimento da *esteganografia digital*. Esta consiste em um conjunto de técnicas e algoritmos capazes de permitir uma comunicação digital mais segura em um tempo em que *e-mails* podem estar sendo lidos e computadores pessoais rastreados. Estas técnicas podem variar desde a inserção de imagens em outras — fazendo com que uma imagem aparentemente inocente esconda outra com maior importância sem levantar suspeitas — até a escrita de textos inócuos que escondem algum texto secreto em sua estrutura. Tais técnicas também estão presentes nos modernos equipamentos militares que fazem transmissões de rádio e codificam em ondas-curtas mensagens mais importantes.

Este súbito interesse pela *esteganografia* deve-se, também, à busca por técnicas de *copyright* eficientes e eficazes. A partir do momento em que áudio, vídeo e outras formas de comunicação de mensagens tornaram-se disponíveis em formatos digitais, a facilidade com que qualquer um destes possa ser perfeitamente copiado aumentou exponencialmente. Isto está levando a uma imensa quantidade de reproduções destas formas de comunicação de mensagens não autorizadas pelo mundo todo. Como contra-medidas, técnicas avançadas de "marcas-d'água" (*watermarking*) ou mesmo técnicas de seriação (*fingerprinting*), estruturadas na *esteganografia*, buscam restringir a pirataria indiscriminada.

A proposta do trabalho foi estudar as principais técnicas de *esteganografia* da atualidade, embasadas ou não nas técnicas clássicas, e evidenciar seus impactos na sociedade como um todo. Como resultado, foi criado o **Camaleão**, um *software* para proteção digital que faz uso de técnicas *esteganográfico-digitais*. Deste modo, quaisquer interessados poderão ter um conhecimento ilustrado desta nova área.

1.1 Motivação

Há uma enorme quantidade de aplicações para a *esteganografia* e para o chamado *mascaramento digital de dados*. Dentre as diversas utilidades, pode-se destacar:

- agências militares e de inteligência precisam de comunicações reservadas. Mesmo se o conteúdo é criptografado, a detecção de um sinal nos modernos campos de batalha pode levar rapidamente a identificação e ataque aos remetentes e destinatários. Por esta razão, os militares utilizam técnicas de espalhamento de espectro e modulação;

- a justiça e as agências de inteligência estão interessadas em conhecer estas tecnologias e suas fraquezas, assim como detectar e rastrear mensagens escondidas;

- tentativas recentes de alguns governos, por exemplo o dos EUA, de limitar os usos da criptografia têm estimulado as pessoas a buscar meios alternativos para garantir suas comunicações anônimas e seus direitos à liberdade de expressão [Wallich, 2003];

- esquemas para eleições digitais e dinheiro eletrônico precisam fazer uso de técnicas de comunicação anônimas.

Assim sendo, a *esteganografia* pode aumentar a privacidade individual. Esta não vem para substituir a criptografia. Vem, em contrapartida, para complementá-la. Os poderes da segurança digital podem aumentar consideravelmente quando,

ao transmitir uma mensagem, esta for criptografada e, em seguida, *esteganografada*. Por quê? Imagine a dificuldade em quebrar um código ao qual não se sabe, ao menos, de sua existência.

1.2 Objetivos

O trabalho visou atender aos seguintes objetivos:

- propiciar um maior contato com as principais técnicas de proteção digital e, em especial, as técnicas de *esteganografia*;
- estudar técnicas clássicas de *esteganografia* e suas contribuições para as modernas técnicas *esteganográfico-digitais*;
- pesquisar técnicas *esteganográfico-digitais* existentes atualmente;
- analisar o desempenho de tais técnicas e seu aproveitamento real como meio de proteção digital;
- identificar as vantagens e as desvantagens de tais técnicas;
- desenvolver um produto de *software* para acompanhar o funcionamento de algumas técnicas *esteganográficas*. Implementar pelo menos três variações destas técnicas;
- disponibilizar todo o material bibliográfico utilizado para o desenvolvimento da pesquisa. Desta forma, há a divulgação dos estudos de privacidade e proteção digital, bem como a situação corrente do trabalho. Para isso, foi construída uma página (*site*) e disponibilizada na rede mundial de computadores (*internet*);
- divulgar mais este tema, que, certamente, não sairá das mídias informativas nos próximos anos.

1.3 Metodologia

O trabalho foi desenvolvido utilizando os materiais e os métodos descritos a seguir:

- foi realizado um levantamento bibliográfico, na *internet* e em bibliotecas, de artigos científicos clássicos e atuais relacionados ao tema;

- em paralelo, foi realizado um estudo do que seria a *esteganografia*, propriamente dita. Isto foi feito através de uma análise detalhada do material coletado;
- também em paralelo, foi realizado um estudo sobre os impactos da *esteganografia* no mundo. As mudanças que estão ocorrendo, o que está e o que não está sendo afetado entre outras;
- findas estas etapas, foram encaminhados estudos das técnicas *esteganográficas* clássicas e as suas contribuições para os modernos sistemas *esteganográficos* atuais;
- feito isso, estudou-se algumas técnicas *esteganográfico-digitais*. Estas são o estado da arte da *esteganografia*;
- após estes estudos preliminares, iniciou-se um estudo de como seriam implementadas, computacionalmente, tais técnicas servindo como ferramenta didática a futuros interessados;
- implementou-se o **Camaleão**, uma vez que as suas formas já foram definidas. A preocupação de construir códigos-fonte manuteníveis foi constante. O paradigma de programação utilizado foi a orientação a objetos e a linguagem de programação foi Java [Sun Microsystems, 2003] devido a alguns aspectos intrínsecos considerados importantes, por exemplo, a portabilidade entre sistemas operacionais [Deitel and Deitel, 2001];
- terminada a implementação, passou-se à etapa de testes em laboratório com o uso de exemplos práticos;
- uma vez que o produto de *software* estava funcionando satisfatoriamente, passou-se para a fase de finalização onde foi desenvolvida uma documentação e posterior divulgação na *internet*;

1.4 Descrição dos capítulos posteriores

A seguir, é apresentada uma descrição suscinta dos capítulos deste trabalho. O capítulo 2 apresenta os principais termos utilizados. O capítulo 3 mostra uma retrospectiva da *esteganografia* desde os seus primórdios até os dias atuais. Em seguida, o capítulo 4 apresenta as justificativas deste trabalho e os principais impactos sociais do mascaramento digital de informações. O capítulo 5 discorre sobre as principais técnicas *esteganográficas* da atualidade e algumas perspectivas de robustez. O capítulo 6 mostra as principais técnicas de *esteganálise*. O capítulo 7 apresenta

os resultados desta pesquisa em relação ao campo da *esteganografia*. O capítulo 8 mostra os resultados referentes à *esteganálise*. Finalmente, o capítulo 9 apresenta as principais conclusões referentes ao trabalho bem como algumas propostas de trabalhos futuros e algumas contribuições.

Capítulo 2

Terminologia

2.1 Considerações iniciais

Como já dito, há um interesse cada vez maior, por diferentes comunidades de pesquisa, no campo da *esteganografia*, marcas d'água e seriação digitais. Com certeza, isso leva a uma certa confusão na terminologia. A seguir, encontra-se um estudo dos principais termos utilizados nestas áreas. É importante salientar que estas definições ainda não são totalmente aceitas, podendo existir pequenas variações na literatura.

2.2 Definições

Segundo [Petitcolas et al., 1999], o modelo geral de ocultamento de dados (*information hiding*) pode ser descrito como se segue. O dado embutido (*embedded data*) é a mensagem que se deseja enviar de maneira secreta. Freqüentemente, este dado é escondido em uma mensagem inócua (sem maior importância) conhecida como mensagem de cobertura (*cover-message*). As mensagens de cobertura podem variar de nome de acordo com o meio de cobertura sendo utilizado. Deste modo, pode-se definir uma imagem de cobertura (*cover-image*), áudio de cobertura (*cover-audio*) ou texto de cobertura (*cover-text*). Após o processo de inserção dos dados na mensagem de cobertura, obtém-se o chamado estego-objeto (*stego-object*), uma mensagem inócua contendo secretamente uma mensagem de maior importância. A figura 2.1 apresenta como o processo pode ser interpretado. Um indivíduo escolhe o dado a ser escondido e, a partir de uma chave, mascara estes dados em uma imagem de cobertura previamente selecionada. O resultado é a estego-imagem a ser enviada.

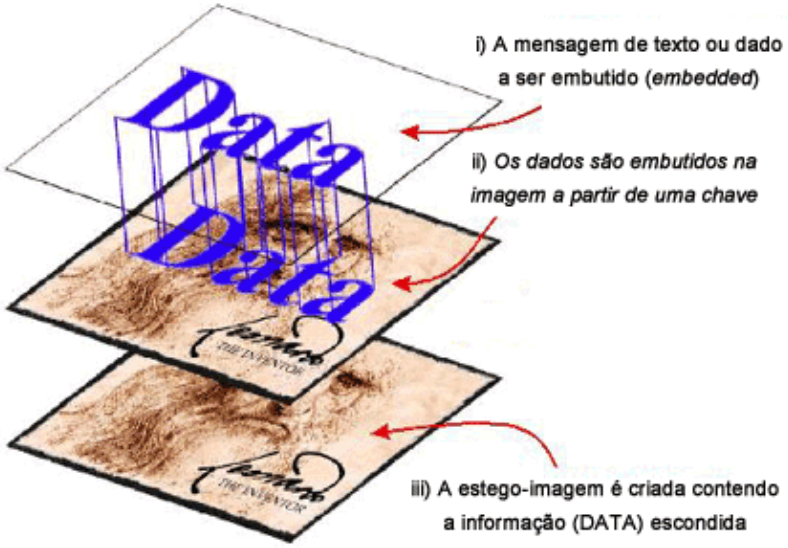

Figura 2.1: Exemplo de ocultamento de uma mensagem

Uma estego-chave (*stego-key*) é utilizada para controlar o processo de ocultamento de forma a restringir a detecção e/ou recuperação dos dados do material embutido.

Parafraseando [Petitcolas et al., 1999], um ataque com sucesso à *esteganografia* consiste em conseguir detectar a existência de uma mensagem escondida em algum meio observado. Por outro lado, os sistemas de marcação de *copyright* ou *watermarking* têm requisitos adicionais de robustez contra possíveis ataques. Deste modo, um ataque bem-sucedido consiste em conseguir detectar e remover a marcação digital.

Continuando, o sistema de seriação digital (*fingerprinting*), também conhecido como *labels*, consiste de uma série de números embutidos no material a ser protegido. Isto permite identificar, por exemplo, que um cliente quebrou um acordo de propriedade intelectual.

Finalmente, pode-se delimitar a grande-área de pesquisa conhecida como ocultamento da informação (*information hiding*) como apresentado hierarquicamente na figura 2.2.

No segundo nível da hierarquia têm-se: *canais abertos*, *esteganografia*, *anonimato* e *marcação de copyright*.

Entende-se por *canais secretos*, a criação de uma comunicação entre duas partes em que o meio é secreto e seguro. Um exemplo seria as conversações militares

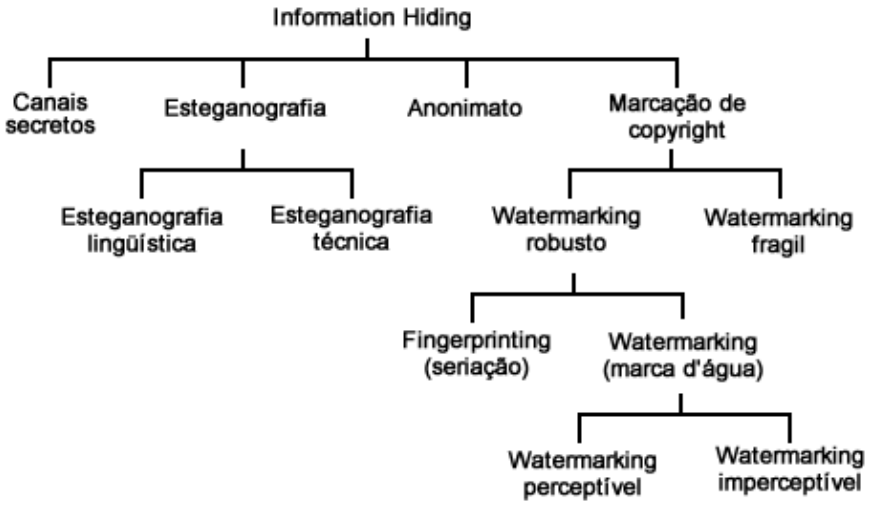

Figura 2.2: A hierarquia do *information hiding* [Pfitzmann, 1996]

em faixas de freqüências moduladas.

Continuando, a arte da *esteganografia* constitui a segunda ramificação da hierarquia. Pode ser dividida em *lingüística* e *técnica*. Quando a mensagem é fisicamente escondida, tal como a maioria dos exemplos que serão apresentados no capítulo 3, configura-se a chamada *esteganografia técnica*. Por outro lado, quando a mensagem é trabalhada e o seu ocultamento depende de propriedades lingüísticas, tal como a *esteganografia digital*, configura-se a chamada *esteganografia lingüística*.

Anonimato é um conjunto de técnicas para tentar navegar na *internet*, por exemplo, sem ser localizado. Isto poderia ser feito utilizando *sites* de desvio, por exemplo o `www.anonymizer.com`, e/ou *remailers* — *sites* capazes de enviar mensagens secretas não revelando seu remetente —.

Marcação de copyright é a tentativa de manter ou provar a propriedade intelectual sobre algum tipo de mídia, seja esta eletrônica ou impressa. Neste sentido, *sistemas de marcação robustos* (*watermarking robusto*) são aqueles que, mesmo após tentativas de remoção, permanecem intactos. Por outro lado, *sistemas de marcação frágeis* (*Watermarking frágil*) são aqueles em que qualquer modificação na mídia acarreta perda na marcação. Estes sistemas são úteis para impedir a cópia ilegal. Ao se copiar um material original, o resultado é um material não marcado e, por conseguinte, pirata. *Sistemas de marcação imperceptível* (*Watermarking imperceptível*) são aqueles em que as logomarcas dos autores, por

exemplo, encontram-se no material, mas não são diretamente visíveis. Em contrapartida, *marcação visível* (*Watermarking visível*) é aquela em que o autor deseja mostrar sua autoria a todos que observarem a sua criação. Um exemplo desta última forma é formado pelas imagens disponibilizadas na biblioteca do Vaticano `http://bav.vatican.va`. Segundo [Mintzer et al., 1996], nesta biblioteca as imagens possuem um sistema de marcação digital visível como pode ser observado na figura 2.3.

Figura 2.3: Exemplo de *marcação visível*. Biblioteca do Vaticano

2.3 Considerações finais

O campo de mascaramento digital de dados está em constante evolução. Muitos outros termos podem ser criados, modificados ou mesmo deixarem de existir repentinamente. Aqueles que foram aqui citados têm por objetivo facilitar o entendimento do trabalho desenvolvido e sua respectiva classificação na hierarquia.

Capítulo 3

Análise histórica

3.1 Considerações iniciais

Este capítulo apresenta alguns aspectos históricos relacionados à arte das comunicações em segredo. Ao final, apresentam-se alguns aspectos atuais ligados à *esteganografia digital*.

3.2 Aspectos históricos

Através de toda a história, as pessoas têm tentado as mais inúmeras formas de esconder informações dentro de outros meios, buscando, de alguma forma, mais privacidade para seus meios de comunicação. Duas excelentes fontes podem ser encontradas em [Kuhn, 1996] e [Norman, 1980].

Um dos primeiros registros sobre *esteganografia* aparece em algumas descrições de Heródoto, o pai da História, com vários casos sobre sua utilização. Um deles conta que um homem, de nome Harpagus, matou uma lebre e escondeu uma mensagem em suas entranhas. Em seguida, ele enviou a lebre através de seu mensageiro que se passou por um caçador [Petitcolas et al., 1999].

Em outro caso, no século V AC, um grego de nome Histaieus, a fim de encorajar Aristágoras de Mileto e seus compatriotas a começar uma revolta contra Medes e os Persas, raspou a cabeça de um de seus escravos mais confiáveis e tatuou uma mensagem em sua cabeça. Assim que os seus cabelos cresceram, o escravo foi enviado à Grécia com instruções de raspar sua cabeça permitindo aos seus amigos receberem a mensagem [Petitcolas et al., 1999].

Outra técnica bastante utilizada através da História foi o uso de tabletes de madeira cobertos de cera. Estes tabletes serviam como meio de escrita para a época, Grécia Antiga. Os textos eram escritos sobre a cera e, quando se tornavam inúteis,

a cera era derretida e uma nova camada de cera era colocada sobre a madeira. Isto gerava outro tablete de cera novo e pronto para a escrita. Seguindo esta idéia, Heródoto conta que Demeratus, um grego exilado na corte persa, ficara sabendo que o rei da Pérsia, Xerxes, o Grande, estava planejando invadir seu país natal. Movido de sentimentos de patriotismo, Demeratus resolveu encontrar um meio de avisar a corte grega sobre os planos audaciosos de Xerxes. A maneira encontrada foi utilizar os já famosos tabletes de cera. No entanto, ele não agiu pela forma normal em que se escrevia nos tabletes. Ao invés de escrever na cera sobre a madeira, o que tornaria seu texto visível a todos, tal como se fosse um texto em folha de papel atualmente, Demeratus derreteu toda a cera, escreveu a mensagem na própria madeira e depois a recobriu com cera novamente como se estivesse construindo um tablete de cera novo. Este procedimento fez com que o texto na madeira ficasse encoberto pela cera. Os tabletes foram enviados como se fossem tabletes em branco para exportação. Passaram sem problemas na fronteira persa e chegaram em tempo na Grécia. Contudo, como ninguém na Grécia sabia do procedimento do emissor da mensagem, os tabletes ficaram um bom tempo sem serem decifrados. Isto prosseguiu até que uma mulher grega de nome Gorgo, desconfiada dos tais tabletes, resolveu derreter a cera. Com isso, Gorgo tornara-se a primeira mulher criptoanalista da história e a corte grega fora salva pela engenhosa idéia de Demeratus [Kahn, 1996].

Outro relato interessante vem do grego Enéas, o Tático, escritor de várias matérias militares. Ele inventou uma técnica *esteganográfica* intitulada *astrogal*. O *astrogal* consistia em uma madeira com vários furos, cada qual representando uma letra. Quando alguém desejasse enviar uma mensagem, este deveria passar um barbante pelos furos correspondentes às letras da mensagem a ser transmitida. Cabia ao receptor da mensagem acompanhar as várias ligações de pontos feitas pelo barbante e, assim, decifrar a mensagem. Quando era interceptado por alguém, este engenho era tido apenas como um brinquedo de criança. Dois mil anos mais tarde, remetentes ingleses empregaram o mesmo método, não para garantir o segredo de suas cartas, mas para evitar o pagamento de taxas muito caras. Na realidade, antes da reforma do serviço postal em meados de 1850, enviar uma carta custava cerca de um *shilling*[1] para cada cem milhas de distância. Os jornais, no entanto, eram isentos de taxas. Graças a furinhos de agulha, os espertos ingleses enviavam suas mensagens gratuitamente. Este procedimento foi utilizado também pelos alemães durante a *Primeira Guerra Mundial* [Petitcolas et al., 1999].

Durante a Renascença, Giovanni Porta, um dos maiores criptoanalistas de seu tempo, "aperfeiçoou" a técnica da lebre de Harpagus. A proposta de Porta era

[1]Uma das divisões da moeda americana no século XIX. Equivalente à vigésima parte de uma libra esterlina.

alimentar um cachorro com a mensagem. Desta forma, o cachorro seria enviado como seu portador. O receptor ao recebê-lo, o mataria e recuperaria a mensagem. Porta também descobriu como esconder uma mensagem em um ovo cozido. Para tal, basta escrever sobre a casca com uma tinta contendo uma onça de alume ($\pm$ 29 g) diluído em cerca de meio litro de vinagre. Com isso, a solução penetra a casca e se deposita sob esta. Depois, basta abrir o ovo para ler a mensagem [Singh, 2001].

Segundo [Kahn, 1996], Porta é bastante conhecido no campo da comunicação secreta. Também é sua a criação da famosa *cifra indecifrável* (*Le chiffre indéchiffrable*) [Singh, 2001], um dos primeiros sistemas de criptografia por substituição polialfabética. A figura 3.1 apresenta este famoso trabalho de Porta.

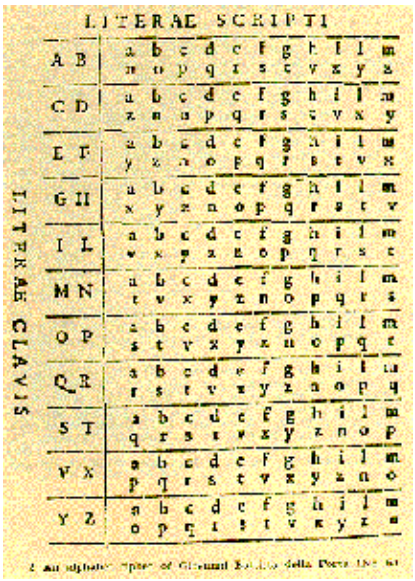

Figura 3.1: A cifra polialfabética de Porta

Ainda nesta época, Johannes Trithemius, um abade alemão, publicou uma trilogia em latim intitulada *Steganographia: hoe est ars per occultam scripturam animi sui voluntatem absentibus aperiendi certa*. No terceiro volume desta obra, Trithemius escondeu o Salmo 23 da Bíblia Sagrada através da utilização de algumas tabelas contendo números. Os escritos foram descobertos apenas no século XX devido aos esforços dos pesquisadores Thomas Ernst, da Universidade de Pittsburg, e Jim Reeds, do AT&T Labs [Kolata, 2003].

