

CAMALEÃO: UM SOFTWARE DE ESTEGANOGRAFIA PARA PROTEÇÃO E PRIVACIDADE DIGITAL

Anderson Rocha*
Siome Goldenstein

Instituto de Computação
Universidade Estadual de Campinas
13083-970, Campinas - SP
{anderson.rocha, siome}@ic.unicamp.br

Heitor Costa
Lucas Chaves

Depto. de Ciência da Computação
Universidade Federal de Lavras
37200-000, Lavras - MG
{heitor, lucas}@ufla.br

RESUMO

A busca por novos meios eficientes e eficazes de proteção digital é um campo de pesquisas fundamentado nas mais variadas áreas da ciência. A esteganografia configura-se como uma 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 software capaz de prover a comunicação pela internet por fazer uso de técnicas esteganográficas tais como cifragem por blocos e permutações cíclicas em imagens digitais.

ABSTRACT

Digital protection is a research area which needs efficient ways to make it possible. The steganography is configured as one of these electronic protection ways. 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. Therefore, the main objective of work was to develop a software that allows communication on the internet by using steganographic techniques such as cypher blocks and cyclic permutations in digital images.

1 INTRODUÇÃO

A busca por novos meios eficientes e eficazes de proteção digital é um campo de pesquisas 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. *Esteganografia* deriva do grego, sendo *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 (POPA, 1998).

A *esteganografia* inclui um vasto conjunto de métodos para comunicações secretas desenvolvidos ao longo da história. Atualmente, trabalha-se na es-

truturação 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 os *e-mails* dos usuários de computador podem estar sendo lidos e os seus passos 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.

Este artigo apresenta as principais técnicas de *esteganografia* da atualidade disponibilizadas em um *software*. O principal objetivo do trabalho foi aumentar a robustez das técnicas existentes. Para isso, criou-se uma nova técnica *esteganográfica*, unindo a força da cifragem de blocos do algoritmo criptográfico DES (SCHNEIER, 1995) e um conjunto de permutações cíclicas, às técnicas existentes.

A Seção 2 apresenta os principais termos utilizados, a Seção 3 apresenta o estado da arte listando as mais importantes técnicas de *esteganografia* em imagens e alguns dos mais conhecidos *softwares* do gênero atualmente. O funcionamento do *algoritmo* de criptografia DES é apresentado na Seção 4. A Seção 5

*Bolsista CNPq 2001-2003, Capes 2004 semestre 1 e Fapesp 2004-2006.

apresenta o *Camaleão*, um software para proteção digital utilizando esteganografia, desenvolvido pelos autores. Finalmente, a Seção 6 apresenta algumas propostas de extensão do trabalho e a Seção 7 discute sobre algumas considerações finais.

2 TERMINOLOGIA

No modelo geral de ocultamento de dados (*information hiding*), o dado embutido (*embedded data*) é a mensagem que se deseja enviar de maneira secreta. Frequentemente, 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 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*) que é uma mensagem inócua contendo secretamente uma mensagem de maior importância. A Figura 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 (PETITCOLAS et al., 1999). Uma estego-chave

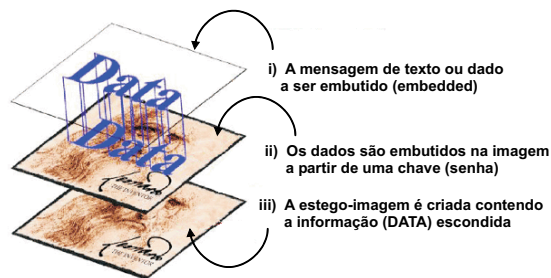


Figura 1: Ocultamento de uma mensagem

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

3 ESTADO DA ARTE

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 técnicas pode ser aplicada às imagens, com graus variados de sucesso. O método de inserção no *bit* menos significativo é provavelmente uma das técnicas mais utilizadas de *esteganografia* em imagem. O *Camaleão* utilizou essencialmente esta técnica para o mascaramento dos dados.

