Um Estudo de Técnicas de Esteganálise em Estego-Imagens com Texto Embutido com LSB

A Study of Steganalysis Techniques in Stego-Images with Embedded Text with LSB

RESUMO

Este trabalho faz um estudo comparativo de eficiência da ferramenta StegExpose, que reúne os métodos de esteganálise LSB: Chi Square, RS Analysis, Primary Sets, e Sample Pair, além de um método de fusão próprio. Para cada uma das 30 imagens da base de dados, embutiu-se texto usando cinco ferramentas de esteganografia, Stepic, LSBSteg, Steghide, Outguess e F5, e variando o tamanhos da mensagem secreta, em incrementos de 10%. Ao fim dos experimentos constatou-se que o desempenho de um método de esteganálise depende de qual foi a técnica de esteganografia, assim as ferramentas de esteganálise detectaram dados escondidos nas estego-imagens geradas pelo Stepic e LSBSteg, porém não detectaram nas estego-imagens geradas pelo Steghide, Outguess e F5, que usam métodos diferentes do LSB.

Palavras-chaves: Esteganografia. LSB. Esteganálise.

ABSTRACT

This work presents a comparative study of the efficiency of the StegExpose tool. StegExpose implements the LSB steganalysis methods: Chi Square, RS Analysis, Primary Sets, and Sample Pair, as well as a proprietary fusion method. For each of the 30 images in the database, text was embedded using five steganography tools: Stepic, LSBSteg, Steghide, Outguess and F5. Also, the sizes of the secret message was iterated in increments of 10%. The experiments show that the performance of a steganalysis method depends on the steganography technique used. Therefore, the steganalysis tools detected data hidden in the stego-images generated by Stepic and LSBSteg, but did not detect stego-images generated by Steghide, Outguess and F5, which use different methods of LSB.

Keywords: Steganography. LSB. Steganalysis.

1. INTRODUÇÃO

"The Prisioner's Problem" é um problema clássico descrito pelo criptógrafo Gustavus J. Simmons (SIMMONS, 1984). Neste cenário, dois cúmplices de um crime foram presos e estão trancados em celas separadas e distantes. O diretor do presídio tem motivos para suspeitar que os prisioneiros querem coordenar um plano de fuga, assim permitirá que existam trocas de mensagens desde que o diretor possa ler as mensagens e as considerar inócuas.

O diretor do presídio permitirá a troca de mensagens na esperança de que possa enganar um dos detentos a aceitar uma comunicação fraudulenta criada pelo próprio diretor. Os prisioneiros, por outro lado, estão dispostos a aceitar essas condições, ou seja, a aceitar algum risco de engano, a fim de serem capazes de se comunicar, uma vez que eles precisam coordenar seus planos. Para isso, terão de enganar o diretor, encontrando uma maneira de se comunicar secretamente nas trocas, isto

é, de estabelecer um “canal subliminar” entre elas, às vistas do diretor, mesmo que as próprias mensagens não contenham segredo.

Uma técnica possível para atingir o objetivo desta troca de informação com dados escondidos (feita pelos presos cúmplices) é a esteganografia (JULIO; BRAZIL; ALBUQUERQUE, 2007), e a técnica de análise de detectar mensagens escondidas nos meios de comunicação (feita pelo diretor) é a esteganálise (CHANDRAMOULI; KHARRAZI; MEMON, 2003).

Esteganografia, em um contexto atual, é a prática de esconder um arquivo digital dentro de outro. Um desses arquivos é o arquivo de cobertura (ou *cover-message* ou *carrier-medium*), enquanto o outro arquivo é um segredo que se deseja esconder dentro do arquivo de cobertura, chamado de dado embutido ou *embedded data*. Após a inserção do dado embutido na mensagem de cobertura se obtém o estego-objeto ou esteganograma ou *stego-object* (SUJATHA, 2013). Os arquivos, tanto de cobertura, quanto o segredo, podem ser imagens, vídeos, áudio, ou simplesmente texto. A palavra “objeto” no termo estego-objeto pode ser alterada para o tipo de arquivo, podendo ser estego-imagem ou estego-áudio, por exemplo.