Outra técnica interessante que aparece durante a História faz uso de inúmeras variações de tintas "invisíveis" (*invisible inks*). Tais tintas não são novidades e

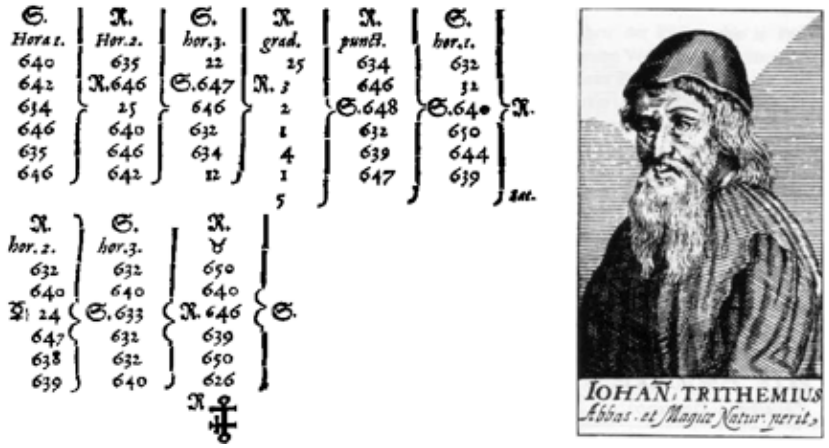

Figura 3.2: Trithemius e uma das tabelas encontradas em *Steganographia*

já apareciam em relatos de Plínio, o Velho, e Ovídio no século I DC. Ovídio, em sua *Arte do amor*, propusera o uso do leite para escrita de textos "invisíveis". Para decodificar a mensagem, o receptor deveria borrifar o papel com ferrugem ou carbono negro. Estas substâncias aderiam ao leite e a mensagem era revelada [Kuhn, 1996] e [Kahn, 1996].

As primeiras tintas eram simples fluidos orgânicos que não exigiam nenhuma técnica especial para serem reveladas. Algumas vezes, bastava apenas aquecer o papel e a mensagem aparecia. Isto pode ser confirmado com as tintas baseadas em fluidos de suco de limão, por exemplo.

No entanto, durante a primeira guerra mundial, espiões alemães colocavam pequenos "pontos" de tinta invisível sobre letras de revistas e jornais de grande circulação. As folhas de revistas "pontuadas", quando aquecidas, revelavam a seqüência das letras e, por conseguinte, toda a mensagem ali escondida [Kuhn, 1996].

Suspeitando de atividades semelhantes às dos alemães na primeira grande guerra, os americanos recrutaram inúmeros profissionais qualificados durante a guerra-fria. O objetivo era examinar minuciosamente as principais publicações impressas em circulação no país em busca de mensagens secretas dos soviéticos.

Como resultado do progresso global da ciência, outras formas mais poderosas de tintas invisíveis foram aparecendo através da história. De forma geral, as tintas invisíveis são químicas que, misturadas a outras químicas, tornam o resultado visível. Alguns historiadores tais como [Kahn, 1996] e [Singh, 2001] mencionam o uso de tais químicas desde os tempos clássicos. Uma delas é o uso do ácido galotânico — feito a partir de nozes —. Mensagens escritas com ácido galotânico

se tornam visíveis apenas em contato com sulfato de cobre.

Um exemplo mais próximo dos tempos contemporâneos foi aplicado pelo espião nazista George Dasch, na segunda guerra mundial. Dasch escreveu mensagens em seu lenço utilizando uma solução de sulfato de cobre. A mensagem poderia ser decodificada utilizando vapor de amônia [Kuhn, 1996].

Durante as duas guerras mundiais, os químicos tinham de estudar várias formas possíveis e imagináveis de combinações químicas das mais diversas substâncias. Estas seriam para esconder ou mesmo para descobrir mensagens e criar procedimentos-padrão de detecção para censores nas fronteiras. Estes tinham que utilizar inúmeras escovas sobre mensagens interceptadas, borrifar uma enorme combinação de químicas, entre outras coisas, objetivando descobrir mensagens secretas ali colocadas utilizando-se tintas invisíveis [Singh, 2001].

De acordo com [Johnson and Jajodia, 1998], outros exemplos, através da história, aparecem no campo da fotografia. Devido aos inúmeros avanços neste campo, com um grande aumento na qualidade das fotos, bem como uma sucessiva redução em seus tamanhos, tornou-se possível reduzir fotos de páginas inteiras de texto a tamanhos consideráveis. Uma aplicação desta técnica de redução aconteceu na guerra franco-prussiana. Quando Paris estava sitiada pela Prússia, seus habitantes escreviam mensagens e fotografavam-nas. Em seguida, reduziam ao máximo os negativos. Utilizando-se de pombos-correio, enviavam as mensagens para fora de Paris, conseguindo estabelecer um canal de comunicação com os arredores da cidade sitiada.

Na segunda guerra mundial, com um aumento na qualidade das câmeras, lentes e filmes, tornou-se possível aos espiões nazistas, a criação de uma das formas mais interessantes e engenhosas de comunicação secreta. As mensagens nazistas eram fotografadas e, posteriormente, reduzidas ao tamanho de pontos finais (**.**) em uma sentença. Assim, uma nova mensagem totalmente inocente era escrita contendo o filme ultra-reduzido como final das sentenças. A mensagem gerada era enviada sem levantar maiores suspeitas. Esta engenhosidade ficou conhecida como *tecnologia do micro-ponto* [Singh, 2001].

Outras formas clássicas de comunicação secreta são os *semagramas* e os *códigos abertos* [Kahn, 1996].

Semagramas são formas de comunicação secreta que não estão na forma escrita. A utilização dos semagramas também pode ser encontrada na segunda guerra mundial. Como narra [Kahn, 1996], em certa ocasião, os censores americanos interceptaram um carregamento de relógios e mudaram toda a sua disposição na caixa, bem como a de seus ponteiros. Havia o medo de que disposição dos ponteiros e dos relógios escondesse alguma mensagem secreta.

Por outro lado, *códigos abertos* fazem o uso da ilusão ou de palavras có-

digo. Como exemplo, têm-se as ações de Vallerie Dickinson — uma espiã a serviço do Japão na segunda grande guerra — que usava vestidos de bonecas para avisar aos japoneses sobre ações americanas. Pequenos vestidos representavam *destroyers* e grandes vestidos poderiam representar *couraçados* ou *cruisers* [Petitcolas et al., 1999].

Cifras nulas (*null ciphers*) também foram muito utilizadas. De acordo com [Johnson and Jajodia, 1998], através desta técnica uma mensagem é escondida dentro de outra aparentemente inocente. Um exemplo clássico é a obra *Hypnerotomachia Poliphili* de 1499. Neste livro, um padre, de nome Colona, codificou a mensagem "Padre Colona ama Polia apaixonadamente" (*Father Colona Passionately loves Polia*) em cada primeira letra de um novo capítulo. A Igreja Católica não tolerou o abuso e, quando tempos depois, decifrou a mensagem, condenou o padre à morte. Um exemplo mais claro de *cifras nulas* pode ser encontrado a seguir. Este texto é uma cifra nula enviada por um espião alemão na segunda grande guerra.

> *A**p**parently n**e**utral's p**r**otest i**s** t**h**oroughly d**i**scounted a**n**d i**g**nored. Is**m**an h**a**rd it. B**l**ockade i**s**sue a**ff**ects p**r**etext f**o**r e**m**bargo o**n** b**y**-products, e**j**ecting s**u**ets a**n**d v**e**getable o**i**ls.*

A mensagem codificada pode ser extraída pela captura de toda segunda letra de cada palavra. Isto resulta em:

Pershing sails from NY June 1.

Girolammo Cardano também dá uma grande contribuição à *esteganografia* através de seu engenho conhecido como "grelha de cardano". Tal técnica consistia em um papelão com furos em locais estratégicos. Tanto o emissor quando o receptor, em posse de uma grelha dessas poderia se comunicar colocando-a sobre uma grande quantidade de texto e apenas apareceriam as palavras sob os furos da grelha [Singh, 2001].

Um exemplo mais atual de *esteganografia* pode ser encontrado no governo da primeira-ministra britânica Margareth Tatcher nos anos oitenta. Desconfiada de que alguns de seus ministros não lhe eram mais leais, a ministra ordenou à sua casa civil que codificasse os principais documentos do governo com *códigos de deslocamento de linha* de modo que não se pudesse falsificá-los. Infelizmente, a primeira-ministra não teve êxito e a informação vazou para a imprensa. Seus planos tiveram de ser estrategicamente reconsiderados [Singh, 2001].

De acordo com [Judge, 2001], em certas ocasiões, os emissores não possuem o interesse em esconder as mensagens. No entanto, se todos aqueles que são capazes de entendê-la deixarem de existir a mensagem torna-se, de alguma forma,

escondida dado que não há mais quem a decifre. Neste sentido, pode-se citar os "geoglifos" do platô de Nazca no Peru. Um exemplo encontra-se na figura 3.3. Estes foram decifrados recentemente a partir de uma vista aérea. Atualmente, tra-

Figura 3.3: Um "geoglifo" no platô de Nazca, Peru.

ficantes de drogas escondem "papelotes" dentro de seus corpos, engolindo-os. São as chamadas "mulas" (jargão policial). Isto remonta à técnica de Giovanni Porta no Renascimento.

No Brasil, até meados da década de 80, algumas provas de concursos públicos eram corrigidas utilizando-se cartões perfurados semelhantes à técnica da grelha inventada por Girolammo Cardano. Quando postas sobre os cartões-respostas dos candidatos, revelavam se o candidato havia acertado ou errado as questões da prova.

3.3 Considerações finais

Atualmente, a *esteganografia* não foi esquecida. Ela foi modificada em sinal de acompanhamento dos novos tempos. Na era da informação, não faz mais sentido escrever textos em tabletes de madeira ou mesmo borrifar pontos em uma revista através da utilização de tintas "invisíveis". Qualquer meio de *esteganografia* na atualidade, inevitavelmente, deve utilizar meios contemporâneos de tecnologia. Embora, em alguns casos, estes meios sejam apenas aperfeiçoamentos de técnicas clássicas.

Neste sentido, várias pesquisas têm sido feitas no campo da *esteganografia digital*. Existe um grande número de documentos digitais disponíveis na *internet*. E, em muitas ocasiões, as pessoas desejam trocar informações de forma rápida e segura. De acordo com [Kumagai, 2003], [Cass, 2003] e [Wallich, 2003], acontecimentos recentes, como o atentado terrorista ao *World Trade Center* em 11 de setembro de 2001, fizeram com que as autoridades passassem a "vigiar" tudo o que circula de forma criptografada ou não pela grande rede. Isto quer dizer que, se antes uma mensagem criptografada poderia passar despercebida, agora ela pode ser interpretada como uma mensagem de alguém suspeito que tem algo a esconder. Em meio a toda esta paranóia, a *esteganografia* vem ganhando grande destaque e conquistando seu espaço.

Outra razão pela qual a *esteganografia digital* vem ganhando destaque na mídia deve-se aos estudos de *copyright* e *watermarking* de documentos eletrônicos. À medida que aumenta a pirataria pela rede mundial de computadores, novos meios mais eficientes e eficazes de proteção intelectual são estudados no intuito de conter as cópias não-autorizadas.

Como será explicado no capítulo 5, há inúmeras formas de esteganografia digital atualmente. Pode-se utilizar imagens, sons, textos, entre outros, como meios de cobertura para mensagens a serem transmitidas.

Capítulo 4

Impactos sociais

4.1 Considerações iniciais

Este capítulo tem a finalidade de apresentar os principais impactos sociais causados pelo uso da *esteganografia* digital. São tratados importantes aspectos relacionados ao tema, bem como os usos legais e ilegais do mascaramento de informações.

4.2 Contextualização

O termo *esteganografia* ganhou muito destaque nas mídias informativas após o ataque terrorista ao *World Trade Center* em 11 de setembro de 2001 [Wallich, 2003], [Cass, 2003], [Kumagai, 2003].

Muito se falou, após este lastimável acontecimento, que o grupo terrorista *Al Qaeda*, liderado pelo milionário saudita Osama bin Laden, estaria se comunicando com suas células espalhadas pelo mundo através de mensagens escondidas em imagens digitais. Tais imagens estariam sendo distribuídas através de *chats*, grupos de discussão, *e-mails*, leilões eletrônicos entre outros meios.

Praticamente, dois anos depois nada foi comprovado. Este é justamente o grande trunfo da *esteganografia*. Uma vez que as mensagens tenham sido muito bem-escondidas, não se consegue recuperá-las. Quando muito, desconfia-se de sua presença.

Deste modo, a *esteganografia* apresenta-se como uma tecnologia apta a auxiliar as pessoas a aumentarem sua privacidade *on-line*. Juntamente com a criptografia, os cidadãos têm em mãos uma forma robusta e altamente eficiente para manter suas informações íntegras e protegidas. No entanto, este alto grau de sigilo preocupa as autoridades políticas e policiais. Uma vez que as comunicações estejam

seguras, qualquer indivíduo de um país pode elaborar em segredo os mais mirabolantes planos que desejar. Estes planos podem muito bem tratar-se de um atentado terrorista e/ou ameaçar a segurança nacional de um país. Segundo [EFF, 2003], muitas propostas de controle de privacidade já existem ou estão em andamento tais como:

- *PATRIOT*[1];
- *Carnivore*[2];
- *DMCA*[3];
- *CAPPS II*[4];

Tem-se, atualmente, uma visão deturpada de que segurança e privacidade são termos antagônicos. Isto é, caso as pessoas não possam ser vigiadas, elas representam perigo ou para outras pessoas ou para o país.

Foi neste contexto que os EUA, assinaram em outubro de 2001, o *Ato Anti-Terrorismo PATRIOT*, um verdadeiro ato de desrespeito à liberdade invividual de qualquer cidadão do mundo segundo a EFF (*Eletronic Frontier Foundation*). Neste ato, o presidente americano George W. Bush, através de seu imenso poder à frente da nação mais poderosa da terra, impõe políticas, agora legítimas, que dão às autoridades americanas o direito de espionar qualquer cidadão sem aviso prévio ou posterior em nome da segurança nacional norte-americana [EFF, 2003].

Atitudes como essa impulsionam os cidadãos a buscarem meios eficientes e eficazes de se protegerem das "insanidades" de seus governos. Mais uma vez, a *esteganografia* apresenta-se como uma poderosa ferramenta de auxílio.

Inúmeras outras formas de abuso e espionagem para com os cidadãos podem ser citadas. Poucos sabem, mas a maioria do conteúdo em circulação pela *Internet* é constantemente vigiado pelo projeto *Echelon*. Este projeto visa filtrar toda a informação em circulação pela rede em busca de terroristas. O projeto existe em uma associação dos países EUA, Reino Unido, Austrália e Canadá. Outro projeto igualmente terrível é o *UKUSA*. O termo é uma justaposição das siglas UK (*United Kingdom*) e USA (*United States of America*). Este projeto visa filtrar toda e qualquer informação em nome da segurança nacional dos países envolvidos. O *Echelon* e o *UKUSA* são dois atentados contra a liberdade de expressão dos cidadãos.

[1] Provide Appropriate Tools Required to Intercept and Obstruct Terrorism
[2] Programa do FBI para vigiar o correio eletrônico.
[3] Digital Milenium Copyright Act
[4] Computer Assisted Passenger Pre-Screening System

Infelizmente, como toda a tecnologia pode ser aplicada para o bem e/ou para o mal, com a *esteganografia* não é diferente. Nas seções a seguir, encontram-se os principais usos legais e ilegais deste poderoso campo de pesquisas.

4.3 Usos legais

O mascaramento de informações tem uma grande variadade de usos legais em produtos e protocolos. Pequenas mudanças ou combinação de vários algoritmos criam diferentes ferramentas com diferentes usos. Seguem algumas das aplicações mais interessantes:

- *estruturas de dados aprimoradas:* informações podem ser escondidas de modo a funcionarem como estruturas de dados avançadas. Seria possível armazenar informações médicas a respeito de um paciente em seu próprio raio X. Fotos de satélite poderiam, também, armazenar dados geográficos sobre os locais observados;
- *marcas d'água resistentes:* os criadores de conteúdo digital tais como livros, filmes e arquivos de áudio querem adicionar informações escondidas em suas criações de modo a garantir a sua propriedade. Vários tipos de marcações (*watermarks*) podem ser feitos. Desde aquelas que permanecem no documento até que ele seja totalmente destruído às que desaparecem após qualquer tipo de modificação no meio em que se encontra;
- *rastreamento de documentos:* informações escondidas podem identificar os verdadeiros donos de um determinado conteúdo digital (*fingerprinting*). Deste modo, um documento pode facilmente ser classificado como pirateado ou não;
- *autenticação de documentos:* pode-se inserir uma assinatura de autenticidade em documentos digitais. Deste modo, um *software* apenas irá executar e/ou mostrar o conteúdo do documento caso ele seja autêntico;
- *comunicações privadas:* o mascaramento de informações pode ser utilizado para estabelecer comunicações seguras entre os cidadãos através da rede mundial de computadores (*internet*).

4.4 Usos ilegais

Como nem tudo é perfeito, a *esteganografia* também se configura como um poderoso campo de pesquisas que pode ser aplicado com intenções ilegais. Segundo o

relatório anual sobre crimes de alta tecnologia (*High Technology Crimes Annual Report*) [USPS, 2003] e [NHTCU, 2003], as chamadas *dez bestas do tecnocalipse* poderiam fazer amplo uso da *esteganografia* como forma de ampliar seu poder, a saber:

- fraudes e lavagem de dinheiro;
- comunicações criminais;
- roubo de propriedade intelectual;
- *hacking*;
- desvio de dinheiro;
- jogos e pornografia;
- assédio e extorsão;
- disseminação de vírus;
- pedofilia;
- tráfico de pessoas.

Todos estes meios podem ser combinados de inúmeras formas diferentes. Um *hacker* com um mínimo de habilidade poderia esconder um vírus com alto poder de destruição dentro de uma imagem aparentemente inocente. Um indivíduo que obtivesse a imagem não desconfiaria de nada. Posteriormente, o *hacker*, valendo-se de falhas de segurança nos sistemas-alvo, poderia ativar o vírus. Com este tipo de vírus, operações sofisticadas poderiam ser feitas, tais como: capturar senhas de acesso, passar-se pelo indivíduo dono do sistema invadido, desviar dinheiro, entre outras coisas.

Além disso, relatos recentes da *Unidade Nacional de Crimes High-Tech*, Reino Unido, afirmam que pedófilos estariam compartilhando imagens pela *internet* através do seu mascaramento em imagens aparentemente inocentes. Uma operação de varredura pela rede apontou para uma irmandade pedófila inglesa que também atuava na maior parte dos países europeus, [NHTCU, 2003].

4.5 Considerações finais

Este trabalho versa sobre tecnologia e tecnologia é neutra. O trabalho ensina como lançar fórmulas aparentemente mágicas para fazer com que informações sejam

mascaradas. Como isto será utilizado? Não se sabe. Aos cientistas, cabem apenas mostrar “como” e não “para que” usar as descobertas.

Capítulo 5

Técnicas de esteganografia

5.1 Considerações iniciais

Este capítulo apresenta as principais técnicas de *esteganografia digital* presentes na atualidade. Logo após, são apresentadas técnicas para mascaramento em imagens, formas de aumentar a robustez e, finalmente, outros meios em que se poderia usar o mascaramento de mensagens.

5.2 Visão geral

De acordo com [Popa, 1998], os principais algoritmos de *esteganografia digital* são baseados na substituição de componentes de ruído de um objeto digital por uma mensagem secreta pseudo-randômica.

Após o processo de embutir os dados, o estego-objeto gerado pode ser dividido em duas classes. Este pode ser um *stream cover* ou um *random access cover*. O primeiro é formado por um *stream* de dados contínuos como, por exemplo, uma transmissão telefônica. O último pode ser um arquivo do formato ".WAV" [Aura, 1996].

Comparativamente, tem-se que, utilizando-se técnicas de geração de *stream-covers*, não se pode identificar os tamanhos dos dados escondidos nem onde estes começam ou terminam no objeto de cobertura. A sua geração é feita a partir de um *keystream generator*, algo como uma chave de *criptografia* que diz em que ordem os bits devem ser inseridos e recuperados. Esta técnica é conhecida como *método do intervalo randômico* [Popa, 1998].

Por outro lado, os arquivos classificados como *random access cover* permitem ao emissor da mensagem colocar os dados em qualquer ordem no objeto de

cobertura, assim como é possível conhecer onde é o início e o fim da mensagem escondida.