3.1 Inserção LSB

Técnicas baseadas em LSB (*Least Significant Bit*) 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 (POPA, 1998; PETITCOLAS et al., 1999; WAYNER, 2002).

Acompanhe o exemplo da Figura 2 para entender melhor. Suponha que se deseja esconder a letra **E** dentro da porção de imagem. Na Figura 2, têm-se

```
(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 2: Porção de uma imagem de cobertura

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. Na

```
(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 3: Porção da estego-imagem gerada pela porção de imagem 2

Figura 3, os *bits* destacados pelo quadrado representam as modificações necessárias nos LSBs para esconder a letra **E**.

3.2 Filtragem e mascaramento

Técnicas de filtragem e mascaramento são mais robustas que a inserção LSB no sentido de gerarem estego-imagens imunes a técnicas de compressão e recorte. Ao contrário das modificações LSB, estas técnicas 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 ruído¹ tornando as informações detectáveis.

3.3 Algoritmos e transformações

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 conhecidas (POPA, 1998; JOHNSON; JAJODIA, 1998).

3.4 Softwares disponíveis

Aplicações de *esteganografia* estão disponíveis na *internet* para executar em uma grande variedade de plataformas incluindo DOS, Windows, Mac OS e Unix/Linux.

*Ezstego*² e *Stego Online*³ são duas ferramentas escritas na linguagem de programação *Java* e limitadas a imagens indexadas de oito *bits* e em formato GIF. Estas duas ferramentas foram desenvolvidas por Romana Machado.

Henry Hastur desenvolveu duas outras ferramentas: *Mandelsteg* e *Stealth*⁴. *Mandelsteg* gera imagens de fractais para esconder as mensagens. *Stealth* é um programa que recebe uma mensagem criptografada pelo PGP, retira seu cabeçalho e a esconde em um arquivo. A retirada do cabeçalho de identificação é importante pois geralmente este contém informações do tipo de método criptográfico utilizado.

Duas outras ferramentas capazes de trabalhar em associação com a *criptografia* são *Ray Arachelian's White Noise Storm*⁵ e o *S-Tools*⁶.

Colin Maroney desenvolveu o *Hide and Seek*⁷. Esta ferramenta é capaz de mascarar uma lista de arquivos em uma imagem, mas não faz uso de *cripto-*

grafia. Niels Provos desenvolveu o *Outguess*⁸ capaz de esconder mensagens com alto grau de robustez em vários formatos de arquivos de imagens. Testes estatísticos de primeira ordem não são capazes de detectar mensagens mascaradas com este *software* (PROVOS; HONEYMAN, 2003; WAYNER, 2002).

Finalmente, outra ferramenta de destaque é *Jpeg-Jsteg*⁹ capaz de fazer o mascaramento de informações utilizando os *pixels* mais significativos de uma imagem *jpg*.

3.5 Esteganálise

Grande parte das técnicas de *esteganografia* possuem falhas e/ou inserem artefatos (padrões) detectáveis nos objetos de cobertura. Algumas vezes, basta um indivíduo interessado em descobrir indevidamente a mensagem fazer um exame mais detalhado destes artefatos para descobrir que há mensagens escondidas. No entanto, quando o processo de mascaramento de informações foi robusto, as tentativas de recuperar 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 cifras.

4 O ALGORITMO DES

O DES (*Data Encryption Standard*) é um criptosistema simétrico que opera em blocos de 64 bits usando uma chave de 56 bits (a chave que o algoritmo recebe como entrada tem 64 bits, porém 8 bits são de paridade). O DES é um cifrador de Feistel, possuindo 16 rodadas, cada rodada usando uma subchave de 48 bits gerada a partir da chave inicial. O mesmo algoritmo é usado tanto para cifrar quanto para decifrar. Em cada rodada é aplicada uma função, chamada função da rodada, que transforma os bits da entrada através de permutações e substituições, usando as subchaves como parâmetros (SCHNEIER, 1995).