Figura 1 – Arquitetura genérica de um sistema de esteganografia.

Fonte: elaborado pelo autor (2017).

A Figura 1 apresenta a arquitetura genérica de um sistema de esteganografia, no qual uma mensagem é embutida numa imagem, gerando um esteganograma ou no caso, uma estego-imagem. Depois a estego-imagem deve passar por um processo de extração para que a mensagem secreta seja retirada. Um esteganograma deve ter as mesmas características estatísticas do aquivo de cobertura original. De forma contrária, o sistema esteganográfico seria inseguro, pois a presença de uma mensagem secreta seria facilmente detectada (WESTFELD; PFITZMANN, 1999).

Há várias aplicações de esteganografia, pois informações podem ser escondidas de modo a funcionarem como estruturas de dados avançadas, tais como o armazenamento de informações médicas a respeito de um paciente em seu próprio raio X; fotos de satélite poderiam armazenar dados geográficos sobre os locais observados e inclusão de uma assinatura de autenticidade em documentos digitais.

Infelizmente, técnicas esteganográficas podem ser usadas ilegitimamente para transmitir imagens de pornografia ou pedofilia. Em 2002, a ação policial “Operations Twins” culminou na captura de criminosos associados ao grupo “Shadowz Brotherhood”, uma organização de pedofilia que distribuía pornografia infantil usando esteganografia em imagens que pareciam inocentes (MAZURCZYK et al., 2016), (WENDZEL et al., 2014). Em 2010, veio a tona a revelação que espiões russos nos Estados Unidos trocavam comunicações com Moscou codificando instruções em imagens de aparência inocente em sites públicos. A acusada rede de espionagem russa começou a usar esteganografia em 2005, de acordo com a queixa criminal do Departamento de Justiça contra os conspiradores (SHACHTMAN, 2010).

Com todas estas possibilidades de usos ilícitos, agências de inteligência de governos buscam desvendar e rastrear mensagens secretas (NSTC, 2006). É uma tarefa árdua, pois a única entrada da ferramenta de esteganálise é um arquivo e deve-se responder se há ou não dado escondido. A ferramenta não dispõe da imagem original para traçar qualquer tipo de cômputo de diferenças. Recuperar os dados escondidos, em geral, está além da capacidade da maioria das técnicas, segundo Julio, Brazil e Albuquerque (2007), dado que não se conhece como os bits podem estar espalhados pelo objeto de cobertura. Desta forma, os melhores algoritmos de esteganálise, em geral, não são capazes de dizer qual é a informação, mas devem responder se há ou não dados escondidos.

O propósito deste trabalho é analisar a eficiência de diferentes técnicas de esteganálise a fim de sondar imagens que podem ou não conter informações escondidas. Os questionamentos que se pretende responder com o estudo deste trabalho são: qual é o acerto das ferramentas de esteganálise em relação à estego-imagens com texto embutido? Este acerto varia com o tamanho do texto embutido?

Para responder à tais questionamentos, serão usadas ferramentas existentes de esteganografia e esteganálise. Há centenas de ferramentas de esteganografia disponíveis, apenas na listagem da página do Dr. Neil F. Johnson (JOHNSON, 2012), há 110 (cento e dez) ferramentas. Foram escolhidas três ferramentas, Outguess (PROVOS, 2001), o Steghide (HETZL, 2002) e o F5 (HETZL, 2002), pois segundo o Miche (2010) são algumas das ferramentas mais conceituadas para a esteganografia. Além destes, foram escolhidas mais duas ferramentas de esteganografia: o Stepic (DOMNITSER, 2007) e o LSBSteg (DAVID, 2015). Foi selecionada uma ferramenta de esteganálise, a StegExpose (BOEHM, 2014), que reune implementações de alguns dos principais métodos de esteganálise: Chi Square (WESTFELD; PFITZMANN, 1999), RS Analysis (FRIDRICH; GOLJAN; DU, 2001a), Primary Sets (DUMITRESCU; WU; MEMON, 2002) e Sample Pair (DUMITRESCU; WU; WANG, 2003).