Freqüentemente, os *bits* de cobertura são os menos significativos (*LSB — least significant bits*) do objeto de cobertura. Segundo [Popa, 1998], os *bits* menos significativos têm algumas propriedades estatísticas como a entropia e histograma. Mudanças em alguma destas propriedades poderiam resultar em perdas na qualidade do objeto de cobertura utilizado. Deste modo, a mensagem escondida precisaria "imitar", com grande estilo, os *bits* do objeto de cobertura. Uma possibilidade é gerar vários objetos de cobertura e, então, selecionar aquele com menor variação nas propriedades estatísticas dos *bits* menos significativos. Esta técnica é conhecida como *método da seleção* [Popa, 1998]. Outra possibilidade é gerar uma função chamada imitadora. Tal função teria o objetivo de modificar os *bits* da mensagem a ser escondida de forma que estes tenham a forma mais próxima possível dos *bits* do objeto de cobertura. Esta técnica é conhecida como *método construtivo* [Popa, 1998].

De forma geral, tanto o emissor quanto o receptor da mensagem compartilham uma chave secreta e a usam com um gerador de *streams* (*stream generator*) de modo a conseguir selecionar os vários locais do objeto de cobertura que serão utilizados para esconder a mensagem desejada.

Embora as técnicas baseadas em LSB consigam esconder os dados aos olhos humanos, elas podem ser facilmente destruídas computacionalmente utilizando algoritmos de compressão com perdas de dados (*lossy compression algorithms*). Estes algoritmos selecionam apenas as partes mais significativas do objeto de cobertura. Isto significa que os *bits* menos significativos têm uma chance menor de serem selecionados [Katzenbeisser and Petitcolas, 2000].

Outra forma de destruir os dados escondidos utilizando técnicas baseadas em LSB consiste em fazer pequenas alterações no objeto de cobertura utilizando filtros de baixa passagem (*low-pass filters*). Estes filtros são capazes de inserir modificações superficiais nos objetos de cobertura praticamente invalidando-os [Katzenbeisser and Petitcolas, 2000].

Uma forma para contornar tais ataques é esconder a mensagem em vários locais do objeto de cobertura. Além disso, a utilização de códigos de correção de erros (*CRCs — check redundance codes*) também se mostra uma solução eficaz.

Como descrito na seção 1.2, objetivou-se com o trabalho, implementar um produto de *software* capaz de trabalhar com algumas técnicas digitais de *esteganografia* baseadas em LSB. Foram escolhidas, essencialmente, as técnicas que utilizam imagens como sendo objetos de cobertura, podendo esconder textos, imagens e/ou quaisquer outros tipos de arquivos. As imagens foram escolhidas como meio de carregamento devido ao seu disseminado manuseio diário pela rede mundial de

computadores e pela relativa facilidade com que os usuários conseguem manipulá-las. A seguir, tem-se uma explicação mais detalhada sobre o processo de utilização de imagens como objetos de cobertura.

5.3 Técnicas de codificação em imagem

Informações podem ser escondidas de muitas maneiras diferentes utilizando imagens como meio de cobertura.

Segundo [Anderson and Petitcolas, 1998], a inserção de uma mensagem plana pode ser feita codificando cada *bit* de informação na imagem. Uma codificação mais complexa pode ser feita para encaixar a mensagem somente em áreas de ruído da imagem, i.e., aquelas em que haverá menor atenção. A mensagem pode também ser dispersa aleatoriamente em toda a superfície de ”ruídos” da imagem.

As abordagens mais comuns de inserção de mensagens em imagens incluem técnicas de:

- inserção no *bit* menos significativo;
- técnicas de filtragem e mascaramento;
- algoritmos e transformações.

Cada uma destas pode ser aplicada às imagens, com graus variados de sucesso. O método de inserção no *bit* menos significativo é provavelmente uma das melhores técnicas de *esteganografia* em imagem.

5.3.1 Inserção no bit menos significativo

Técnicas baseadas em LSB podem ser aplicadas a cada *byte* de uma imagem de 32-*bits*. Estas imagens possuem cada *pixel* codificado em quatro *bytes*. Um para o canal alfa (*alpha transparency*), outro para o canal vermelho (*red*), outro para o canal verde (*green*) e outro para o canal azul (*blue*). Seguramente, pode-se selecionar um *bit* (o menos significativo) em cada *byte* do *pixel* para representar o *bit* a ser escondido sem causar alterações perceptíveis na imagem [Wayner, 2002], [Popa, 1998], [Petitcolas et al., 1999].

Acompanhe o exemplo da figura 5.1 para entender melhor. Suponha que se deseja esconder a letra **E** dentro da porção de imagem.

Na figura 5.1, têm-se três *pixels* da imagem de cobertura. Como a letra **E** pode ser escrita em forma binária segundo seu código ASCII como **10000011**, é suficiente utilizar apenas os dois primeiros *pixels* da imagem. Assim, utilizando-se a técnica LSB, tem-se o resultado mostrado na figura 5.2

(00100111 11101001 11001000 11101010) [a, R, G, B]
(10100111 11001000 11101001 11101000) [a, R, G, B]
(11001000 00100111 11101001 00100111) [a, R, G, B]

Figura 5.1: Porção de uma imagem de cobertura

(00100111 11101000 11001000 11101010) [a, R, G, B]
(10100110 11001000 11101001 11101001) [a, R, G, B]
(11001000 00100111 11101001 00100111) [a, R, G, B]

Figura 5.2: Porção da estego-imagem gerada pela porção de imagem 5.1

Os *bits* em negrito representam os LSBs e os *bits* sublinhados representam as modificações necessárias para esconder a letra **E**.

Como exemplo da grande quantidade de dados que podem ser escondidos, suponha uma imagem com tamanho de 1024 por 768 *pixels*. Neste caso, têm-se um total de 786432 *pixels*. Como cada *pixel* possui 4 *bytes* na sua codificação, têm-se 4 *bits* para o uso de técnicas baseadas em LSB.Assim, existe uma possibilidade de esconder cerca de 390 *kilobytes* de dados neste objeto de cobertura.

Uma forma de prover maior robustez às inserções LSB é trabalhar com *stream-generators* capazes de escolher várias posições diferentes e aleatórias na imagem de cobertura, bem como utilizar chaves *esteganográficas* seguindo o estilo da *criptografia* de chave pública.

5.3.2 Técnicas de filtragem e mascaramento

Segundo [Johnson and Jajodia, 1998], técnicas de *filtragem e mascaramento* são restritas às imagens em tons de cinza (*grayscale*). Estas técnicas escondem a informação através da criação de uma imagem semelhante às marcações de *copyright* em papel. Isto acontece porque as técnicas de *watermarking* garantem que, mesmo se a imagem for modificada por métodos de compressão, a marcação não será removida.

Filtragem e mascaramento são técnicas mais robustas que a inserção LSB no sentido de gerarem estego-imagens imunes a técnicas de compressão e recorte. Ao contrário das modificações LSB, filtragem e mascaramento trabalham com modificações nos *bits mais significativos* das imagens. As imagens de cobertura devem ser em tons de cinza porque estas técnicas não são eficientes em imagens coloridas [Popa, 1998]. Isto deve-se ao fato de que modificações em *bits* mais significativos de imagens em cores geram alta quantidade de "ruído" tornando as informações

detectáveis.

5.3.3 Algoritmos e transformações

Manipulações LSB são rápidas e relativamente fáceis de serem implementadas. No entanto, estas técnicas produzem estego-imagens que podem ser facilmente destruídas através do manuseio da imagem com recorte e/ou compressão [Artz, 2001].

Por outro lado, sabe-se que a compressão de imagens é uma das formas mais eficientes de armazenar imagens de alta qualidade. Desta forma, os algoritmos de transformação geralmente trabalham com formas mais sofisticadas de manuseio de imagens como brilho, saturação e compressão das imagens.

Utilizando técnicas como a *transformação discreta do cosseno*, *transformada discreta de Fourier* e *transformada Z*, entre outras, estes algoritmos tomam como aliado o principal inimigo da inserção LSB: a compressão. Por isso, configuram-se como as mais sofisticadas técnicas de mascaramento de informações em imagens conhecidas [Johnson and Jajodia, 1998] e [Popa, 1998].

5.4 Como aumentar a robustez

Qualquer esquema de *esteganografia* por mais robusto que seja ainda pode, de alguma forma, ser incrementado. De forma geral, deve-se sempre ter em mente o *Princípio de Kerchoff*: a segurança do algoritmo deve estar na chave secreta; sem a chave secreta um agressor (alguém interessado em descobrir indevidamente a mensagem) terá poucas chances de êxito [Schneier, 1995]. A seguir, encontram-se algumas formas de tornar um sistema *esteganográfico-digital* mais forte e seguro.

5.4.1 Chaves

A inserção de *bits* de uma mensagem de forma seqüencial em uma imagem de cobertura é matematicamente deselegante e pouco segura.

Neste caso, pode-se fazer o uso de chaves de mascaramento. Pode-se, por exemplo, usar chaves que indiquem a distância do próximo *bit* a ser inserido em relação ao último LSB selecionado. Este tipo de chave é conhecido como chave de deslocamento. Isto quer dizer que P, o próximo LSB a ser alterado em uma imagem de cobertura, é dado por $P = P_u + C_a$ onde P_u é a posição ocupada pelo último LSB alterado na imagem de cobertura e C_a é a entrada equivalente a $P_u + 1$ na chave de deslocamento. Um mascaramento pode tornar-se mais difícil de quebrar caso a escolha da chave tenha sido feita de modo que os dados na estego-imagem sejam o mais espalhados e não relacionados quanto possível.

5.4.2 Mistura dos dados

A inserção das informações como são vistas no mundo real em uma imagem pode alterar fortemente os seus padrões estatísticos. Este tipo de alteração pode não ser desejável. Deste modo, é interessante tornar aleatórios os dados a serem inseridos de modo a alterar o mínimo possível a imagem de cobertura. Para que isso seja possível, não basta apenas permutar os elementos pertencentes à mensagem a ser mascarada, é necessário, no entanto, que para cada *bit* trocado na imagem de cobertura mude outro em uma operação contrária. Isto quer dizer que, caso um *bit* tenha sido recentemente trocado de $0 \rightarrow 1$, outro *bit* na mesma imagem deve ser mudado de $1 \rightarrow 0$. Este tipo de operação irá produzir um sitema de mascaramento robusto e adaptativo [Wayner, 2002].

5.4.3 Geradores pseudo-aleatórios

Muitos programas de computador usam geradores aleatórios de números para adicionar realismo a cenas, sons e jogos. Monstros podem parecer bem melhores e assustadores se um gerador aleatório de números adicionou manchas, verrugas, cicatrizes para suavizar as peles definidas por esferas matemáticas. Informações podem ser escondidas ao invés de números. A localização de manchas e cicatrizes pode então carregar a mensagem escondida [Wayner, 2002].

5.4.4 Mudança da ordem

Uma lista pode muito bem ser apenas uma lista de elementos. No entanto, a ordem em que os elementos estão nesta lista pode revelar informações surpreendentes [Wayner, 2002].

5.4.5 Compactação

Quanto menor a mensagem a ser enviada menor as chances de sua detecção. Deste modo, pode-se utilizar a compactação dos dados. Este tipo de operação ainda irá permitir o envio de grande quantidade de informação num espaço relativamente pequeno de armazenamento.

5.4.6 Divisão da informação

Segundo [Wayner, 2002], não há razão pela qual os dados precisam ser escondidos em apenas um pacote[1] Os dados podem ser divididos em qualquer número

[1]Cada pacote representa uma parte da mensagem a ser enviada. Caso o pacote seja único, a mensagem não foi dividida [Wayner, 2002].

de pacotes e tomar diferentes rotas para seu destino. Algoritmos sofisticados podem dividir a informação de modo que qualquer subconjunto k de n pacotes seja suficiente para recuperar a informação escondida.

5.4.7 Criptografia

Além de tudo isso, pode-se utilizar a *criptografia*. Antes de efetuar qualquer mascaramento, pode-se criptografar os dados. Depois de recuperá-los do outro lado, basta descriptografá-los.

5.4.8 Combinação das técnicas

Pode-se combinar todas as técnicas de inúmeras maneiras diferentes. Inicialmente a informação pode ser escondida segundo a ordem dos elementos em uma lista, depois esta lista pode ser compactada e criptografada. Em seguida, pode-se dividir a informação em vários pacotes de modo que apenas um subconjunto destes seja necessário para recuperar a informação. Finalmente, procede-se o mascaramento usando algoritmos adaptativos para manter os padrões estruturais e estatísticos do meio de cobertura.

5.5 Outros meios para o mascaramento

Além das diversas *técnicas esteganográficas* citadas até o momento, é possível realizar o mascaramento de informações em diversos outros meios e dispositivos. A seguir, tem-se uma pequena lista de possibilidades:

- tecnologias de holografia;
- infravermelhos;
- *pagers*;
- tintas, termocrômica, fotocrômica;
- seqüências de DNA;
- áreas vazias em discos rígidos.

Para mais informações sobre cada uma destas tecnologias, consulte [Wayner, 2002], [Petitcolas et al., 1999], [Anderson and Petitcolas, 1998], [Artz, 2001] e [Johnson and Jajodia, 1998].

O mais intrigante destes meios é o mascaramento de informações em seqüências de DNA. A proposição, feita inicialmente por [Clelland et al., 1999], é baseada no fato de que, na verdade, qualquer material genético é formado por cadeias de quatro nucleotídeos (Adenina, Citosina, Guanina e Timina) as quais podem ser comparadas a um alfabeto de quatro letras: A, C, G e T. Deste modo, dado que os cientistas atualmente são capazes de fabricar cadeias de DNA com um conjunto predeterminado de nucleotídeos, nada impede a atribuição de uma letra do alfabeto, um número ou sinais de pontuação a um grupo de três nucleotídeos, (por exemplo, “A”= CGA, “B”= CCA, . . .). Assim, é possível compor uma “mensagem genética”. Para disfarçar as pistas, pode-se misturar algumas outras sequências aleatórias de nucleotídeos. O resultado pode ser visível apenas ao microscópio eletrônico. Como possível aplicação, pode-se imaginar que uma empresa que produza uma nova espécie de tomate queira incluir sua marca de fábrica nas moléculas do tomate a fim de evitar as imitações.

Capítulo 6

Técnicas de esteganálise

6.1 Considerações iniciais

Grande parte das técnicas de *esteganografia* possuem falhas e/ou inserem artefatos (padrões) detectáveis nos objetos de cobertura. Algumas vezes, basta um agressor (alguém interessado em descobrir indevidamente a mensagem) fazer um exame mais detalhado destes artefatos para descobrir que há mensagens escondidas. Outras vezes, o processo de mascaramento de informações foi robusto e as tentativas de recuperar ilicitamente as mensagens podem ser bastante difíceis. Ao campo das pesquisas relacionado às tentativas de descobrir mensagens secretas dá-se o nome de *esteganálise*, uma alusão à *criptoanálise*, o campo de pesquisas relacionado à quebra de códigos.

Atualmente, as pesquisas em *esteganálise* estão mais concentradas em simplesmente identificar a presença de mensagens escondidas ao invés de extraí-las. Recuperar os dados escondidos, no momento, está além das capacidades da maioria dos testes porque muitos algoritmos de mascaramento utilizam geradores aleatórios criptográficos muito seguros para misturar a informação no processo de mascaramento. Na maioria das vezes, os *bits* são espalhados pelo objeto de cobertura. Os melhores algoritmos de *esteganálise* não são capazes de dizer onde está a informação, mas, provavelmente, podem dizer que os dados estão presentes.

Segundo [Wayner, 2002], a identificação da existência de uma mensagem escondida é suficiente para um agressor. As mensagens são, muitas vezes, frágeis e um agressor pode, sem muita dificuldade, destruir a mensagem mesmo sem tê-la recuperado. Algumas vezes, os dados podem ser destruídos simplesmente destruindo o objeto de cobertura. Outras vezes, basta aplicar um gerador de números aleatórios nos bits menos significativos destruindo qualquer mensagem (informação) ali presente. Ataques geométricos podem ser aplicados aos objetos de

cobertura de modo que, se o agressor não pode recuperar a mensagem, então o destinatário também não o poderá fazer.

Todos estes ataques dependem da identificação de algumas características de um objeto de cobertura (como imagens, vídeos, sons) que foram alteradas pelo processo de mascaramento.

Não há qualquer garantia de que um algoritmo *esteganográfico* possa resistir à *esteganálise*.

> *Pode-se desenvolver um software que seja capaz de enganar todos os computadores uma vez, ou mesmo pode-se enganar alguns computadores todas as vezes. No entanto, nunca se poderá desenvolver um software capaz de enganar todos os computadores todas as vezes.* (Anônimo)

A seguir, são apresentados os principais tipos de abordagens da *esteganálise* e os principais ataques. Finalmente, é apresentado o *RS-Esteganálise* — um dos métodos estatísticos mais eficientes na detecção de mensagens escondidas em imagens até o momento —. Tal método é apresentado por ter sido utilizado na análise das *estego-imagens* produzidas pelo **Camaleão**, desenvolvido durante este trabalho, ver capítulo 8.

6.2 Tipos de ataques

As principais abordagens de ataques à *esteganografia* em imagens são mostradas a seguir. Mais informações podem ser obtidas em [Tzschoppe et al., 2003], [Westfeld and Pfitzmann, 2003], [Wayner, 2002] e [Fridrich et al., 2002].

Ataques aurais

Alguns ataques retiram as partes significativas da imagem como um meio de facilitar aos olhos humanos a busca por anomalias na imagem. Um teste comum é mostrar os *bits* menos signifitivos da imagem. Ruído[1] completamente randômico freqüentemente revela a existência de uma mensagem escondida, uma vez que as imperfeições de câmeras, *scanners* e outros meios digitalizadores sempre deixam marcas de grande estrutura nos *bits* menos significativos[2].

[1] Ruído aqui não é o ruído no sentido de *Processamento Digital de Imagens*. Este ruído explicita apenas a constatação de que o *bits* presentes não dizem nada (não possuem nenhum padrão) e estão aleatoriamente distribuídos

[2] Grande parte de câmeras digitais ou mesmo digitalizadores utilizam os LSBs das imagens para acrescentar seus próprios padrões. Isto elimina o ruído existente e acrescenta os padrões desejados pelos fabricantes.

Além disso, o cérebro humano é capaz de descobrir as mais sutis diferenças. Esta é a razão pela qual muitas marcações de áudio (*audio watermarking*) de grandes gravadoras são frustradas graças aos ouvidos de músicos bem-treinados.

Ataques Estruturais

O formato do arquivo de dados freqüentemente muda assim que outra mensagem é inserida. Nesses casos, um sistema capaz de analisar padrões estruturais seria capaz de descobrir a mensagem escondida. Por exemplo, ao esconder mensagens em imagens indexadas (baseadas em paletas de cores), pode ser necessário usar versões diferentes de paletas. Este tipo de atitude muda as características estruturais da imagem de cobertura, logo as chances de detecção da presença de uma mensagem escondida aumentam.

Ataques estatísticos

Os padrões dos *pixels* e seus *bits* menos significativos freqüentemente revelam a existência de uma mensagem secreta nos perfis estatísticos [Wayner, 2002]. Os novos dados não têm os mesmos perfis esperados.

Muitos dos estudos de matemática estatística objetivam classificar se um dado fenômeno ocorre ao acaso. Cientistas usam estas ferramentas para determinar se suas teorias explicam bem tal fenômeno. Estas técnicas estatísticas também podem ser usadas para determinar se uma dada imagem e/ou som têm alguma mensagem escondida. Isto é possível porque, na maioria das vezes, os dados escondidos são mais aleatórios que os dados que foram substituídos no processo de mascaramento. Alguns testes conhecidos são χ^2 (*Chi Squared Test*) [Wayner, 2002], e o *RS-Steganalysis* [Fridrich et al., 2001].

6.3 χ^2

Este é um dos mais simples testes estatísticos relacionados à detecção de fenômenos aleatórios. Este método consiste em somar as discrepâncias. Seja $\{e_0, e_1, \ldots\}$ o número de vezes que uma seqüência de eventos ocorre. Neste caso, isto pode ser o número de vezes que um *bit* menos significativo é 1 ou 0. Os eventos podem ser classificados de acordo com a tabela 6.1.

Seja $E(e_i)$ o número esperado de vezes que um evento poderia ocorrer em uma amostragem totalmente randômica. A taxa de aleatoriedade é dada por:

$$\chi^2 = \sum \frac{(e_i - E(e_i))^2}{E(e_i)} \tag{6.1}$$

Evento	Bit	Bit vizinho
e_0	0	0
e_1	0	1
e_2	1	0
e_3	1	1

Tabela 6.1: Eventos relacionados a dois LSBs

Grandes valores indicam uma condição não aleatória, aquela em que a imagem foi criada por uma câmera digital ou um *scanner* e está em sua forma original. Valores muito pequenos indicam um alto grau de aleatoriedade e, por conseguinte, isto está ligado com a presença de mensagens criptografadas.

6.4 RS-Esteganálise

Proposto por [Fridrich et al., 2001], este é um dos métodos de detecção mais robustos disponíveis. Suas análises podem ser válidas tanto para imagens em tons de cinza quanto coloridas. Não há distinção na profundidade de cores da imagem analisada, isto é, o método é válido tanto para imagens de 8 bpp (*bits* por *pixel*) quanto para imagens de 32 bpp. Algumas extensões do modelo inicial foram feitas e em algumas delas é possível até estimar o tamanho da mensagem escondida. No entanto, neste trabalho de pesquisa o método foi restrito a apenas identificar a presença ou ausência de uma mensagem escondida.