O espaço de chaves do DES tem tamanho 2^{56} , sendo algumas chaves consideradas fracas. Não se considera este tamanho seguro para o poder computacional atual, então é comum a utilização do Triplo DES (3DES) com três chaves, K_1, K_2, K_3 , no esquema EDE: para um bloco de texto a cifrar x , o ciframento é dado por $E_{K_3}(D_{K_2}(E_{K_1}(x)))$, onde E é a aplicação de ciframento com DES e D é a aplicação de deciframento com DES.

¹Ruído 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<http://www.stego.com>

³<http://www.stego.com>

⁴<ftp://idea.sec.dsi.unimi.it/pub/security/>

⁵<ftp://csua.berkeley.edu/pub/cypherpunks/>

⁶<ftp://idea.sec.dsi.unimi.it/pub/security/>

⁷<ftp://csua.berkeley.edu/pub/cypherpunks/>

⁸<http://www.outguess.org/>

⁹<ftp://funet.fi/pub/crypt/steganography>

Sobre modos de operação: modos de operação define como cifrar um texto claro com tamanho maior do que o tamanho do bloco n de um criptosistema simétrico. O mais simples é o modo ECB (*electronic-codebook*), em que o bloco de texto claro é dividido em blocos de tamanho n , sendo cada bloco cifrado separadamente. O problema com este modo de operação é que blocos de texto claro idênticos geram blocos de texto cifrado idênticos. Uma alternativa é o modo CBC (*cipher-block chaining*), que efetua uma operação XOR do bloco de texto claro atual com o bloco de texto cifrado anterior. O modo ECB é recomendado apenas para textos de tamanho pequeno.

5 RESULTADOS E DISCUSSÃO

Como resultado deste trabalho, foi desenvolvido o *Camaleão: um software para proteção digital utilizando esteganografia*. Para mais informações, consulte (ROCHA, 2003).

Camaleão é um *software* que permite a comunicação pela *internet* através do uso da *esteganografia*. O *software* possui várias características tais como:

- *ambiente multiplataforma:* por ter sido desenvolvido na linguagem de programação Java (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. Para isso, o usuário deve 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, português e inglês;
- *código aberto:* disponibilizado sob a licença de uso GPL (*General Public GNU Licence*) ou licença pública geral GNU (FSF, 2003), o *Camaleão* pode ser modificado e utilizado livremente desde que sejam mantidas intactas as referências aos autores originais;
- *tipos de mascaramento e de recuperação:* o *Camaleão* permite o mascaramento de qualquer tipo de arquivo dentro de outras imagens. As imagens de cobertura podem ser de extensão *jpg* ou *png*. A imagem de saída (contendo o mascaramento) tem a extensão *png*. O processo de mascaramento, linear ou aleatório, 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 não¹⁰. Nas seções a seguir, encontram-se mais detalhes a respeito de cada processo;

- *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 diferentes¹¹. O principal método de mascaramento utilizado foi desenvolvido pelos próprios autores. Este método é caracterizado pela divisão dos dados a serem mascarados em vários blocos que são cifrados com o algoritmo criptográfico DES e ciclicamente permutados de acordo com a chave de deslocamento gerada. Finalmente, os dados são escondidos na imagem de cobertura. Uma maior explicação deste método é dada na Seção 5.3.2.

A Figura 4 apresenta uma tela do *Camaleão*. As



Figura 4: Mascaramento de um texto

imagens produzidas após o processo de mascaramento são perceptualmente idênticas. Os seres humanos conseguem capturar mudanças em uma imagem quando estas ocorrem em um fator acima de 3% (WAYNER, 2002). No caso, como o *Camaleão* trabalha apenas com o *bit* menos significativo de um dado *pixel*, o conjunto total de mudanças, considerando 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

¹⁰De forma geral, para cada entrada (*bit*) de uma mensagem a ser mascarada existe uma entrada (*deslocamento*) respectiva(o) 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.

¹¹Chaves de módulo k têm todas as suas entradas pertencentes ao intervalo $\{0, k - 1\}$.

sobe para 1,56%, ainda imperceptível à maioria dos seres humanos (WAYNER, 2002).

A Figura 5, de 133,2 KB, apresenta a *estego-imagem* resultante em que 35% dos seus LSBs foram alterados pelo processo de mascaramento feito pelo *Camaleão*. Não houve danos à imagem. Um observador não é capaz de afirmar com certeza se esta imagem possui ou não possui uma mensagem escondida.



Figura 5: Exemplo de estego-imagem – 133,2 KB

5.1 O processo de mascaramento

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 $\{0, \dots, m\}$, sendo m 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 é válido (SCHNEIER, 1995). Sem o conhecimento da sucessão correta de distâncias entre os *bits*, um indivíduo terá poucas chances de êxito ao tentar recuperar a mensagem escondida.

A Figura 6 descreve o processo de mascaramento. O emissor modifica o conjunto de *bits* da mensagem 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. O conjunto

Stream original	Stream a ser mascarado	Chave (distâncias)	Stream a ser enviado
00110110	0	0	00110110
00100111	1	0	00100111
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 6: O processo de mascaramento segundo uma chave de deslocamento

de *bits* a ser mascarado pode ser um texto plano¹², 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 *stream* de *bits*¹³.

Exemplo: mascaramento de um conjunto de *bits*. O nome *conjunto de bits* (*streams de bits*) refere-se à maneira como o sistema interpreta os arquivos neste processo, *byte* a *byte*, isto é, independentemente da entrada (qualquer que seja a semântica da mesma), tudo é uma *seqüência* de *bits*. Esta funcionalidade permite o mascaramento de uma lista de até 256 arquivos de uma só vez dentro de uma imagem¹⁴. O procedimento acontece de acordo com a Figura 7. Dado um conjunto de *bits* contendo vários arquivos a serem escondidos, cria-se o cabeçalho de acordo com a equação 1. Tendo a mensagem pronta para o mascaramento, tudo passa a ser interpretado como a mensagem a ser eficientemente mascarada pelo sistema através do processo descrito na Seção 5.3.2. Nas definições a seguir dois arquivos são escondidos em uma imagem. O cabeçalho STEGBIN# refere-se ao processo de mascaramento de um *stream* de *bits*.

¹²Texto contendo apenas caracteres. Normalmente arquivo com extensão .TXT.

¹³*Stream de bits* aqui refere-se a qualquer conjunto de *bits* que represente alguma informação útil computacionalmente. Ex.: arquivos, programas entre outros.

¹⁴Isto porque apenas um *byte* foi utilizado para representar, no cabeçalho inserido, o número de arquivos mascarados em uma estego-imagem

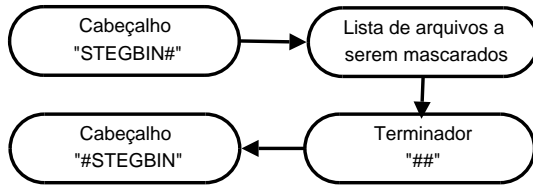


Figura 7: Mascaramento de arquivos

- $C = \text{STEGBIN\#}$ (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)
- $S = \text{\$}$ (Separador de campos)
- $TA_1 =$ Tamanho ou número de *bytes* de N_1
- $TA_2 =$ Tamanho ou número de *bytes* de N_2
- $T = \text{\#\#}$ (Terminador de mensagens)

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

$$M = C + NA + S + N_1 + S + TA_1 + S + N_2 + S + TA_2 + S + B_1 + B_2 + T, \quad (1)$$

Uma vez montada a mensagem, o mascaramento ocorre similarmente ao processo mostrado na Figura 7. A posição de cada *bit* é dada por uma chave de deslocamento ou pelas configurações-padrão, caso não haja uma chave. Caso o mascaramento esteja utilizando as medidas de robustez propostas neste trabalho, a mensagem M montada é submetida a todo o processo descrito na Seção 5.3.2 e só após este pré-processamento ocorrerá o mascaramento segundo mostra a Figura 7.