Definições

Seja IMG a imagem testada. IMG possui MxN *pixels* e cada *pixel* tem valores dados por um conjunto P. Como exemplo, para uma imagem de 8 bpp, tem-se $P = \{0, \ldots, 255\}$. Então, divide-se IMG em grupos de *pixels* disjuntos G de n *pixels* adjacentes

$$G = (x_1, \ldots, x_n) \in R. \tag{6.2}$$

Como exemplo, pode-se escolher grupos de $n = 4$ *pixels* adjacentes. Feito isso, define-se uma função de discriminação f responsável por atribuir um número real $f(x_1, \ldots, x_n)$ para cada grupo de *pixels* $G = (x_1, \ldots, x_n)$. Quanto mais aleatório for o grupo de *pixels*, maior o valor da função de discriminação. A função é como segue:

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n-1} |x_{i+1} - x_i| \tag{6.3}$$

Finalmente, define-se uma operação inversível F sobre P chamada *flipping*. Por *flipping* entende-se a permutação dos níveis de cores e consiste em 2 ciclos. Assim, F tem a propriedade que $F^2 =$ Identidade ou $F(F(x))) = x$ para todo $x \in P$. A permutação $F_1 : 0 \leftrightarrow 1, 2 \leftrightarrow 3, \ldots, 254 \leftrightarrow 255$ corresponde a *flipar* (negar) o LSB de cada nível de cor. Adicionalmente pode-se definir uma função de *shifting* (deslocamento) $F_{-1} : -1 \leftrightarrow 0, 1 \leftrightarrow 2, \ldots, 255 \leftrightarrow 256$, ou

$$F_{-1}(x) = F_1(x + 1) - 1 \; \forall \; x \tag{6.4}$$

Para completar, define-se F_0 como sendo a permutação de identidade $F(x) = x \; \forall \; x \; \in \; P$. Estas operações são utilizadas para classificar os grupos de *pixels* em três categorias diferentes R, S e U:

- Grupos regulares : $G \in R \Leftrightarrow f(F(G)) > f(G)$
- Grupos singulares : $G \in S \Leftrightarrow f(F(G)) < f(G)$
- Grupos não-usáveis : $G \in U \Leftrightarrow f(F(G)) = f(G)$

Nestas expressões, $F(G)$ significa que a função de *flipping* F foi aplicada para os componentes do vetor $G = (x_1, \ldots, x_n)$. Caso seja desejado aplicar diferentes *flippings* em diferentes *pixels*, deve-se usar uma máscara M que irá denotar quais os *pixels* deverão sofrer alterações. A máscara M é uma n-tupla com valores $\{-1, 0, 1\}$. Define-se o grupo alterado GA como:

$$GA = (F_{M(1)}(x_1), F_{M(2)}(x_2), \ldots, F_{M(n)}(x_n)) \tag{6.5}$$

O objetivo da função F é perturbar os *pixels* de uma forma pouco significativa tal como aconteceria no processo de mascaramento de uma mensagem.

O método

Seja R_M percentagem do número de grupos regulares em relação ao total de grupos para a máscara M. De forma similar, S_M irá denotar o número relativo de grupos singulares. Tem-se que $R_M + S_M \leq 1$ e $R_{-M} + S_{-M} \leq 1$, para a máscara negativa. A hipótese estatística para o método é que, em imagens típicas, o valor

esperado de R_M é aproximadamente igual ao de R_{-M} e o mesmo é verdade para S_M e S_{-M}.

A equação 6.6 foi empiricamente comprovada por [Fridrich et al., 2001]. A randomização do plano LSB força a diferença entre R_M e S_M para zero à medida que o tamanho m da mensagem escondida cresce. Depois de alterar os LSBs de 50 por cento dos *pixels* (é o que acontece quando se esconde uma mensagem aleatória em todos os *pixels*), obtém-se $R_M \cong S_M$, isto é o mesmo que dizer que a capacidade de mascaramento no plano LSB agora é zero. O fato surpreendente é que um efeito contrário acontece com R_{-M} e S_{-M}, sua diferença aumenta proporcionalmente ao tamanho da mensagem escondida.

$$R_M \cong R_{-M} \text{ e } S_M \cong S_{-M} \tag{6.6}$$

Desta forma, ao analisar IMG (a imagem testada), esta provavelmente estará escondendo uma mensagem se:

- *Condição 1*: $R_M - R_{-M} = i$ e i é muito grande;
- *Condição 2*: $R_M - S_M = k$ e k é muito grande;

Valores muito grandes para i acontecem quando $i \geq 2.5\%$ do total de grupos. Valores muito grandes para k acontecem quando $k \geq 25\%$ do total de grupos. Um mascaramento detectável ocorre toda vez que a primeira condição é verdadeira. Caso apenas a segunda condição seja verdadeira, há apenas uma suspeita de que houve um mascaramento [Fridrich et al., 2002].

6.5 Considerações finais

Existem muitos outros métodos de *esteganálise* não citados neste trabalho. Na maioria das vezes, estes métodos são desenvolvidos para detectar os mascaramentos feitos por um *software* de *esteganografia* em específico. Raras são as vezes em que um método genérico é robusto o suficiente para detectar vários tipos de mascaramento. É nesse sentido que o RS-Esteganálise está sendo muito utilizado e aperfeiçoado constantamente e, segundo [Fridrich et al., 2002], novas versões do método prometem funcionar para qualquer tipo de imagem digital.

Capítulo 7

Resultados e discussão – esteganografia

7.1 Considerações iniciais

Como resultado desta pesquisa, foi desenvolvido o *Camaleão: um software para segurança digital utilizando esteganografia*[1]. Nas seções a seguir, são apresentadas as principais características e funcionalidade do **Camaleão**, tais como, os processos de mascaramento, a recuperação das informações, o gerenciamento das chaves de deslocamento e a sua relação com a robustez. Ao final, apresenta-se a modelagem do sistema, utilizando a notação UML (*Unified Modeling Language*) [Booch et al., 1999].

7.2 O Camaleão

Camaleão é um *software* que permite a comunicação segura pela *internet* por fazer uso da *esteganografia*. O produto possui várias características tais como:

- *ambiente multiplataforma:* por ter sido desenvolvido na linguagem de programação Java [Sun Microsystems, 2003], o funcionamento do **Camaleão** torna-se praticamente independente do sistema operacional utilizado. O sistema funcionou bem sobre os sistemas operacionais Linux, Windows 9x, Windows XP e Mac OS X. Embora não testado, o **Camaleão** provavelmente funcionará sem problemas sobre o sistema Solaris. Para isso, o usuário deve

[1] Para copiar o *software* desenvolvido ou mesmo obter mais informações sobre o projeto acesse o *site* `www.comp.ufla.br/undersun`

ter em seu computador a máquina virtual java (JVM – *Java Virtual Machine*) 1.4 ou superior;

- *ambiente bilíngue:* Visando alcançar o maior número de pessoas o **Camaleão** foi desenvolvido em dois idiomas. O idioma português que funciona em modo nativo e o idioma inglês que funciona como opcional;

- *código aberto:* o **Camaleão** é disponibilizado sob a licença de uso GPL (General Public GNU Licence) ou licença pública geral GNU [FSF, 2003]. De acordo com esta licença, o **Camaleão** pode ser modificado e utilizado livremente desde que se mantenha as referências aos autores originais intactas;

- *tipos de mascaramento e recuperação:* o **Camaleão** permite o mascaramento de textos, imagens e quaisquer outros arquivos binários dentro de outras imagens. As imagens de cobertura podem ser de sufixo (extensão) *.jpg* ou *.png*. A imagem de saída (contendo o mascaramento) tem o sufixo *.png*. O processo de mascaramento pode ser baseado em chave de deslocamento ou a partir das configurações-padrão. Caso seja baseado em chave de deslocamento, esta pode ser periódica ou não [2]. Além disso, o mascaramento pode ser linear ou aleatório.

- *robustez:* visando ter uma maior segurança, o sistema permite a geração de chaves de deslocamento configuráveis. É possível gerar chaves de vários tamanhos diferentes sob vários módulos diferentes[3].

7.3 A ferramenta em execução

As figuras 7.1 a 7.8 apresentam algumas telas do **Camaleão**.

[2] De forma geral, para cada entrada (*bit*) de uma mensagem a ser mascarada existe uma entrada (*deslocamento*) respectivo na chave de deslocamento. Chaves periódicas possuem menos entradas que a mensagem a ser mascarada. Assim que acabam as entradas da chave, usa-se novamente as mesmas entradas. Por outro lado, chaves não-periódicas têm o número de entradas maior ou igual ao número de entradas da mensagem a ser escondida

[3] Chaves de módulo k têm todas as suas entradas menores que k. Para mais detalhes ver a seção 7.6

Figura 7.1: Tela inicial do sistema

Figura 7.2: Mascaramento de um texto

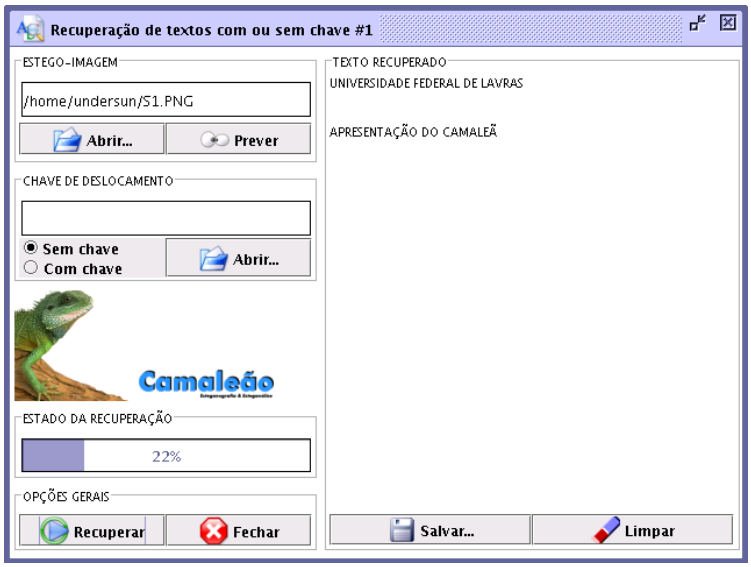

Figura 7.3: Recuperação de um texto

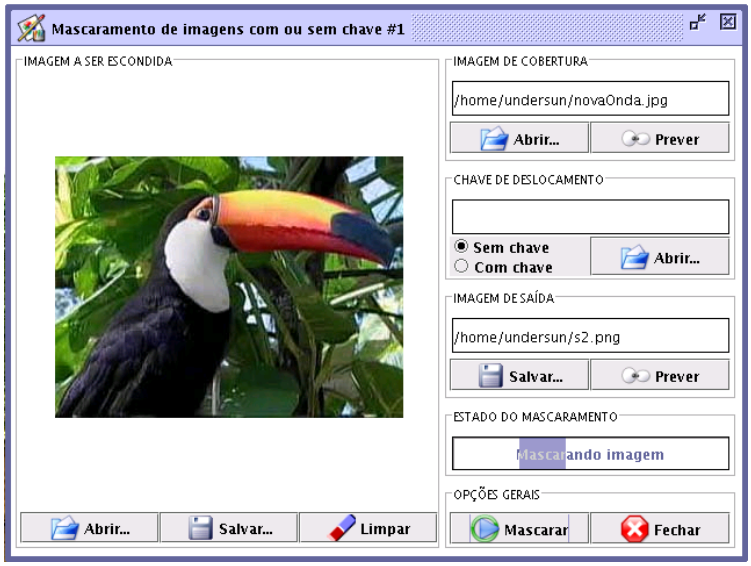

Figura 7.4: Mascaramento de uma imagem

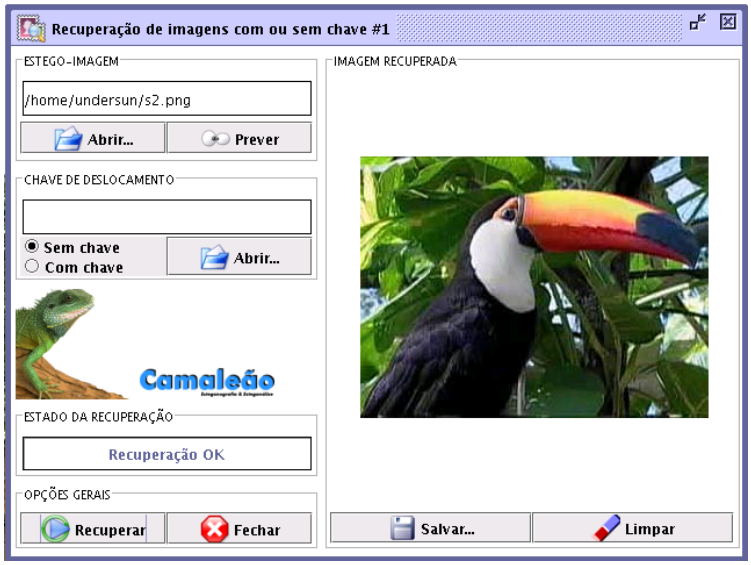

Figura 7.5: Recuperação de uma imagem

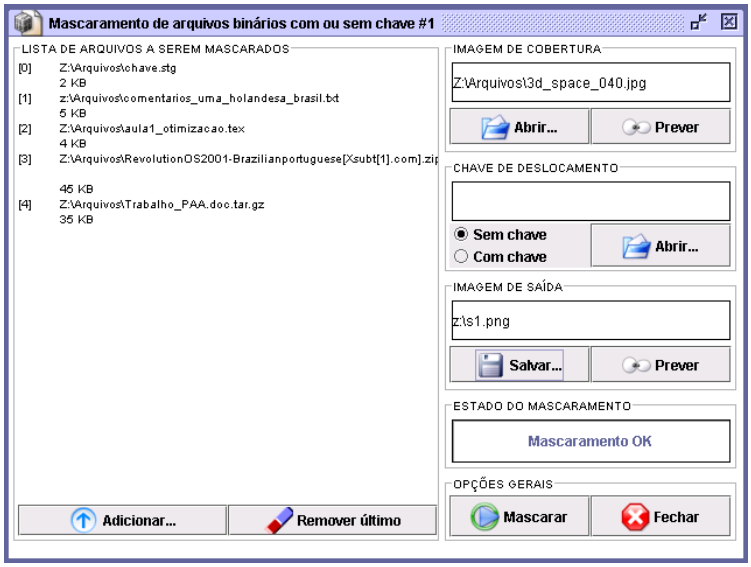

Figura 7.6: Mascaramento de uma lista de arquivos binários

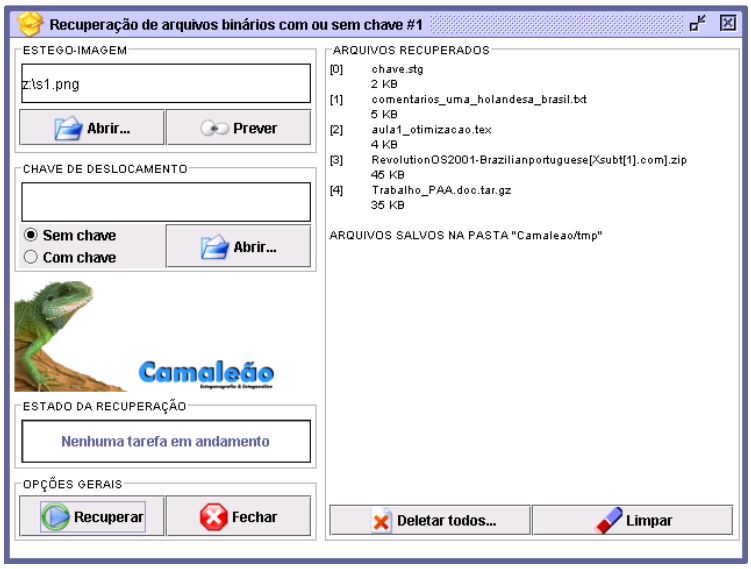

Figura 7.7: Recuperação de uma lista de arquivos binários

Figura 7.8: Gerenciamento de chaves de deslocamento

As imagens produzidas após o processo de mascaramento são praticamente idênticas. Segundo [Wayner, 2002], os humanos conseguem capturar mudanças em uma imagem quando estas ocorrem em um fator acima de 3%. No caso, como o **Camaleão** trabalha apenas com o *bit* menos significativo, o conjunto total de mudanças no caso de uma mensagem que afete todos os LSBs é de apenas 0,78% — dado que cada componente de cor tem 8 *bits* a alteração no último *bit* afeta o conjunto em $\frac{2}{256}$% —. Caso o segundo *bit* menos significativo também seja alterado, a taxa de alteração sobe para 1,56%, ainda imperceptível à maioria dos seres humanos.

A figura 7.9 é uma imagem antes de um mascaramento. Seu tamanho é de 133,2 KB. A figura 7.10, de 134KB, apresenta a estego-imagem resultante em que 35% dos seus LSBs foram alterados pelo processo de mascaramento feito pelo **Camaleão**. Elas são praticamente idênticas.

A discrepância no tamanho final das imagens não é relevante dado que apenas a imagem *arara_saida.png* será enviada ao destinatário[4]. Caso um interceptador capture a estego-imagem antes que ela chegue ao seu destino ele não terá como comparar os tamanhos dado que ele não possui a imagem original.

Figura 7.9: Imagem *arara.png* antes do mascaramento – 133,2KB

[4]A diferença no arquivo de saída é exclusivamente relacionada ao mecanismo de gravação de imagens da linguagem de programação utilizada. Nenhuma informação é acrescentada à imagem de cobertura.

Figura 7.10: Imagem *arara_saida.png* após o mascaramento – 134,0KB

7.4 O processo de mascaramento da informação

Neste processo, os *bits* menos significativos de uma imagem de cobertura são alterados segundo as configurações dos *bits* de um segundo arquivo. Este segundo arquivo é a mensagem que se deseja enviar em segredo.

A distância entre dois sucessivos *bits* escondidos é o número de *bits* menos significativos entre eles e é controlado por um número aleatório. Tais distâncias pertencem ao intervalo $\{0, \ldots, m\}$, onde m denota um valor máximo, e é um segredo entre o emissor e o receptor. Esta chave corresponde a uma chave simétrica em um *criptosistema* simétrico. Deste modo, o princípio de Kerchoff, seção 5.4, é válido. Sem o conhecimento da sucessão correta de distâncias entre os *bits*, qualquer agressor terá poucas chances de êxito ao tentar recuperar a mensagem escondida.

A figura 7.11 descreve o processo de mascaramento. O emissor modifica o *stream* original usando a chave secreta. Caso não exista uma chave de deslocamento, o **Camaleão** efetua o mascaramento segundo as configurações-padrão, isto é, simulando um deslocamento de 1 para todo *bit* a ser mascarado. Para mascarar o primeiro *bit* o emissor precisa saber quantos *bits* deve saltar. No primeiro caso, deve-se saltar um *bit* dado que o deslocamento é zero. Deste modo, basta saltar do LSB atual para o próximo e efetuar o mascaramento. No entanto, caso o deslocamento seja de dois deve-se contar três LSBs a partir do LSB sendo atualmente utilizado e efetuar o mascaramento.

Stream original	*Stream* a ser mascarado	Chave (distâncias)	*Stream* a ser enviado
00110110	0	0	00110110
00100111	1	0	00100110
10100000	1	2	10100000
10101001	0	0	10101001
00000010	0	1	00000010
10111010	.	.	10111011
00011100	.	.	00011100
01111110	.	.	01111110
01000101	.	.	01000101
11100011	.	.	11100011
10000001	.	.	10000001

Figura 7.11: O processo de mascaramento segundo uma chave de deslocamento

O *stream* a ser mascarado pode ser um texto plano[5], uma imagem de 32 *bits* de cor por *pixel* (32 bpp), ou qualquer outro arquivo do computador. Toda mensagem, antes de ser codificada pelo **Camaleão**, passa por um pré-processamento responsável por inserir um cabeçalho de reconhecimento. A seguir, apresentam-se as principais decisões relacionadas ao mascaramento de um texto e de uma imagem.

7.4.1 Mascaramento de textos

Para que um texto possa ser mascarado, ele precisa de um cabeçalho de reconhecimento e de um marcador de fim de mensagem, figura 7.12.

O mascaramento pode ser caracterizado como *linear* ou *randômico*. Defina-se:

- C = STEGTXT# (Cabeçalho)
- M = ALO MUNDO (Mensagem a ser escondida)

[5]Texto contendo apenas caracteres. Normalmente arquivo com extensão *.TXT*

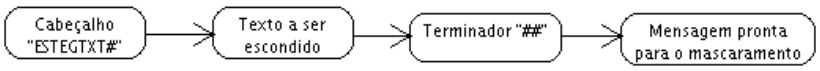

Figura 7.12: O processo de mascaramento de textos

- T = ## (Terminador de mensagens)
- M' = MDLOONU A (Permutação de M)

Para maiores esclarecimentos sobre como é o processo de permutação de M veja a seção 7.7.

Mascaramento linear

O texto é mascarado seqüencialmente assim como ele pode ser lido. Deste modo, a mensagem CMT é escondida na imagem de cobertura.

Mascaramento randômico

O texto continua mascarado seqüencialmente, no entanto, é simulado o mascaramento randômico através da permutação de M que produz M'. Finalmente, procede-se o mascaramento com a mensagem $CM'T$.