5.2 O processo de recuperação

Neste processo, os LSBs 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 8 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.

Exemplo: recuperação de um *stream de bits*.

Para que uma lista de arquivos contidas em um *stream de bits* 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 9. Observe que para recuperar os *bits* que realmente correspondem aos *bits* da mensagem, o conjunto total de LSBs é analisado para que as permutações e todo o processo descrito na Seção 5.3.2 sejam desfeitos. Para que ocorra o processo de recuperação,

Stream recebido	Chave (distâncias)	Mensagem recuperada
00110110	0	0
00100110	0	1
10100000	2	1
10101001	0	0
00000010	1	0
10111011	.	.
00011100	.	.
01111110	.	.
01000101	.	.
11100011	.	.
10000001	.	.

Figura 8: O processo de recuperação segundo uma chave de deslocamento

deve-se interpretar o conjunto de *bits* recuperados e separar os campos. Defina-se:

- $MR =$ Mensagem recuperada já com as permutações e outras medidas de robustez desfeitas
- $C = \text{STEGBIN\#}$ (Cabeçalho)

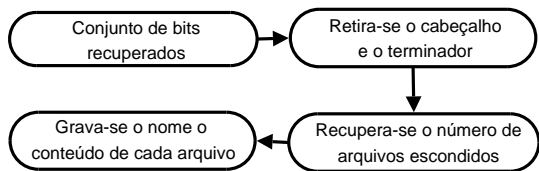


Figura 9: Recuperação de arquivos

- 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:

$$MR = C + NA + S + N_1 + S + TA_1 + S + N_2 + S + TA_2 + S + B_1 + B_2 + T, \quad (2)$$

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*.

5.3 Robustez do software

O *Camaleão* é uma solução robusta o suficiente para ser utilizada pelo público em geral. A robustez se deve, em parte, a três fatores: as chaves de deslocamento, as permutações cíclicas entre os blocos e a cifragem interna dos blocos que podem ocorrer na mensagem antes do mascaramento. Estas operações tornam a mensagem ininteligível de modo que, apenas com a chave correta consegue-se reverter os efeitos das permutações efetuadas.

5.3.1 As chaves

A chave de deslocamento é uma seqüência numérica de tamanho n . Os elementos pertencentes à chave pertencem ao intervalo $\{0, \dots, m\}$ sendo m 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. Atualmente o gerador utilizado é SHA1 (SCHNEIER, 1995).

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 deve ser repetida ao 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 maior ou igual ao tamanho da mensagem.

5.3.2 Cifragem de blocos e permutações

Visando tornar o sistema mais robusto a tentativas de detecção das mensagens escondidas, os autores desenvolveram uma abordagem chamada *cifragem por blocos com permutações cíclicas*. Com isso, consegue-se simular o mascaramento aleatório dos *bits* em uma imagem de cobertura. O processo é apresentado na Figura 10. A mensagem a ser escondida é dividida em N blocos de tamanho fixo. A partir desse momento, os N blocos são independentemente criptografados com o algoritmo 3DES. O tamanho de cada chave é de 56 *bits* totalizando uma chave de 168 *bits* ao final. As três chaves simétricas utilizadas são calculadas a partir da chave de deslocamento gerada de modo que o receptor da mensagem utilizando a mesma chave de deslocamento também consiga produzir as três chaves do algoritmo 3DES quando do recebimento da estego-imagem. O procedimento de cifragem utilizado foi o CDC *cifra-decifra-cifra* e o modo de operação escolhido foi o ECB *electronic-codebook*.