7.4.2 Mascaramento de imagens

Para que uma imagem possa ser mascarada, ela precisa de um cabeçalho de reconhecimento, de sua largura e altura em *pixels* e de um marcador de fim de mensagem, figura 7.13. Defina-se IMG a imagem a ser escondida:

O mascaramento pode ser caracterizado como *linear* ou *randômico*. Defina-se:

- L = Largura da imagem a ser escondida
- A = Altura da imagem a ser escondida
- C = STEGIMG# + L + A (Cabeçalho)
- M = 255 123 36 15 100 200 129 33 ... (RGBs da imagem a ser escondida)

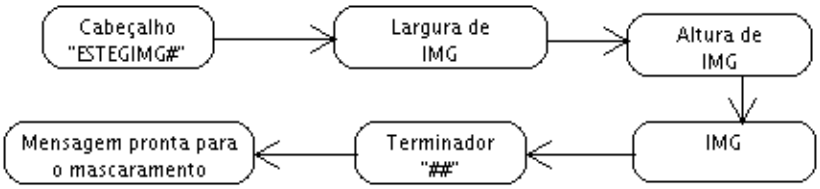

Figura 7.13: O processo de mascaramento de imagens

- T = ## (Terminador de mensagens)
- M' = 202 31 121 51 65 213 30 89... (Permutação de M)

Para maiores esclarecimentos sobre como é o processo de permutação de M veja a seção 7.7.

Mascaramento linear

A imagem é escondida seqüencialmente assim como ela pode ser vista. A mensagem CMT é inserida na imagem de cobertura.

Mascaramento randômico

A imagem continua escondida seqüencialmente, no entanto, é simulado o mascaramento randômico através da permutação de M que produz M'. Finalmente, procede-se o mascaramento com a mensagem $CM'T$.

7.4.3 Mascaramento de arquivos binários

O nome *arquivos binários* deve-se à maneira como o sistema interpreta os arquivos neste processo, *byte* a *byte*, isto é, independentemente da entrada, para o **Camaleão** tudo é um *stream* de *bits*. Esta funcionalidade permite o mascaramento de uma lista de até 256 arquivos de uma só vez dentro de uma imagem[6]. É a opção mais perigosa do sistema, um vez que se pode guardar qualquer coisa representável por 0's e 1's em uma inocente fotografia digital.

O procedimento acontece de acordo com a figura 7.14.

Para entender o processo, defina-se:

[6]Isto porque apenas um *byte* foi utilizado para representar, no cabeçalho inserido, o número de arquivos mascarados em uma estego-imagem

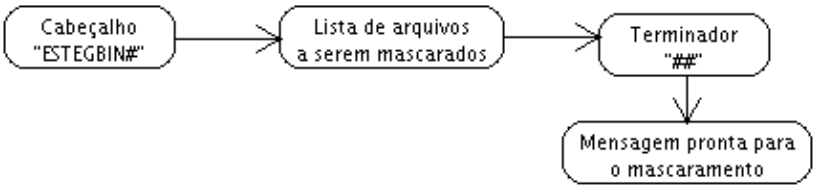

Figura 7.14: O processo de mascaramento de arquivos binários

- C = STEGBIN# (Cabeçalho)
- NA = Número de arquivos na lista
- N_1 = Nome do primeiro arquivo a ser escondido
- N_2 = Nome do segundo arquivo a ser escondido
- B_1 = 01100110 (*bits* que representam N_1)
- B_2 = 11100111 (*bits* que representam N_2)
- S = $$ (Separador de campos)
- TA_1 = Tamanho ou número de *bytes* de N_1
- TA_2 = Tamanho ou número de *bytes* de N_2
- T = ## (Terminador de mensagens)

Para que ocorra o processo de mascaramento, deve-se montar a mensagem a ser mascarada:

$$M = C+NA+S+N_1+S+TA_1+S+N_2+S+TA_2+S+B_1+B_2+T \quad (7.1)$$

Uma vez que montada a mensagem, o mascaramento ocorre similarmente ao mascaramento de textos. A posição de cada *bit* é dada por uma chave de deslocamento ou pelas configurações-padrão, caso não haja uma chave.

7.5 O processo de recuperação da informação

Neste processo, os *bits* menos significativos de uma imagem de cobertura são todos extraídos e colocados em uma lista. Os *bits* serão posteriormente selecionados para a formação da mensagem final segundo as configurações da chave de deslocamento que está em posse do receptor da mensagem.

A figura 7.15 descreve o processo de recuperação. O receptor captura os *bits* certos a partir dos deslocamentos da chave. Isto quer dizer que, para recuperar o primeiro *bit* o receptor verifica o deslocamento relativo na chave. Como a primeira entrada da chave é zero, o segundo LSB da tabela contém um *bit* a ser recuperado. O deslocamento para o terceiro *bit* a ser recuperado também é zero, logo o segundo LSB também contém um *bit* válido. No entanto, o terceiro *bit* a ser recuperado está no sexto LSB dado que o último LSB utilizado foi o terceiro e o deslocamento relativo ao terceiro *bit* válido é de 2.

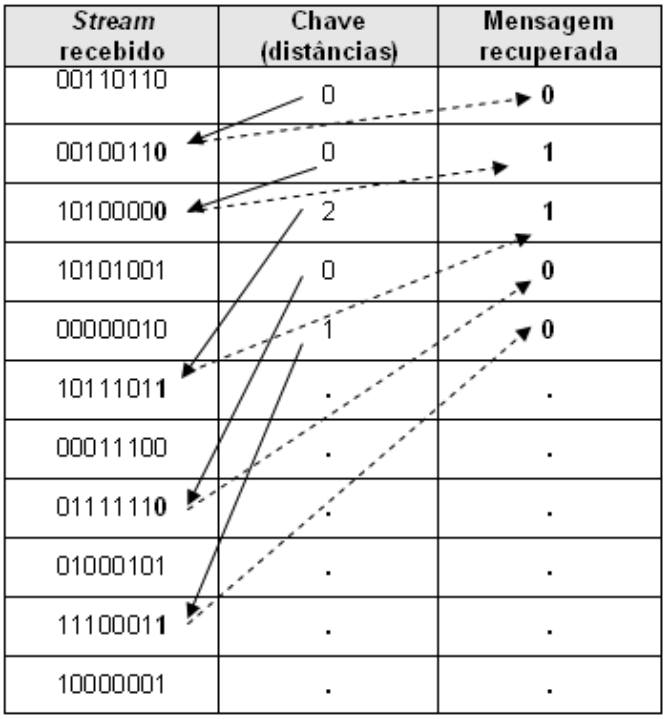

Figura 7.15: O processo de recuperação segundo uma chave de deslocamento

7.5.1 Recuperação de textos

Para que um texto possa ser recuperado, é necessário interpretar o conjunto de *bits* recuperados e verificar o cabeçalho. O processo ocorre de acordo com a figura 7.16.

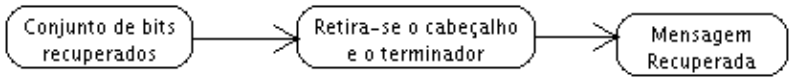

Figura 7.16: O processo de recuperação de textos

A recuperação pode ser caracterizada como *linear* ou *randômica*. Defina-se:

- C = STEGTXT# (Cabeçalho)
- M^1 = ALO MUNDO (Mensagem recuperada linearmente)
- M^2 = MDLOONU A (Mensagem recuperada randomicamente)
- T = ## (Terminador de mensagens)
- M' = ALO MUNDO (M^2 sem as permutações aplicadas)

Recuperação linear

O texto é recuperado seqüencialmente a partir da estego-imagem. Após a recuperação, obtém-se a mensagem M^1 que pode ser lida imediatamente.

Recuperação randômica

Caso o texto tenha sido randomicamente mascarado o processo de recuperação obtém M^2 que não pode ser lido imediatamente. Para que isso seja possível, é necessário que as permutações sobre M^2 sejam desfeitas. Feito isso, obtém M' que pode ser lida sem problemas.

7.5.2 Recuperação de imagens

Para que uma imagem possa ser recuperada, é necessário interpretar o conjunto de *bits* recuperados e verificar o cabeçalho para saber qual a largura e altura da imagem escondida, figura 7.17. Seja IMG a imagem escondida:

A recuperação pode ser caracterizada como *linear* ou *randômica*. Defina-se:

Figura 7.17: O processo de recuperação de imagens

- L = Largura da imagem a ser escondida
- A = Altura da imagem a ser escondida
- C = STEGIMG# + L + A (Cabeçalho)
- M^1 = 255 123 36 15 100 200 129 33 ... (RGBs da imagem recuperada linearmente)
- M^2 = 202 31 121 51 65 213 30 89 ... (RGBs da imagem recuperada randomicamente)
- T = ## (Terminador de mensagens)
- M' = 255 123 36 15 100 200 129 33 ... (M^2 sem as permutações aplicadas)

Recuperação linear

A imagem é recuperada seqüencialmente a partir da estego-imagem. Após a recuperação, obtém-se a mensagem M^1 que pode ser visualizada imediatamente.

Recuperação randômica

Caso a imagem tenha sido randomicamente escondida, o processo de recuperação obtém M^2 que não pode ser visualizada imediatamente. Para que isso seja possível, é necessário que as permutações sobre M^2 sejam desfeitas. Feito isso, obtém-se M' que pode ser visualizada sem problemas.

7.5.3 Recuperação de arquivos binários

Para que uma lista de arquivos binários possa ser recuperada é necessário interpretar o conjunto de *bits* recuperados e verificar cada campo fazendo a extração dos arquivos segundo os dados do cabeçalho da mensagem recuperada. O processo ocorre de acordo com a figura 7.18.

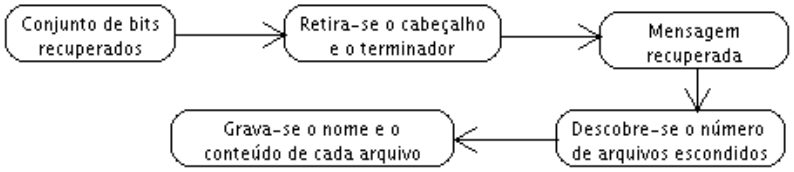

Figura 7.18: O processo de recuperação de arquivos binários

Para que ocorra o processo de recuperação, deve-se interpretar o conjunto de *bits* recuperados e separar os campos. Defina-se:

- MR = Mensagem recuperada
- C = STEGBIN# (Cabeçalho)
- NA = Número de arquivos na lista
- N_1 = Nome do primeiro arquivo recuperado
- N_2 = Nome do segundo arquivo recuperado
- B_1 = 01100110 (*Bytes* que representam N_1)
- B_2 = 11100111 (*Bytes* que representam N_2)
- S = $$ (Separador campos)
- TA_1 = Tamanho ou número de *bytes* de N_1
- TA_2 = Tamanho ou número de *bytes* de N_2
- T = ## (Terminador de mensagens)

A mensagem que foi recuperada tem os seguintes campos:

$$MR = C+NA+S+N_1+S+TA_1+S+N_2+S+TA_2+S+B_1+B_2+T, \quad (7.2)$$

logo, deve-se gravar o primeiro arquivo com nome N_1, tamanho TA_1 e com o conjunto de *bytes* B_1 e assim sucessivamente.

Cada arquivo da lista recuperada é automaticamente gravado em disco em uma pasta temporária *tmp* criada pelo **Camaleão**.

7.6 As chaves

A chave de deslocamento é uma seqüência numérica de tamanho n. Os elementos pertencentes à chave pertencem ao intervalo $\{0, \ldots, m\}$ onde m denota um valor máximo. O valor m é dado a partir do módulo k em que se deseja criar a chave.

Exemplo: Seja C, uma chave de tamanho $n = 5$ e módulo $k = 3$. Cada elemento de C será um número inteiro pertencente ao intervalo $\{0, 1, 2\}$. A chave é criada a partir de um gerador pseudo-aleatório de números.

O **Camaleão** trabalha com chaves de vários tamanhos e módulos. São permitidos três tipos de operações, a saber:

1. *Chave nula:* deve-se usar as configurações-padrão para esta operação. Nenhuma chave de deslocamento foi fornecida;

2. *Chave periódica:* a chave fornecida tem tamanho menor que a mensagem. Esta então deve ser repetida ao se atingir sua n-ésima entrada. Este procedimento cria uma chave do tamanho T da mensagem, mas com $\frac{T}{n}$ períodos;

3. *Chave não-periódica:* a chave fornecida tem o mesmo tamanho da mensagem.

7.7 As permutações

Para que seja possível simular o mascaramento aleatório dos *bits* em uma imagem de cobertura, estes passam por um processo de permutações continuadas de modo que sejam bastante misturados. Seja SP a seqüência de permutações dadas por um conjunto de posições aleatoriamente gerado a partir de uma semente. Seja S a semente:

$$S = \sum_{i=1}^{n} C_i + i \tag{7.3}$$

onde C é o conjunto representando a chave de deslocamento fornecida e n é a cardinalidade de C.

Exemplo: considere $C = [2\ 1\ 0\ 2]$ uma chave de deslocamento. Esta chave é de tamanho $n = 4$ e atua no módulo $Z = 3$, isto é, os números presentes na chave pertencem ao intervalo 0, 1, 2. Para calcular S tem-se que:

$$S = (2 + 0) + (1 + 1) + (0 + 2) + (2 + 3) \tag{7.4}$$

Com isso, a semente gerada é $S = 11$. SP será gerada a partir de um gerador de números pseudo-aleatórios com a semente S.

A utilização de uma semente é necessária para que seja possível montar a mesma seqüência de permutações no lado receptor da mensagem. A criação de S em função da chave de deslocamento aumenta a robustez de forma que qualquer modificação na chave, por mais insignificante que seja, impede a recuperação correta da mensagem.

7.8 Modelagem do sistema

A seguir, são mostrados os aspectos de modelagem do **Camaleão** utilizando alguns diagramas, considerados relevantes para a sua implementação, presentes na notação UML [Booch et al., 1999]. Inicialmente, são apresentados os casos de uso. Em seguida, dado que o **Camaleão** foi implementado de maneira a atender o desenvolvimento de *software* em duas camadas (*two tiers*), [Ambler, 1998], — subsistema de *Interface Homem-Máquina* (IHM) e subsistema de *Lógica de Negócios* (LN) —, apresentam-se as classes desses subsistemas bem como a ligação entre ambos através de um *middleware*. Finalmente, é mostrado o funcionamento do **Camaleão** através de diagramas de atividades. Os aspectos e as decisões desta modelagem foram apresentados por [Costa, 2001].

7.8.1 Diagrama de casos de uso

Os principais casos de uso identificados estão descritos seguindo o *template* (estrutura) reduzido proposto por [Costa, 2001] e são apresentados na figura 7.19.

Para informações detalhadas sobre cada caso de uso (descrição dos casos de uso), ver **Apêndice A**.

7.8.2 Diagrama de classes do subsistema de IHM

A figura 7.20 apresenta as classes presentes no subsistema de *Interface Homem-Máquina*.

Para informações detalhadas sobre cada classe desse subsistema, ver **Apêndice B.1**.

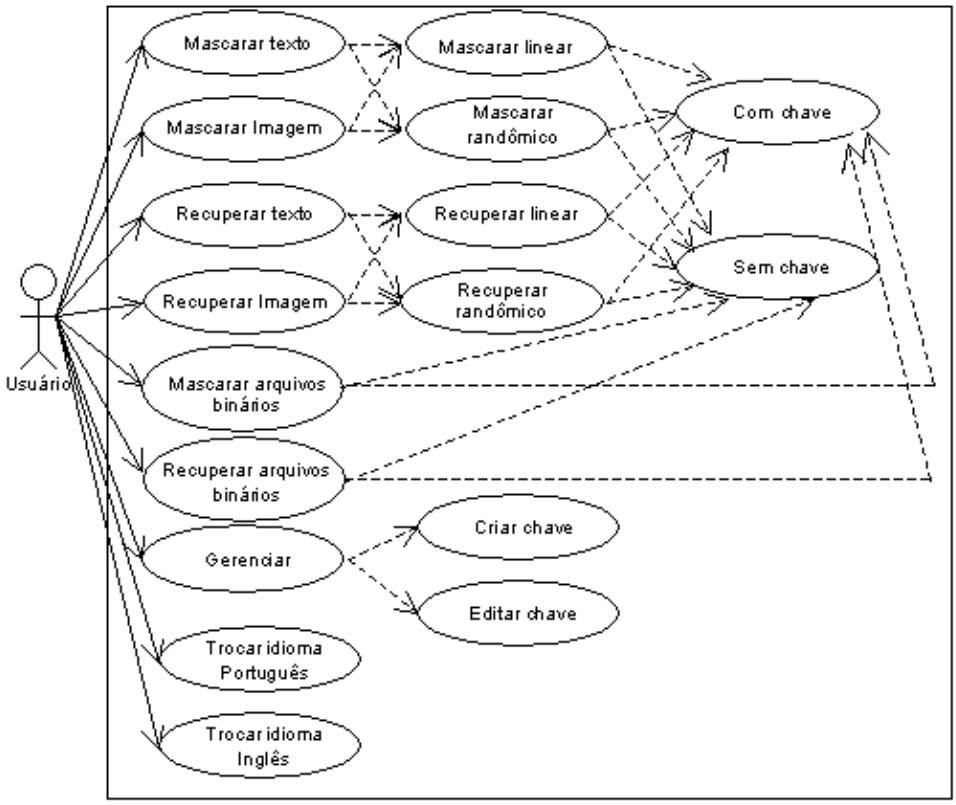

Figura 7.19: Casos de uso do sistema

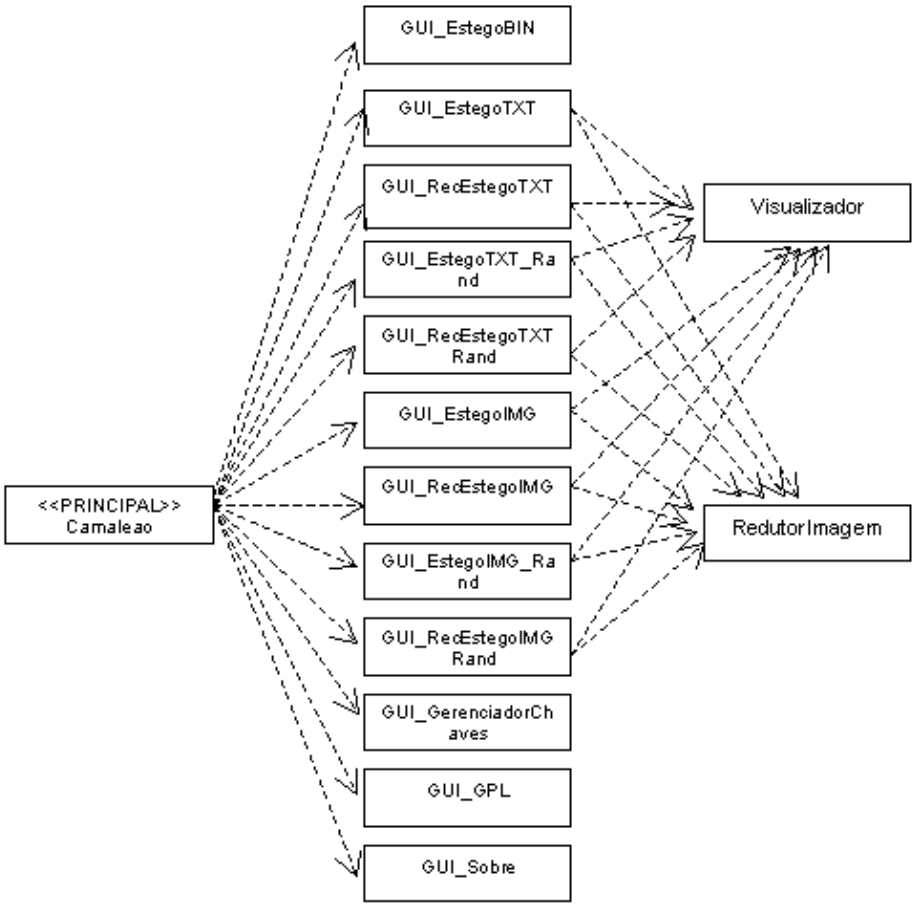

Figura 7.20: Classes do subsistema de IHM

7.8.3 Diagrama de classes do subsistema de LN

A figura 7.21 apresenta as classes presentes no subsistema de *Lógica de Negócios*.

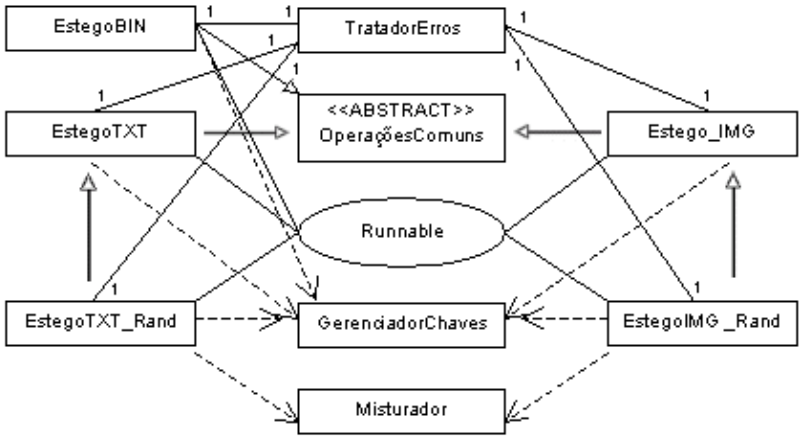

Figura 7.21: Classes do subsistema de LN

Para informações detalhadas sobre cada classe desse subsistema, ver **Apêndice B.2**.

7.8.4 Middleware

Como visto, o projeto **Camaleão** foi desenvolvido em duas camadas, o que corresponde a dois subsistemas, respectivamente. O subsistema *Interface Homem-Máquina*, figura 7.22, e o subsistema *Lógica de Negócios*, figura 7.23. A comunicação entre estes dois subsistemas é feita pelo subsistema *Middleware* que contém apenas duas classes, a classe *Tarefa*[7] e a classe *SwingWorker*[8].

7.8.5 Diagrama de atividades do subsistema de IHM

O funcionamento do subsistema de *Interface Homem-Máquina* (IHM) pode ser acompanhado através dos diagramas apresentados nas figuras 7.24 a 7.31.

[7] Para mais detalhes sobre esta classe, ver Apêndice B.3.

[8] Esta classe serve para trabalhar com *Threads* dedicadas em uma interface gráfica. Para um maior detalhamento, consulte `http://java.sun.com/docs/books/tutorial/uiswing/misc/threads.html`

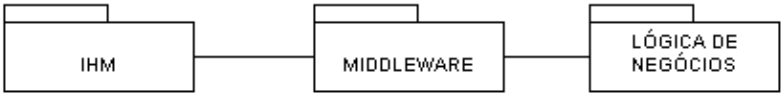

Figura 7.22: Subsistemas do **Camaleão**

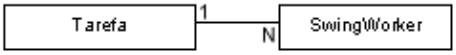

Figura 7.23: Subsistema *Middleware*

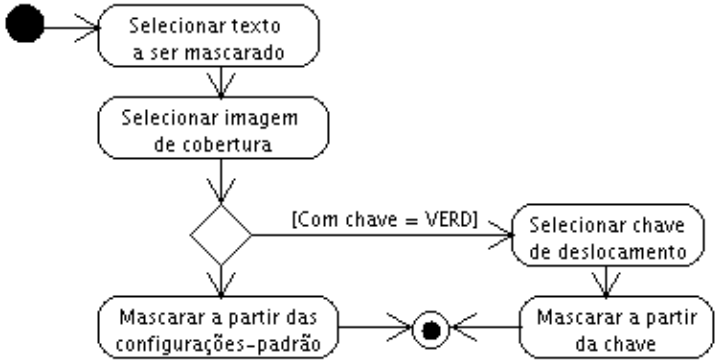

Figura 7.24: Mascarar texto

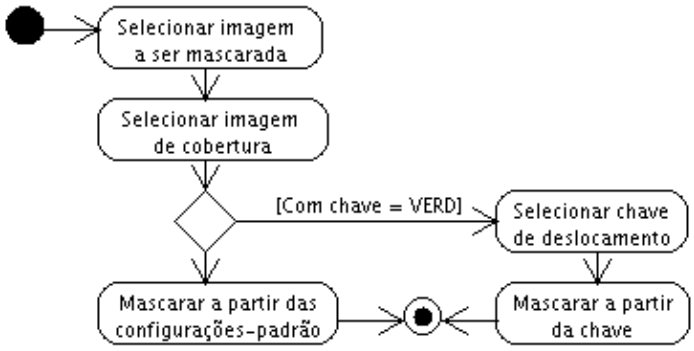

Figura 7.25: Mascarar imagem

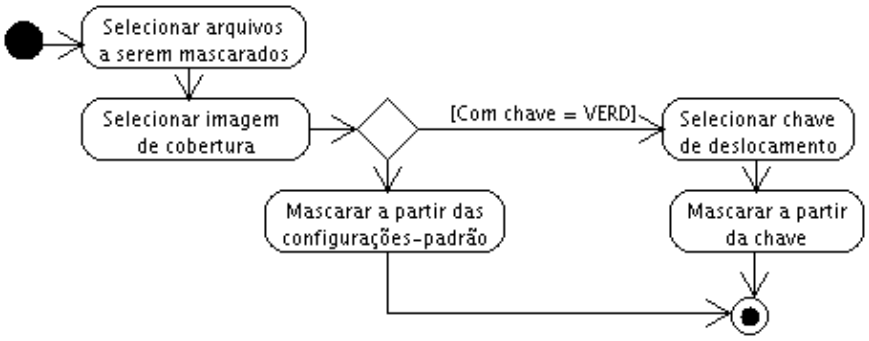

Figura 7.26: Mascarar lista de arquivos binários

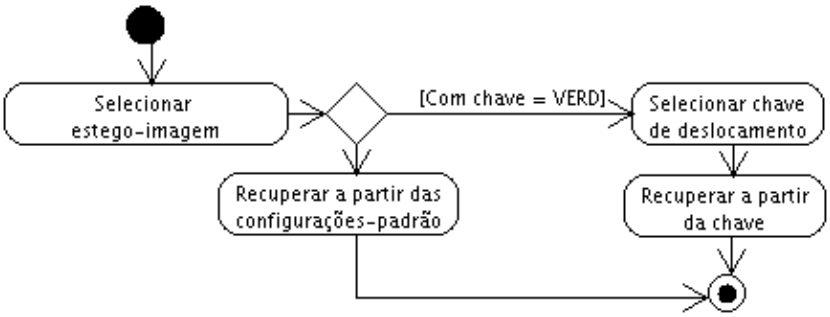

Figura 7.27: Recuperar texto

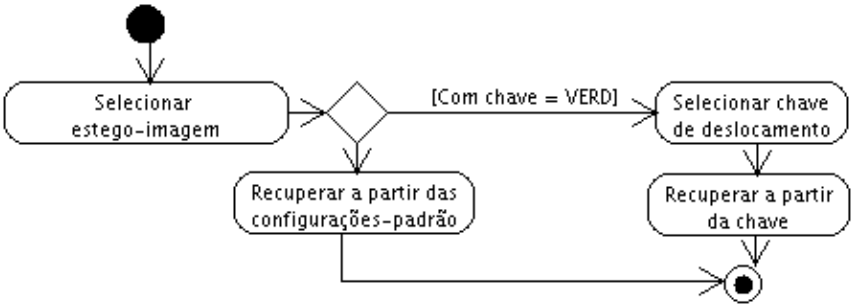

Figura 7.28: Recuperar imagem

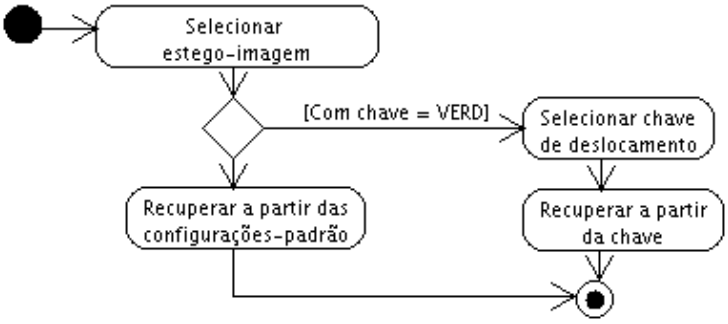

Figura 7.29: Recuperar lista de arquivos binários

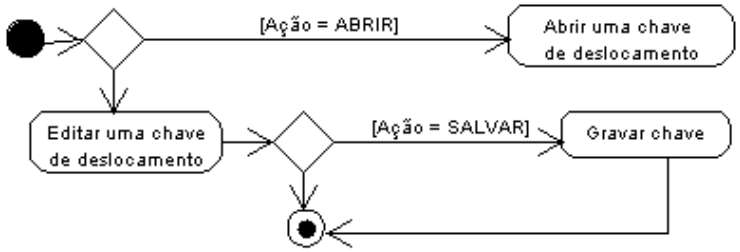

Figura 7.30: Gerenciar chaves

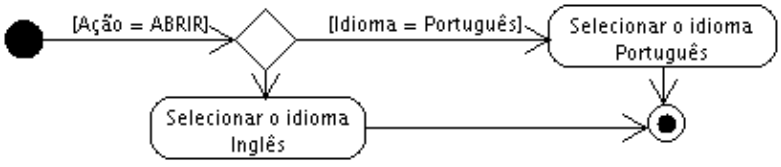

Figura 7.31: Selecionar idioma

7.8.6 Diagrama de atividades do subsistema de LN

O funcionamento do subsistema de *Lógica de Negócios* (LN) pode ser acompanhado através dos diagramas apresentados nas figuras 7.32 a 7.40.

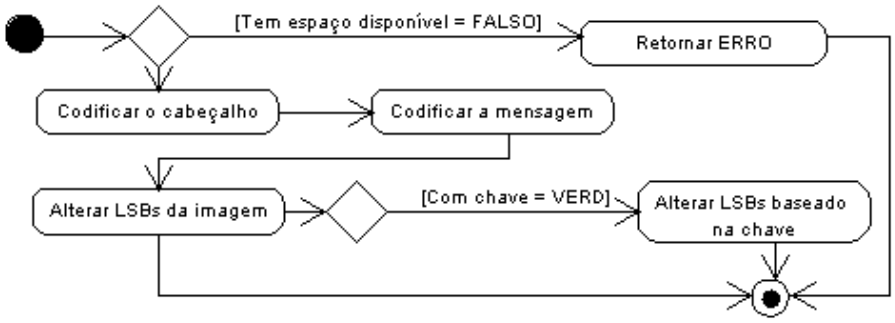

Figura 7.32: Mascarar textos, imagens e arquivos binários de forma linear

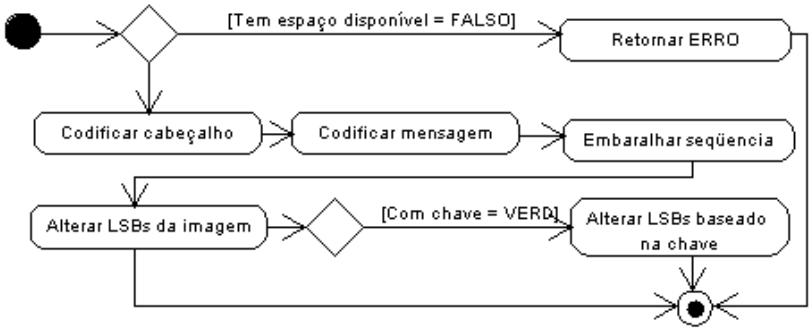

Figura 7.33: Mascarar textos e imagens de forma randômica

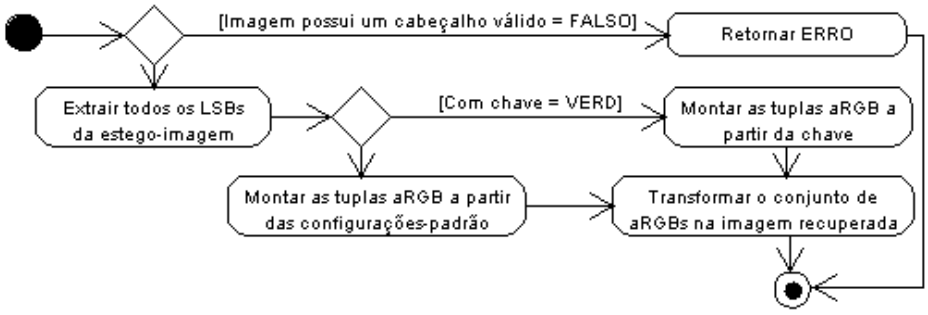

Figura 7.34: Recuperar imagem de forma linear

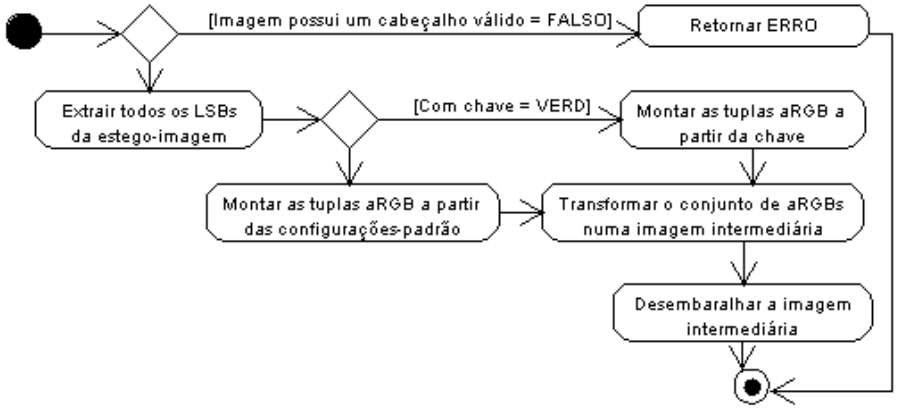

Figura 7.35: Recuperar imagem de forma randômica

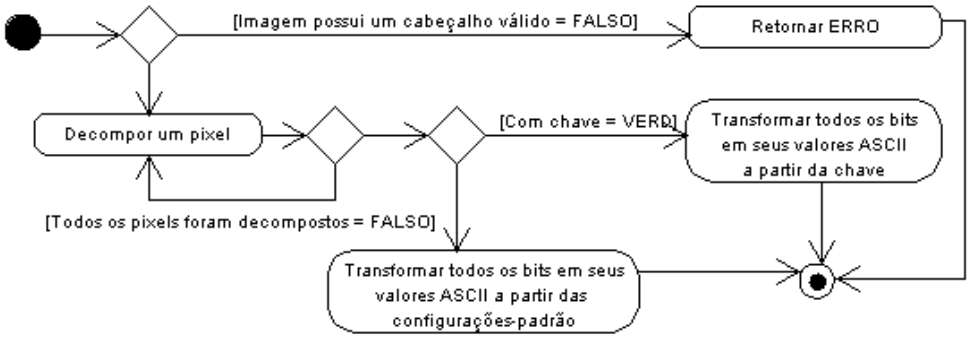

Figura 7.36: Recuperar texto de forma linear

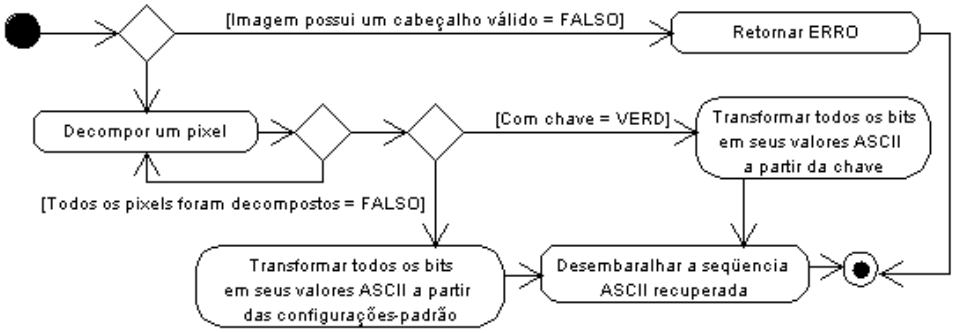

Figura 7.37: Recuperar texto de forma randômica

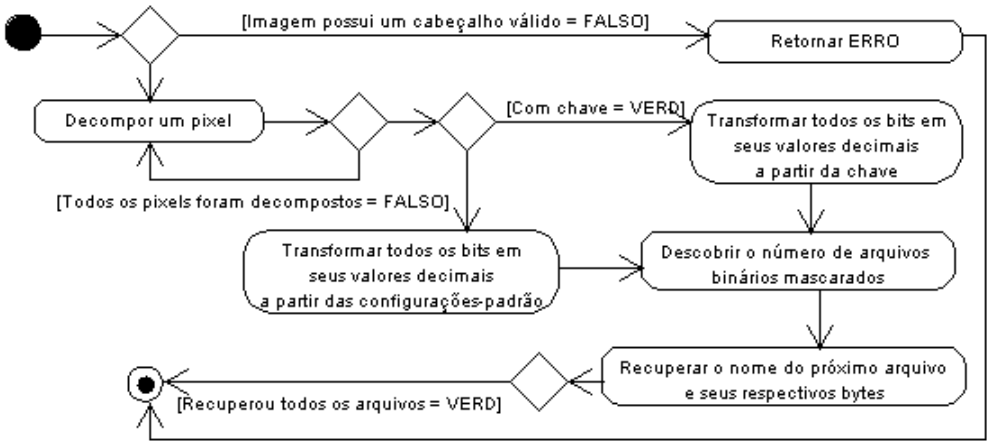

Figura 7.38: Recuperar arquivos binários

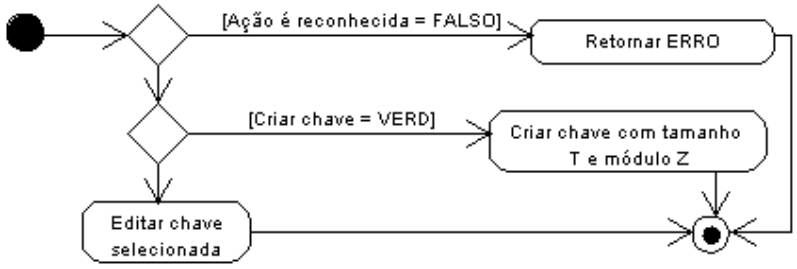

Figura 7.39: Gerenciar chaves de deslocamento

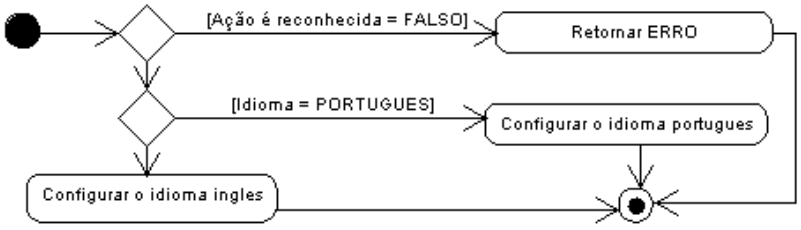

Figura 7.40: Selecionar idioma

Capítulo 8

Resultados e discussão – esteganálise

8.1 Considerações iniciais

Como visto na seção 6.4, neste trabalho utilizou-se o *RS-Esteganálise* para verificar a presença ou ausência de mensagens escondidas após a codificação feita pelo **Camaleão**.

A seguir, apresentam-se os resultados obtidos. A análise foi feita apenas sobre o mascaramento de imagens, uma vez que não há perda de generalidade. Tanto as imagens de cobertura quanto as estego-imagens utilizadas foram de 32 bpp (*bits* por *pixel*).

Ao analisar as imagens produzidas pelo **Camaleão**, verificou-se que a *condição 1* da seção 6.4 ($R_M - R_{-M} = i$ e i é muito grande) é válida para todos os casos em que se analisa estego-imagens. Isto quer dizer que, diferenças entre R_M e $R_{-M} \leq$ 2,5% não configuram um mascaramento detectável. Isto é, caso $R_M - R_{-M} \leq 2,5\%$, o método não é capaz de dizer se houve ou não um mascaramento. No entanto, com a *condição 2* ($R_M - S_M = k$ e k é muito grande) verificou-se o contrário: em imagens típicas, a diferença entre R_M e S_M é muito grande e em estego-imagens a diferença converge para zero. Isto quer dizer que, nas estego-imagens analisadas, o valor de k encontrado foi menor que 25% do total de grupos. Como o objetivo do *RS-Esteganálise* é apenas dizer se há ou não uma mensagem escondida em uma imagem, basta atentar a este detalhe e o método continuará funcionando. As discrepâncias podem ter sido causadas por uma interpretação diferente das definições. O fato dos resultados terem sido contrários ao método de [Fridrich et al., 2001] não invalida a análise.

8.2 Análise 1

Para esta análise utilizou-se a imagem de cobertura IC_1, figura 8.1.

Figura 8.1: Imagem de cobertura 1 (IC_1)

IC_1 tem 1024x768 *pixels* totalizando 786432 *pixels* o que resulta em 294912 *bytes* disponíveis para o mascaramento.

Para IC_1, foram feitos três testes. Todos os testes utilizaram a imagem alvo IA_1, figura 8.2. A máscara M utilizada foi $M = [0, 1, 1, 0]$ e $-M = [0, -1, -1, 0]$.

Figura 8.2: Imagem alvo IA_1

1. **Teste 1:** IA_1 foi mascarada com o tamanho 100x75 totalizando 7500 *pixels* a serem mascarados. Isto resultou em 30000 *bytes* ou uma mudança aproximada de 9% nos LSBs de IC_1.

2. **Teste 2:** IA_1 foi mascarada com o tamanho 200x150 totalizando 30000 *pixels* a serem mascarados. Isto resultou em 120000 *bytes* ou uma mudança aproximada de 40% nos LSBs de IC_1.

3. **Teste 3:** IA_1 foi mascarada com o tamanho 310x233 totalizando 72000 *pixels* a serem mascarados. Isto resultou em 288920 *bytes* ou uma mudança aproximada de 98% nos LSBs de IC_1.

Análise da imagem de cobertura IC_1

Os resultados aparecem na tabela 8.1.

Classificação	**Número de grupos**	**Percentagem dos grupos**
R_M	94129	48,0
S_M	15764	8,7
R_{-M}	94452	48,0
S_{-M}	15273	8,4

Tabela 8.1: Análise da imagem de cobertura

1. $R_M - S_M \approx$ 39,3%

2. $R_M - R_{-M} \approx$ 0%

Pela verificação 2, conclui-se que é muito provável que esta imagem não possua uma mensagem escondida.

Teste 1

Os resultados aparecem na tabela 8.2.