Este procedimento produz N blocos cifrados C . Os blocos C são então permutados entre si ciclicamente

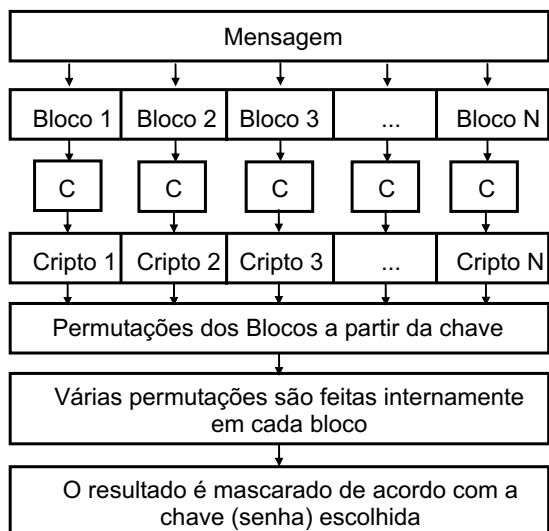


Figura 10: Fluxo de mascaramento

de acordo com a chave de deslocamento. Em seguida, mais um processo de *cifragem* é feito internamente em cada bloco. Desta vez, o processo é feito através de permutações cíclicas. O conjunto de *bits* resultante de todas estas operações é então escondido na imagem de cobertura a partir da chave de deslocamento gerada. Todo o processo é feito de modo a manter a *estego-imagem* resultante estatisticamente o mais próximo possível da imagem original.

Uma imagem típica possui 50% de seus LSBs iguais a 1 (WAYNER, 2002). Desta forma, um método robusto deve manter esta proporção durante o mascaramento. Para manter o número de 1's e 0's nos LSBs na mesma proporção, os LSBs não utilizados na cifragem dos blocos (LSBs inúteis) são utilizados como fator de balanceamento. Desta forma, ao mascarar uma mensagem de 2000 *bits* com a proporção de sete 1's para um 0, os LSBs não utilizados devem ser mudados até que esta proporção volte a ser próxima de um para um.

A seqüência de permutações a serem realizadas é dada pela chave de deslocamento. Seja SP a seqüência de permutações dadas por um conjunto de posições aleatoriamente gerado a partir de uma semente. Seja S a semente:

$$S = \sum_{i=1}^{ck} K_i + i, \quad (3)$$

sendo K o conjunto representando a chave de deslocamento fornecida e ck é a cardinalidade de K .

Exemplo: considere $K = [2 \ 1 \ 0 \ 2]$ uma chave de deslocamento. Esta chave é de tamanho $ck = 4$ 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:

$$S = (2 + 0) + (1 + 1) + (0 + 2) + (2 + 3). \quad (4)$$

Com isso, a semente gerada é $S = 11$. SP é 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.

É importante observar que o tamanho em *bits* da chave de deslocamento é diretamente proporcional à segurança do sistema. A utilização de chaves com mais de 1024 *bits* torna praticamente impossível a atuação de um indivíduo que queira recuperar a mensagem indevidamente (WAYNER, 2002).

5.3.3 Vantagens do método desenvolvido

Para validar o método desenvolvido, inúmeros testes foram feitos através da submissão das *estego-imagens* produzidas a dois dos mais tradicionais testes de *esteganálise* disponíveis atualmente: *Chi-Square Test* (WAYNER, 2002) e o *RS-Steganalysis* (FRIDRICH et al., 2001).

O *Chi-Square Test* é responsável por contar o número 1's e 0's presentes nos LSBs da imagem. Utilizando a abordagem tradicional, sem a utilização das permutações cíclicas e o balanceamento, o *Chi-Square Test* acusou a presença de 46% das mensagens escondidas. Este fator caiu para 24% após a melhoria implementada.

O *RS-Steganalysis* é um teste mais robusto que o *Chi-Square Test*. O método é responsável por dividir uma imagem em quatro conjuntos R_M , R_{-M} , S_M , S_{-M} . Através da utilização de uma máscara M ele consegue estimar, através dos padrões estatísticos presentes na imagem, a presença ou a ausência de uma mensagem escondida. O método atua em uma dada vizinhança classificando-a dentro dos quatro conjuntos criados. Em imagens típicas, o conjunto R_M é próximo ao conjunto R_{-M} e S_M é próximo de S_{-M} . Para imagens que tenham mensagens escondidas, quanto maior a mensagem escondida na imagem, maior a discrepância entre os conjuntos.