Classificação	Número de grupos	Percentagem dos grupos
R_M	87102	44,0
S_M	19825	10,9
R_{-M}	90427	46,0
S_{-M}	18701	10,3

Tabela 8.2: Teste 1 – Alteração de $\approx 9\%$ dos LSBs da imagem de cobertura

1. $R_M - S_M \approx$ 34,9%
2. $R_M - R_{-M} \approx$ 2%

Pelas verificações não é possível afirmar que esta imagem contém um mascaramento dado que a $R_M - R_{-M}$ não foi maior que 2,5%.

Teste 2

Os resultados aparecem na tabela 8.3.

Classificação	Número de grupos	Percentagem dos grupos
R_M	77711	39,5
S_M	28159	14,0
R_{-M}	85726	47,3
S_{-M}	25946	14,3

Tabela 8.3: Teste 2 – Alteração de $\approx 40\%$ dos LSBs da imagem de cobertura

1. $R_M - S_M \approx$ 25,5%
2. $R_M - R_{-M} \approx$ 7,8%

Pela segunda verificação, é altamente provável que esta imagem contenha uma mensagem escondida.

Teste 3

Os resultados aparecem na tabela 8.4.

Classificação	Número de grupos	Percentagem dos grupos
R_M	60130	33,0
S_M	42241	23,3
R_{-M}	76761	42,3
S_{-M}	38410	21,2

Tabela 8.4: Teste 3 – Alteração de $\approx 98\%$ dos LSBs da imagem de cobertura

1. $R_M - S_M \approx$ 10,3%
2. $R_M - R_{-M} \approx$ 9,3%

Pela segunda verificação, é altamente provável que esta imagem contenha uma mensagem escondida. A primeira constatação confirma a probabilidade.

8.3 Análise 2

Para esta análise, utilizou-se a imagem de cobertura IC_2, figura 8.3.

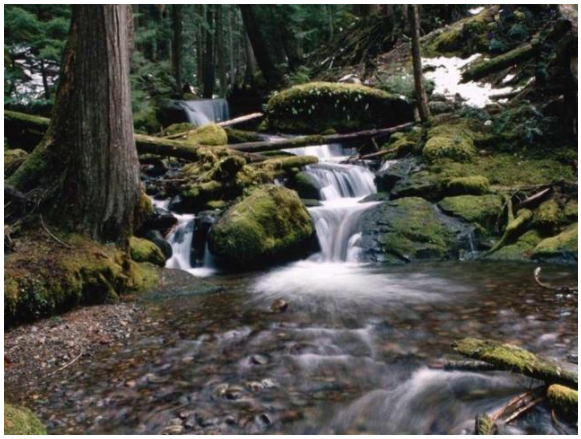

Figura 8.3: Imagem de cobertura 2 (IC_2)

IC_2 tem 1024x768 *pixels* totalizando 786432 *pixels* o que resulta em 294912 *bytes* disponíveis para o mascaramento.

Para IC_2 foram feitos três testes. Todos os testes utilizaram a imagem alvo IA_2, figura 8.4. A máscara M utilizada foi $M = [0, 1, 1, 0]$ e $-M = [0, -1, -1, 0]$. Como segunda verificação, utilizou-se uma máscara alternativa de confirmação $M = [1, 0, 1, 0]$ e $-M = [-1, 0, -1, 0]$

Figura 8.4: Imagem alvo IA_2

A imagem IA_2 antes de ser mascarada passou por um processo de permutações sucessivas. O resultado aparece na figura 8.5 sendo esta a imagem utilizada no mascaramento.

Figura 8.5: Imagem alvo IA_2 randomizada

1. **Teste 1:** IA_2 foi mascarada com o tamanho 80x60 totalizando 4800 *pixels* a serem mascarados. Isto resultou em 19200 *bytes* ou uma mudança aproximada de 6,5% nos LSBs de IC_2.

2. **Teste 2:** IA_2 foi mascarada com o tamanho 240x168 totalizando 40320 *pixels* a serem mascarados. Isto resultou em 161280 *bytes* ou uma mudança aproximada de 54,7% nos LSBs de IC_2.

3. **Teste 3:** IA_2 foi mascarada com o tamanho 310x235 totalizando 72850 *pixels* a serem mascarados. Isto resultou em 291400 *bytes* ou uma mudança aproximada de 99% nos LSBs de IC_2.

Análise da imagem de cobertura IC_2

Os resultados aparecem nas tabelas 8.5 e 8.6.

Classificação	Número de grupos	Percentagem dos grupos
R_M	75812	38,5
S_M	20330	10,4
R_{-M}	77777	39,5
S_{-M}	18800	9,5

Tabela 8.5: Análise da imagem de cobertura. $M = [0, 1, 1, 0]$

1. $R_M - S_M \approx$ 28,1%
2. $R_M - R_{-M} \approx$ 1,0%

Classificação	Número de grupos	Percentagem dos grupos
R_M	128824	65,5
S_M	67784	34,5
R_{-M}	130680	66,5
S_{-M}	65928	33,5

Tabela 8.6: Análise da imagem de cobertura. $M = [1, 0, 0, 1]$

1. $R_M - S_M \approx$ 31,5%
2. $R_M - R_{-M} \approx$ 1,0%

Independentemente das máscaras utilizadas, é altamente provável que IC_2 não possua uma mensagem escondida.

Teste 1

Os resultados aparecem nas tabelas 8.7 e 8.8.

Classificação	Número de grupos	Percentagem dos grupos
R_M	74508	37,9
S_M	21125	10,7
R_{-M}	77014	39,2
S_{-M}	19461	9,9

Tabela 8.7: Teste 1 – Alteração de $\approx 6,5\%$ dos LSBs da imagem de cobertura. $M = [0, 1, 1, 0]$

1. $R_M - S_M \approx$ 27,2%
2. $R_M - R_{-M} \approx$ 1,3%

Classificação	Número de grupos	Percentagem dos grupos
R_M	127707	64,9
S_M	68901	35,0
R_{-M}	129871	66,0
S_{-M}	66737	33,9

Tabela 8.8: Teste 1 – Alteração de $\approx 6,5\%$ dos LSBs da imagem de cobertura. $M = [1, 0, 1, 0]$

1. $R_M - S_M \approx$ 29,9%
2. $R_M - R_{-M} \approx$ 1,1%

Independentemente da máscara utilizada, não é possível afirmar que IC_2 possui uma mensagem escondida.

Teste 2

Os resultados aparecem nas tabelas 8.9 e 8.10.

Classificação	Número de grupos	Percentagem dos grupos
R_M	60092	30,5
S_M	28200	14,4
R_{-M}	70723	35,9
S_{-M}	26969	13,7

Tabela 8.9: Teste 2 – Alteração de $\approx 54,7\%$ dos LSBs da imagem de cobertura. $M = [0, 1, 1, 0]$

1. $R_M - S_M \approx$ 16,1%
2. $R_M - R_{-M} \approx$ 5,5%

Classificação	Número de grupos	Percentagem dos grupos
R_M	116167	59,0
S_M	80441	40,9
R_{-M}	131069	66,6
S_{-M}	75539	38,4

Tabela 8.10: Teste 2 – Alteração de $\approx 54,7\%$ dos LSBs da imagem de cobertura. $M = [1, 0, 1, 0]$

1. $R_M - S_M \approx$ 19,9%
2. $R_M - R_{-M} \approx$ 7,6%

Independentemente da máscara utilizada, é altamente provável que IC_2 possua uma mensagem escondida

Teste 3

Os resultados aparecem nas tabelas 8.11 e 8.12.

Classificação	Número de grupos	Percentagem dos grupos
R_M	45921	23,4
S_M	35021	17,8
R_{-M}	67029	34,1
S_{-M}	33402	17,0

Tabela 8.11: Teste 3 – Alteração de $\approx 99\%$ dos LSBs da imagem de cobertura. $M = [0, 1, 1, 0]$

1. $R_M - S_M \approx$ 5,6%
2. $R_M - R_{-M} \approx$ 10,7%

Classificação	Número de grupos	Percentagem dos grupos
R_M	104983	53,4
S_M	91625	46,6
R_{-M}	123683	62,9
S_{-M}	82925	42,3

Tabela 8.12: Teste 3 – Alteração de $\approx 99\%$ dos LSBs da imagem de cobertura. $M = [1, 0, 1, 0]$

1. $R_M - S_M \approx$ 6,8%
2. $R_M - R_{-M} \approx$ 9,5%

Independentemente da máscara utilizada, é altamente provável que IC_2 possua uma mensagem escondida

8.4 Considerações finais

Como observado nas análises feitas, o *RS-Esteganálise* não conseguiu identificar mensagens escondidas produzidas pelo **Camaleão** quando o tamanho destas é inferior a 10% do total de *bytes* disponíveis para o mascaramento. No entanto, quando o tamanho destas mensagens aumenta, a eficiência do RS na detecção

também aumenta. Além disso, o processo de permutações sucessivas não deixa as estego-imagens resultantes mais robustas dado que o RS têm praticamente a mesma eficiência na detecção de mensagens escondidas nestas imagens. É importante lembrar, no entanto, que o procedimento de permutações sucessivas (mistura dos dados a serem escondidos) é muito eficiente para driblar muitos outros métodos de inferência estatística, como por exemplo o teste do χ^2.

Capítulo 9

Considerações finais

Neste capítulo, são apresentadas as principais conclusões deste trabalho de pesquisa, as contribuições para a sociedade e algumas propostas de trabalhos futuros.

9.1 Conclusões

Este trabalho apresentou a evolução da *esteganografia* ao longo da história e suas aplicações modernas com a chamada *esteganografia digital*. Foram mostradas as principais técnicas de mascaramento e, em especial, mascaramento em imagens. Também foi mostrada a ferramenta *Camaleão: um software para segurança digital utilizando esteganografia* desenvolvido durante o trabalho. Finalmente, fez-se uma análise das estego-imagens produzidas pela ferramenta segundo o método de esteganálise proposto por [Fridrich et al., 2001].

A *esteganografia*, quando bem utilizada, fornece meios eficientes e eficazes na busca por proteção digital. Associando *criptografia* e *esteganografia*, as pessoas têm em mãos o poder de comunicar-se em segredo pela rede mundial de computadores mantendo suas identidades íntegras e secretas. Obviamente, a privacidade pode ser aproveitada com fins ilícitos como apresentado no capítulo 4. No entanto, o papel do autor deste trabalho, enquanto cientista, é fazer ciência para ajudar a sociedade. Dependerá da sociedade saber aplicar o conhecimento da forma correta.

9.2 Contribuições

Acredita-se que este trabalho tenha contribuído substancialmente para a sociedade como um todo.

A ferramenta **Camaleão**, ao ser disponibilizada em código aberto, adquire um imenso potencial didático, dando a qualquer interessado a possibilidade de aprofundar-se um pouco mais em *esteganografia digital*. A seguir, outras contribuições:

- Desenvolvimento de um *site* responsável por agregar conhecimentos relacionados à pesquisa.
- Divulgação tanto na imprensa falada quanto escrita:
  - Entrevista à EPTV, emissora afiliada à Rede Globo de Televisão, em 02/10/2003;
  - Entrevista ao jornal *Estado de Minas* em 30/10/2003;
  - Entrevista ao *Jornal da UFLA* em 21/11/2003.
- Seminários e palestras:
  - Alunos do curso de Ciência da Computação da Universidade Federal de Lavras (UFLA) em 04/11/2003;
  - *I Encontro Regional de Computação* (ERCOMP) e *V Semana de Informática* (SEMINFO) em 06/11/2003 em Formiga, MG;
  - Alunos de pós-graduação *Strictu Sensu* (mestrado e doutorado) em Estatística do departamento de Ciências Exatas da Universidade Federal de Lavras (UFLA) em 24/11/2003.

9.3 Trabalhos futuros

O campo de pesquisas em *esteganografia digital* está em constante evolução. Novas técnicas são inovadas a cada dia. Neste sentido, apresentam-se a seguir algumas melhorias que poderiam ser desenvolvidas e adequadas ao **Camaleão**.

9.3.1 Códigos corretores de erros

Um dos grandes problemas da *esteganografia* em imagens consiste em recuperar a mensagem escondida após um ataque geométrico à estego-imagem. Uma das saídas possíveis é aplicar códigos corretores de erro que possibilitem a recuperação da mensagem sem a necessidade de todos os *bits* estarem presentes no lado receptor. Dado que alguns *bits* tenham sido perdidos durante o ataque geométrico, é possível tirar certas conclusões a partir dos códigos corretores.

Uma maneira de desenvolver esta proposta é utilizando os códigos de *hamming* [Wayner, 2002].

Outra forma de utilizar a correção de erros é mascarar os *bits* através de erros propositalmente inseridos. Um *bit* com erro é um *bit* que esconde outro *bit*.

9.3.2 Compactação

Como dito na seção 5.4, quanto menor a mensagem a ser escondida mais difícil será detectá-la em caso de interceptação da imagem que a cobre. Desta forma, propõe-se a implementação da *compactação de Huffmann*, [Wayner, 2002], para mensagens textuais e a *compactação* JPEG, [The JPEG Group, 2003], para as imagens.

9.3.3 Criptografia

Um sistema *esteganográfico* torna-se bastante seguro se associado à *criptografia*. A figura 9.1 apresenta esta possibilidade.

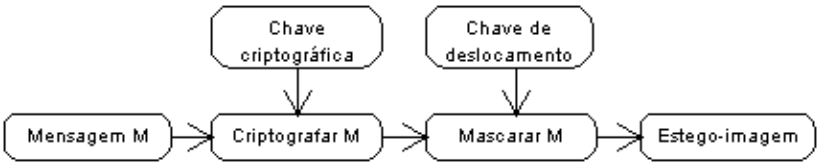

Figura 9.1: Associando a *criptografia* e a *esteganografia*

Uma mensagem, antes de ser mascarada, é criptografada segundo uma chave. Logo após, a partir de uma chave de deslocamento, escolhe-se os *bits* menos significativos (LSBs) que devem ser alterados na imagem de cobertura. Propõe-se a utilização da *criptografia* como um passo adicional antes de efetuar o mascaramento das mensagens na ferramenta desenvolvida.

9.3.4 Outros formatos de imagens

Atualmente, o **Camaleão** produz estego-imagens de 32 bpp (*bits* por *pixel*) no formato PNG [The PNG Group, 2003]. Isto quer dizer que o mascaramento só é possível em imagens cuja compactação é do tipo *lossless* — sem perda de dados —. Conseqüentemente, as imagens são maiores que a maioria das imagens em circulação pela *Internet*. Como uma forma de reduzir as imagens produzidas, propõe-se a possibilidade de mascarar mensagens também em imagens passíveis

de compactação *lossy* — com possível perda de dados —. Note que esta é uma tarefa extremamente complexa e, na maioria das vezes, deve ser aplicada em conjunto com códigos corretores de erros. Além disso, as imagens produzidas têm maior propensão a revelar o conteúdo escondido se estiverem sob um ataque de *esteganálise*.

9.3.5 Padrões estatísticos da imagem de cobertura

Como dito no capítulo 6, a análise estatística é um dos pilares da *esteganálise*. Uma das maneiras de aumentar a robustez do sistema desenvolvido é implementar uma função de imitação (*mimicry function*) responsável por analisar a imagem de cobertura, armazenar seus padrões estatísticos e mascarar a mensagem a partir dos padrões descobertos. Desta forma, após o mascaramento, a imagem produzida tem seus padrões estatísticos pouco alterados. Isto torna a estego-imagem altamente capaz de sobrepujar ataques estatísticos.

9.3.6 Chaves de deslocamento

Um dos principais problemas da *criptografia* é a distribuição de chaves. Não basta ter uma chave altamente segura caso esta ainda deva ser compartilhada. O desenvolvimento de um sistema criptográfico baseado em chave pública tal como o sistema originalmente proposto por Diff-Helmann-Ritchie e posteriormente aplicado ao RSA [Schneier, 1995], torna-se altamente seguro e configurável. O desenvolvimento de um sistema *esteganográfico* semelhante seria uma revolução. Caso algo assim seja descoberto, os desenvolvedores estarão ricos.

9.3.7 Geradores pseudo-aleatórios

O **Camaleão** faz intenso uso de geradores pseudo-aleatórios. Esses geradores tendem a cair em repetição ou mesmo gerar seqüências viciadas — não uniformemente distribuídas —. Propõe-se a sua substituição por geradores por geradores criptográficos pseudo-aleatórios. Esta substituição irá produzir seqüências uniformente distribuídas dando maior robustez ao produto desenvolvido.

9.3.8 Mascaramento de sons

Todas as técnicas implementadas trabalham com arquivos de imagens. Poderia-se extender o **Camaleão** de modo a adaptá-lo para mascarar mensagens em arquivos de sons. Visto que o funcionamento de um arquivo de som é muito parecido com um arquivo de imagem, esta proposta não é tão complexa quanto possa parecer.

Caso isso seja feito, as mensagens mascaradas poderiam ser maiores, uma vez que os arquivos de sons são, na maioria dos casos, maiores que arquivos de imagens.

Referências Bibliográficas

[Ambler, 1998] Ambler, S. W. (1998). *Análise e Projeto Orientado a Objetos*. IBPI Press. ISBN 8-57331-036-7.

[Anderson and Petitcolas, 1998] Anderson, R. J. and Petitcolas, F. A. P. (1998). On the limits of steganography. In *IEEE Journal of Selected Areas in Communications*. Special issue on Copyright & Privacy Protection.

[Artz, 2001] Artz, D. (2001). Digital steganography: hiding data within data. In *IEEE Internet Computing*.

[Aura, 1996] Aura, T. (1996). Practical invisibility in digital communication. In *HUT Seminar on Network Security*. Helsink Univerisity of Technology.

[Booch et al., 1999] Booch, G., Rumbaugh, J., and Jacobson, I. (1999). *The Unified Modeling Language User Guide*. Addison-Wesley Pub Co, Boston, USA. ISBN 0-20157-168-4.

[Cass, 2003] Cass, S. (2003). Listening in. In *IEEE Spectrum*, volume 40, pages 32–37.

[Clelland et al., 1999] Clelland, C. T., Risca, V., and Bancroft, C. (1999). Hiding messages in dna microdots. *Nature*.

[Costa, 2001] Costa, H. A. X. (2001). Diretrizes de manutenibilidade para a construção do modelo de projeto orientado a objetos. In *Exame de Qualificação de Tese de Doutoramento*. Escola Politécnica da Universidade de São Paulo – Departamento de Engenharia Elétrica.

[Deitel and Deitel, 2001] Deitel, H. M. and Deitel, P. J. (2001). *Java como programar*. Bookman Editora, Porto Alegre. Tradução de Edson Furnankiewicz.

[Fridrich et al., 2001] Fridrich, J., Goljan, M., and Du, R. (2001). Detecting lsb steganography in color and grayscale images. In *IEEE Proceeding on Multimedia and Security*. IEEE Multimedia.

[Fridrich et al., 2002] Fridrich, J., Goljan, M., and Du, R. (2002). Reliable detection of lsb steganography in color and grayscale images. *Center for Intelligent Systems, SUNY Binghamton*.

[Johnson and Jajodia, 1998] Johnson, N. and Jajodia, S. (1998). Exploring steganography: seeing the unseen. In *IEEE Internet Computing*.

[Judge, 2001] Judge, J. C. (2001). Steganography: Past, present, future. In *Proceedings of the First International Information-Hiding Workshop*. The SANS Institute. Último acesso em 15 de maio de 2003.

[Kahn, 1996] Kahn, D. (1996). *The CODEBREAKERS: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Scribner, Boston. ISBN 0-68483-130-9.

[Katzenbeisser and Petitcolas, 2000] Katzenbeisser, S. and Petitcolas, F. A. (2000). *Information hiding, techniques for steganography and digital watermarking*. Artech House, Boston.

[Kolata, 2003] Kolata, G. (2003). A mystery unraveled, twice. Disponível em `cryptome.unicast.org/cryptome022401/tri.crack.htm`. Acessado em 15 de abril de 2003.

[Kuhn, 1996] Kuhn, M. G. (1996). The history of steganography. In *Proceedings of the First International Information-Hiding Workshop*. Springer–Verlag, Berlim.

[Kumagai, 2003] Kumagai, J. (2003). Mission impossible? In *IEEE Spectrum*, volume 40, pages 26–31.

[EFF, 2003] EFF (2003). Eff – the eletronic frontier foundation. Disponível em `www.eff.org`.

[FSF, 2003] FSF (2003). Fsf – free software foundation. Disponível em `www.fsf.org`.

[NHTCU, 2003] NHTCU (2003). Nhtcu – national high tech crime unit. Disponível em `www.nhtcu.org`.

[Sun Microsystems, 2003] Sun Microsystems (2003). The java documentation. Disponível em `java.sun.com`.

[The JPEG Group, 2003] The JPEG Group (2003). Specification of the jpeg image format. In *The Joint Expert Photographic Group*. The JPEG Group. Último acesso em 19 de novembro de 2003.

[The PNG Group, 2003] The PNG Group (2003). Specification of the png image format. In *The Libpng Group*. The PNG Group. Último acesso em 25 de outubro de 2003.

[USPS, 2003] USPS (2003). Usps – us postal inspection service. Disponível em `www.usps.com/postalinspectors/ar01intr.pdf`.

[Mintzer et al., 1996] Mintzer, F. C., Boyle, L. E., and Cases, A. N. (1996). Toward on-line, worldwide access to vatican library materials. In *IBM Journal of Research and Development*, volume 40.

[Norman, 1980] Norman, B. (1980). *Secret warfare, the battle of Codes and Ciphers*. Acropolis Books Inc.

[Petitcolas et al., 1999] Petitcolas, F. A., Anderson, R. J., and Kuhn, M. G. (1999). Information hiding - a survey. In *Proceedings of IEEE*. Special issue on Protection on multimedia content.

[Pfitzmann, 1996] Pfitzmann, B. (1996). Information hiding terminology. In *Proceedings of the first international information-hiding workshop*. Springer–Verlag, Berlim.

[Popa, 1998] Popa, R. (1998). An analysis of steganography techniques. Master's thesis, Department of Computer Science and Software Engineering of The "Polytechnic" University of Timisoara, Timisoara, Romênia.

[Schneier, 1995] Schneier, B. (1995). *Applied Cryptography*. John Wiley & Sons, New York. ISBN 0-47111-709-9.

[Singh, 2001] Singh, S. (2001). *O livro dos códigos*. Record, Rio de Janeiro. ISBN 8-50105-598-0.

[Tzschoppe et al., 2003] Tzschoppe, R., B*ä*uml, R., Huber, J. B., and Kaup, A. (2003). Steganographic system based on higher-order statistics. *Center for Intelligent Systems, SUNY Binghamton*.

[Wallich, 2003] Wallich, P. (2003). Getting the message. In *IEEE Spectrum*, volume 40, pages 38–43.

[Wayner, 2002] Wayner, P. (2002). *Disappearing cryptography*. Morgan Kaufmann Publishers, San Francisco. ISBN 1-55860-769-2.

[Westfeld and Pfitzmann, 2003] Westfeld, A. and Pfitzmann, A. (2003). Attacks on steganographics systems. *Department of Computer Science – Dresden University of Technology*.

Apêndice A

Detalhamento dos casos de uso

A seguir, são apresentados os casos de uso do sistema em mais detalhes.

ID caso de uso: 001

Nome: Mascarar texto

Casos de uso utilizados: Mascarar linear, Mascarar randômico
Ator(es): Usuário

Descrição: Permite o mascaramento de um texto em uma imagem de cobertura produzindo uma estego-imagem. O mascaramento pode ser linear ou randômico. Além disso, pode ou não ser baseado em chave de deslocamento.

Execução normal:
- O usuário escolhe um texto
- O usuário escolhe uma imagem de cobertura
- O usuário realiza o mascaramento
- O produto de *software* gera a estego-imagem
- O usuário escolhe um nome de arquivo para armazenar a estego-imagem

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma imagem de cobertura
  - O usuário escolheu uma imagem de cobertura não suportada
  - O usuário escolheu uma mensagem maior que o espaço disponível para o mascaramento

ID caso de uso: 002

Nome: Mascarar imagem

Casos de uso utilizados: Mascarar linear, Mascarar randômico

Ator(es): Usuário

Descrição: Permite o mascaramento de uma imagem em uma imagem de cobertura produzindo uma estego-imagem. O mascaramento pode ser linear ou randômico. Além disso, pode ou não ser baseado em chave de deslocamento.

Execução normal:
- O usuário escolhe uma imagem
- O usuário escolhe uma imagem de cobertura
- O usuário realiza o mascaramento
- O produto de *software* gera a estego-imagem
- O usuário escolhe um nome de arquivo para armazenar a estego-imagem

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma imagem de cobertura
  - O usuário escolheu uma imagem de cobertura não suportada
  - O usuário escolheu uma imagem-alvo não suportada
  - O usuário escolheu uma imagem-alvo maior que o espaço disponível para o mascaramento

ID caso de uso: 003

Nome: Mascarar arquivos binários

Casos de uso utilizados: Mascarar linear
Ator(es): Usuário

Descrição: Permite o mascaramento de uma lista de arquivos quaisquer em uma imagem de cobertura produzindo uma estego-imagem. O mascaramento é realizado de forma linear. Além disso, pode ou não ser baseado em chave de deslocamento.

Execução normal:
- O usuário escolhe uma lista de arquivos
- O usuário escolhe uma imagem de cobertura
- O usuário realiza o mascaramento
- O produto de *software* gera a estego-imagem
- O usuário escolhe um nome de arquivo para armazenar a estego-imagem

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma imagem de cobertura
  - O usuário escolheu uma imagem de cobertura não suportada
  - O usuário não escolheu uma lista de arquivos
  - O usuário escolheu uma lista de arquivos maior que o espaço disponível para o mascaramento

ID caso de uso: 004

Nome: Recuperar texto

Casos de uso utilizados: Recuperar linear, Recuperar randômico

Ator(es): Usuário

Descrição: Permite a recuperação de um texto mascarado em uma estego-imagem. A recuperação pode ser linear ou randômica. Além disso, pode ou não ser baseado em chave de deslocamento

Execução normal:
- O usuário escolhe uma estego-imagem
- O usuário realiza a recuperação
- O produto de *software* encontra o texto mascarado
- O usuário escolhe um nome de arquivo para armazenar o texto encontrado

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma estego-imagem
  - O usuário escolheu uma estego-imagem não suportada
  - O usuário escolheu uma estego-imagem sem nada escondido

ID caso de uso: 005

Nome: Recuperar imagem

Casos de uso utilizados: Recuperar linear, Recuperar randômico

Ator(es): Usuário

Descrição: Permite a recuperação de uma imagem mascarada em uma estego-imagem. A recuperação pode ser linear ou randômica. Além disso, pode ou não ser baseado em chave de deslocamento.

Execução normal:
- O usuário escolhe uma estego-imagem
- O usuário realiza a recuperação
- O produto de *software* encontra a imagem mascarada
- O usuário escolhe um nome de arquivo para armazenar a imagem encontrada

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma estego-imagem
  - O usuário escolheu uma estego-imagem não suportada
  - O usuário escolheu uma estego-imagem sem nada escondido

ID caso de uso: 006

Nome: Recuperar arquivos binários

Casos de uso utilizados: Recuperar linear

Ator(es): Usuário

Descrição: Permite a recuperação de uma lista de arquivos quaisquer mascarada em uma estego-imagem. A recuperação é realizada de forma linear. Além disso, pode ou não ser baseado em chave de deslocamento

Execução normal:
- O usuário escolhe uma estego-imagem
- O usuário realiza a recuperação
- O produto de *software* encontra os arquivos mascarados
- O produto de *software* grava cada arquivo encontrado em uma pasta temporária do disco

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma estego-imagem
  - O usuário escolheu uma estego-imagem não suportada
  - O usuário escolheu uma estego-imagem sem nada escondido

ID caso de uso: 007

Nome: Mascarar randômico

Casos de uso utilizados: Com chave, Sem chave

Ator(es): Usuário

Descrição: Habilita o mascaramento randômico de um objeto de entrada (texto ou imagem a ser mascarado). O processo pode ou não ser baseado em chave de deslocamento.

Execução normal:
• Antes de executar o **Caso de Uso 001** ou o **Caso de Uso 002**, o usuário escolhe o mascaramento randômico.

Execução anormal:
• O produto de *software* irá retornar uma mensagem de erro se:
  – O usuário não escolheu uma imagem de cobertura
  – O usuário escolheu uma imagem de cobertura não suportada
  – O usuário escolheu uma lista de arquivos maior que o espaço disponível para o mascaramento

ID caso de uso: 008

Nome: Mascarar linear

Casos de uso utilizados: Com chave, Sem chave

Ator(es): Usuário

Descrição: Habilita o mascaramento linear de um objeto de entrada (texto ou imagem a ser mascarado). O processo pode ou não ser baseado em chave de deslocamento.

Execução normal:
• Antes de executar o **Caso de Uso 001** ou o **Caso de Uso 002**, o usuário escolhe o mascaramento linear.

Execução anormal:
• O produto de *software* irá retornar uma mensagem de erro se:
  – O usuário não escolheu uma imagem de cobertura
  – O usuário escolheu uma imagem de cobertura não suportada
  – O usuário escolheu uma lista de arquivos maior que o espaço disponível para o mascaramento

ID caso de uso: 009

Nome: Recuperar linear

Casos de uso utilizados: Com chave, Sem chave

Ator(es): Usuário

Descrição: Habilita a recuperação linear de um objeto de entrada (estego-imagem). O processo pode ou não ser baseado em chave de deslocamento.

Execução normal:
• Antes de executar o **Caso de Uso 003** ou o **Caso de Uso 004**, o usuário escolhe a recuperação linear.

Execução anormal:
• O produto de *software* irá retornar uma mensagem de erro se:
  – O usuário não escolheu uma estego-imagem
  – O usuário escolheu uma estego-imagem não suportada
  – O usuário escolheu uma estego-imagem sem nada escondido

ID caso de uso: 010

Nome: Recuperar randômico

Casos de uso utilizados: Com chave, Sem chave

Ator(es): Usuário

Descrição: Habilita a recuperação linear de um objeto de entrada (estego-imagem). O processo pode ou não ser baseado em chave de deslocamento.

Execução normal:
• Antes de executar o **Caso de Uso 003** ou o **Caso de Uso 004**, o usuário escolhe a recuperação randômica.

Execução anormal:
• O produto de *software* irá retornar uma mensagem de erro se:
  – O usuário não escolheu uma estego-imagem
  – O usuário escolheu uma estego-imagem não suportada
  – O usuário escolheu uma estego-imagem sem nada escondido

ID caso de uso: 011

Nome: Sem chave

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite executar um mascaramento (linear ou randômico) ou uma recuperação (linear ou randômica) de um texto ou de uma imagem independente de uma chave de deslocamento.

Execução normal:
• No momento em que o usuário realizar o mascaramento ou a recuperação de um texto ou de uma imagem, ele opta em efetuar esta operação sem utilizar uma chave de deslocamento.

Execução anormal:
• Não há erros relacionados a esse **Caso de Uso**

ID caso de uso: 012

Nome: Com chave

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite executar um mascaramento (linear ou randômico) ou uma recuperação (linear ou randômica) de um texto ou de uma imagem utilizando uma chave de deslocamento.

Execução normal:
• No momento em que o usuário realizar o mascaramento ou a recuperação de um texto ou de uma imagem, ele opta em efetuar esta operação utilizando uma chave de deslocamento.

Execução anormal:
• O produto de *software* irá retornar uma mensagem de erro se:
  – O usuário não escolheu uma chave de deslocamento
  – O usuário escolheu uma chave de deslocamento não suportada

ID caso de uso: 013

Nome: Gerenciar

Casos de uso utilizados: Criar chave, Editar chave

Ator(es): Usuário

Descrição: Permite o gerenciamento de uma chave de deslocamento. Novas chaves podem ser criadas. Além disso, chaves já criadas podem ser editadas.

Execução normal:
• O usuário abre uma chave de deslocamento
• O produto de *software* exibe a chave selecionada
• O usuário então pode executar o **Caso de Uso 014** ou o **Caso de Uso 015**

Execução anormal:
• Não há erros relacionados a esse **Caso de Uso**

ID caso de uso: 014

Nome: Criar chave

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite a criação de novas chaves de deslocamento. As chaves podem ser de tamanhos diferentes bem como atuar em módulos numéricos configuráveis

Execução normal:
- O usuário escolhe o tamanho da chave a ser gerada
- O usuário escolhe o módulo de atuação da chave
- O produto de *software* gera a chave de deslocamento
- O produto de *software* exibe a chave de deslocamento

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O produto de *software* não encontrou *bits* aleatórios suficientes para gerar uma chave de deslocamento

ID caso de uso: 015

Nome: Editar chave

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite a edição de chaves de deslocamento existentes.

Execução normal:
- O usuário escolhe uma chave de deslocamento
- O produto de *software* exibe a chave escolhida
- O usuário edita a chave
- O usuário grava a chave modificada

Execução anormal:
- O produto de *software* irá retornar uma mensagem de erro se:
  - O usuário não escolheu uma chave de deslocamento
  - O usuário escolheu uma chave de deslocamento não suportada

ID caso de uso: 016

Nome: Trocar idioma Português

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite a troca do idioma base do produto de *software*, selecionando o idioma português.

Execução normal:
- O usuário seleciona o idioma *Português* como padrão
- O produto de *software* troca o idioma padrão para o português

Execução anormal:
- Não há erros relacionados a esse **Caso de Uso**

ID caso de uso: 017

Nome: Trocar idioma Inglês

Casos de uso utilizados: —

Ator(es): Usuário

Descrição: Permite a troca do idioma base do produto de *software*, selecionando o idioma inglês.

Execução normal:
• O usuário seleciona o idioma *Inglês* como padrão
• O produto de *software* troca o idioma padrão para o inglês

Execução anormal:
• Não há erros relacionados a esse **Caso de Uso**

Apêndice B

Detalhamento das classes

B.1 Classes do subsistema de IHM

A seguir, encontram-se as classes apresentadas no diagrama de classes da figura 7.20. Estas classes estão presentes no subsistema de *Interface Homem-Máquina* (IHM).

Camaleão
JDesktopPane desktop; # static String idioma; # String titulo; # String msgs[], tooltips[]; # char mnemonicos[]; # Font fonteArial10_B; # Font fonteArial10_N; # JMenuBar barraMenus;
+ static void main(String args[]) + Camaleao(String idioma) - void configurarIdioma() - JMenuBar criarMenus() - void atualizarListaJanelasInternas()

GUI_EstegoTXT
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Timer timer
+ GUI_EstegoTXT(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_EstegoIMG
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Timer timer
+ GUI_EstegoIMG(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_EstegoBIN
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Vector listaArquivos - Timer timer
+ GUI_EstegoBIN(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_EstegoTXT_Rand
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Timer timer
+ GUI_EstegoTXT_Rand (String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_EstegoIMG_Rand
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Timer timer
+ GUI_EstegoTXT_Rand (String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_RecEstegoTXT
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - Timer timer, timerAtz
+ GUI_RecEstegoTXT(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_RecEstegoIMG
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - Timer timer, timerAtz - BufferedImage imagemRecuperada
+ GUI_RecEstegoTXT(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_RecEstegoTXT_Rand
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - BufferedImage estegoImagem - Timer timer, timerAtz
+ GUI_RecEstegoTXT_Rand (String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_RecEstegoIMG_Rand
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - Timer timer, timerAtz - BufferedImage imagemRecuperada
+ GUI_RecEstegoTXT(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

GUI_RecEstegoBIN
- static int contadorDeInstancias - Tarefa tarefa - TratadorErros tratadorErros - Timer timer, timerAtz - Vector listaArquivosRecuperados
+ GUI_RecEstegoBIN(String idioma) - JPanel configurarElementosInterface() - Border criarBorda(String titulo) - void configurarIdioma() - Timer criarTemporizador()

B.2 Classes do subsistema de LN

A seguir, encontram-se as classes apresentadas no diagrama de classes da figura 7.21. Estas classes estão presentes no subsistema de *Lógica de Negócios* (LN).

TratadorErros
- String avisos[] - String warnings[]
+ TratadorErros() + String getErro(int tipoErro) + String getErro(int tipoErro, String informacoesAdicionais) + String getTituloErro(int tipoErro)

OperaçõesComuns
- File refImg
+ Tarefa() + static void finalizar() + boolean configurarMascaramentoTexto(String nomeImgCobertura, String mensagem, String nomeChave) + boolean configurarRecuperacaoTexto(String nomeEstegoImg, String nomeChave) + boolean configurarMascaramentoImagem(String nomeImgCobertura, String imgMensagem, String chave) + boolean configurarRecuperacaoImagem(String nomeEstegoImg, String nomeChave) + void configurarGeradorChave(int tamanhoSeq, int z) + BufferedImage getImagemSaida() + String getMensagemSaida() + String getChaveProduzida() + boolean terminou() + void iniciar(int tipoTarefa) - void setMensagemSaida(String str) - void setImagemSaida(BufferedImage img) - void setChaveProduzida(String seqChave)

GerenciadorChaves
- TratadorErros tratadorErros
+ void run() + int[] gerarSequenciaChaves(int tamanhoSeq, int z) + String gerarSequenciaChavesStr(int tamanhoSeq, int z) + void gravarSequenciaEmArquivo(int sequencia[], String nomeArquivo) + int[] lerSequenciaDeArquivo(String nomeArquivo)

Misturador
—
+ Misturador() + BitSet[] embaralharSequencia(BitSet bytes[], int chave[], int offset) + BitSet[] desembaralharSequencia(BitSet bytes[], int chave[], int offset) + BufferedImage embaralharSequencia(BufferedImage imagem, int chave[]) + BufferedImage desembaralharSequencia(BufferedImage imagem, int chave[]) + int[] embaralharSequencia(int sequencia[], int chave[]) + int[] desembaralharSequencia(int bytes[], int chave[], int offset) + String embaralharSequencia(String seq, int chave[]) + String desembaralharSequencia(String seq, int chave[]) - int getSementeSequencia(int sequencia[]) - int[] getSequenciaTroca(int semente, int tamanhoSequencia, int z, int offset)

EstegoTXT
BitSet bytesOrigem[] # final String cabecalho = “STEGTXT#” # final String eof = “##” # int tamanhoMsg # int marcadorBits - TratadorErros tratadorErros
+ EstegoTXT() + void run() + BufferedImage mascararTexto(BufferedImage imgCobertura, String msg, int chave[]) + String decodificarMensagem(BufferedImage estegoImg, int chave[]) # boolean temEspacoDisponivel(int chave[], int nroBitsDisponiveis) # void mapearTextoEmBits(String msg) # BitSet toBits(int num) # void codificarCabecalho() # void codificarMensagem(String msg) # void alterarLSBs(BufferedImage imagem, int chave[]) # String paraDecimal(BitSet bits)

EstegoTXT_Rand
- TratadorErros tratadorErros
+ EstegoTXT_Rand() + BufferedImage mascararTexto(BufferedImage imgCobertura, String msg, int chave[]) + String decodificarMensagem(BufferedImage estegoImg, int chave[])

EstegoIMG
int marcadorBits # BitSet bytesOrigem[] # final String cabecalho = “STEGIMG#” # final String eof = “##” # final TratadorErros tratadorErros
+ EstegoIMG() + void run() + BufferedImage mascararImagem(BufferedImage imgCobertura, BufferedImage imgAlvo, int chave[]) + BufferedImage decodificarMensagem(BufferedImage estegoImg, int chave[]) # void mapearImagemEmBits(BufferedImage imagem) # boolean temEspacoDisponivel(int chave[], int nroBitsDisponiveis, int tamanhoMsg) # void mapearImagemEmBits(BufferedImage imagem) # BitSet toBits(int num) # void codificarCabecalho() # void codificarMensagem(BufferedImage imagem) # void alterarLSBs(BufferedImage imagem, int chave[]) # BitSet extrairBits(BufferedImage estegoImg, int chave[]) # int[] getRGBs(BitSet bits) # BufferedImage paraImagem(int bytes[])

EstegoIMG_Rand
- TratadorErros tratadorErros
+ EstegoIMG_Rand() + BufferedImage mascararImagem(BufferedImage imgCobertura, BufferedImage imgAlvo, int chave[]) + BufferedImage decodificarMensagem(BufferedImage estegoImg, int chave[])

EstegoBIN
BitSet bytesOrigem[] # String cabecalho = “STEGBIN#” # String eof = “##” # int tamanhoMsg # int marcadorBits # TratadorErros tratadorErros
+ EstegoBIN() + void run() + BufferedImage mascararBytes(BufferedImage imgCobertura, String msg, int chave[]) + Vector decodificarBytes(BufferedImage estegoImg, int chave[]) - Vector getIndice(int v1[], int v2[], int nroEsperadoOcorrencias) # boolean temEspacoDisponivel(int chave[], int nroBitsDisponiveis) # void mapearTextoEmBits(String msg) # BitSet toBits(int num) # void codificarCabecalho() # void codificarMensagem(String msg) # void alterarLSBs(BufferedImage imagem, int chave[]) # int[] decodificarMensagem(BufferedImage estegoImg, int chave[]) # int[] paraDecimal(BitSet bits)

B.3 Classes do subsistema de *Middleware*

A seguir, encontra-se a classe *Tarefa*, apresentada no diagrama de classes das figuras 7.22 e 7.23. Esta classe está presente no subsistema de *Middleware*.

Tarefa
EstegoTXT obj1 # EstegoIMG obj2 # GeradorChave obj3 # EstegoBIN obj4 # BufferedImage imgCobertura, estegoImg, imgMensagem # Vector listaArquivosRecuperados # String msg, msgSaida # int chave[], tipoTarefa, tamanhoSeq, z # BufferedImage imgSaida # static boolean emExecucao # String aviso, warning, strChave # TratadorErros tratadorErros
+ Tarefa() + static void finalizar() + boolean configurarMascaramentoTexto(String nomeImgCobertura, String mensagem, String nomeChave) + boolean configurarRecuperacaoTexto(String nomeEstegoImg, String nomeChave) + boolean configurarMascaramentoImagem(String nomeImgCobertura, String nomeImgMensagem, String nomeChave) + boolean configurarRecuperacaoImagem(String nomeEstegoImg, String nomeChave) + void configurarGeradorChave(int tamanhoSeq, int z) + BufferedImage getImagemSaida() + String getMensagemSaida() + String getChaveProduzida() + boolean terminou() + void iniciar(int tipoTarefa) - void setMensagemSaida(String str) - void setImagemSaida(BufferedImage img) - void setChaveProduzida(String seqChave) - class Acao1, Acao2, Acao3, Acao4, Acao5, Acao6, Acao7, Acao8, Acao9, Acao10