Utilizando a abordagem tradicional, sem a utilização das permutações cíclicas e o balanceamento, o *RS-Steganalysis* detectou a presença de 71% das mensagens escondidas nas imagens testadas. Este fator caiu para 49% após a melhoria implementada.

6 TRABALHOS FUTUROS

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

6.1 Códigos corretores de erros

Um dos grandes problemas da *esteganografia* em imagens consiste em recuperar a mensagem escondida após um ataque geométrico¹⁵ 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 a partir dos códigos corretores, estimar a mensagem escondida quando do recebimento da estego-imagem.

6.2 Padrões estatísticos

A análise estatística é um dos pilares da *esteganálise*. Uma das maneiras de aumentar a robustez do *Camaleão* é implementar uma função de imitação (*mimicry function*). Tal função seria responsável por analisar a imagem de cobertura verificando seus padrões estatísticos juntamente com a distribuição espacial dos dados e do número final de 1's e 0's, e mascarando 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.

7 CONCLUSÕES

Este artigo apresentou as principais técnicas de mascaramento, em especial, mascaramento em imagens. Também foi mostrado o método de mascaramento de dados em imagens baseado na utilização

de cifragem de blocos e permutações cíclicas implementado no *software Camaleão*. Foi mostrado também como o método contribuiu para aumentar a robustez do *software* gerado.

A *esteganografia*, quando bem utilizada, fornece meios eficientes e eficazes na busca por proteção digital. Associando *criptografia* e *esteganografia* as pessoas têm a possibilidade de comunicar-se pela rede mundial de computadores mantendo sua identidade íntegra e secreta exercendo, de forma efetiva, seus direitos à privacidade.

REFERÊNCIAS

FRIDRICH, J.; GOLJAN, M.; DU, R. Detecting lsb steganography in color and grayscale images. In: *IEEE Proceeding on Multimedia and Security*. [S.l.]: IEEE Multimedia, 2001. v. 8, p. 22–28.

FSF. *FSF – Free Software Foundation*. 2003. Disponível em www.fsf.org.

JOHNSON, N.; JAJODIA, S. Exploring steganography: seeing the unseen. In: *IEEE Internet Computing*. [S.l.: s.n.], 1998. v. 31, p. 26–34.

MICROSYSTEMS, S. *The Java Documentation*. 2003. Disponível em java.sun.com.

PETITCOLAS, F. A.; ANDERSON, R. J.; KUHN, M. G. Information hiding - a survey. In: *Proceedings of IEEE*. [S.l.: s.n.], 1999. v. 87, p. 1062–1078. Special issue on Protection on multimedia content.

POPA, R. *An analysis of steganography techniques*. Dissertação (Mestrado) — Department of Computer Science and Software Engineering of The “Polytechnic” University of Timisoara, Timisoara, Romênia, 1998.

PROVOS, N.; HONEYMAN, P. Hide and seek: An introduction to steganography. In: *IEEE Security & Privacy Magazine*. [S.l.: s.n.], 2003. v. 1, p. 32–44.

ROCHA, A. R. *Camaleão: um software para segurança digital utilizando esteganografia*. In: *Monografia de final de curso*. [S.l.]: Departamento de Ciência da Computação, Universidade Federal de Lavras, 2003. Disponível em www.comp.uf1a.br/curso/ano2003/.

SCHNEIER, B. *Applied Cryptography*. New York: John Wiley & Sons, 1995. ISBN 0-47111-709-9.

WAYNER, P. *Disappearing cryptography*. San Francisco: Morgan Kaufmann Publishers, 2002. ISBN 1-55860-769-2.

¹⁵Ataques geométricos referem-se a operações de compressão, recorte, realce, dilatação, erosão entre outras operações com o intuito de destruir alguma possível mensagem escondida.