2. FUNDAMENTAÇÃO TEÓRICA

Inicialmente, é necessário se diferenciar o conceito de criptografia e da esteganografia (CHEDDAD et al., 2010). A esteganografia contrasta com a criptografia quanto ao aspecto da confidencialidade. Na criptografia, terceiros podem ter acesso às mensagens e tem o conhecimento de que há uma informação, mas não conseguem saber qual é a informação confidencial que está sendo trocada ali, sem conhecer o método de descriptografia (por exemplo, através de uma chave). Já na esteganografia, o terceiro não sabe nem se a mensagem contém alguma informação secreta, pois um dos objetivos da esteganografia é modificar o arquivo portador de forma imperceptível, de maneira que nada seja revelado.

As duas técnicas podem ser usadas em conjunto, métodos de esteganografia frequentemente aplicam criptografia na mensagem secreta antes de embutí-la no arquivo de cobertura (PETITCOLAS; ANDERSON; KUHN, 1999).

2.1 Técnicas de Esteganografia

Técnicas de esteganografia que usam imagens como arquivo de cobertura podem atuar de duas formas possíveis na imagem: no domínio espacial ou no domínio da frequência. No domínio espacial, a técnica de esteganografia mais comum é a **Least Significant Bit** (LSB) (CHOUDHARY, 2012). E a manipulação no domínio da frequência é comum em arquivos JPEG, para o qual alguns algoritmos manipulam os coeficientes DCT (*Discrete Cosine Transform*) para esconder a mensagem secreta (CHEDDAD et al., 2010).

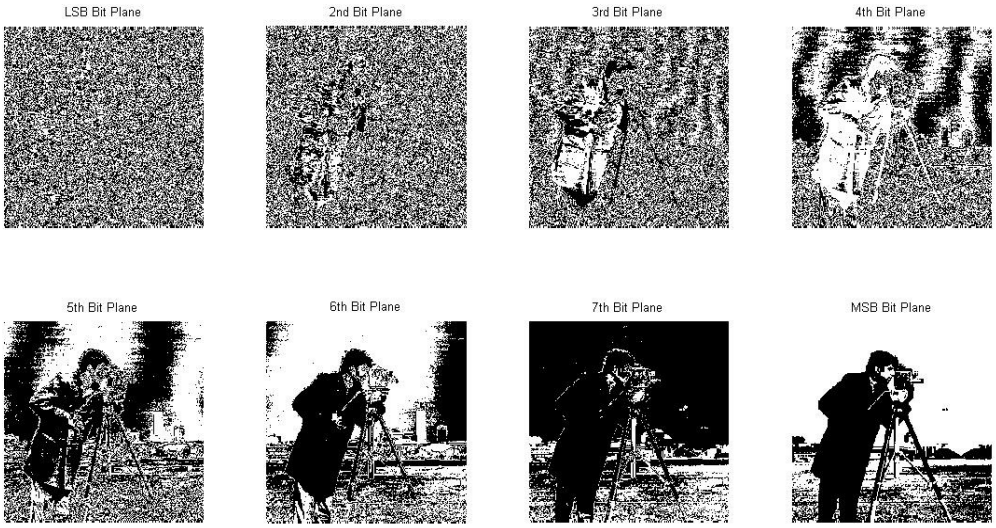
2.1.1 Esteganografia no domínio espacial

O LSB é o método mais básico de esteganografia moderna. O bit menos significativo é o que torna uma sequência de bits par ou impar. No byte 01101101, o bit menos significativo é o bit 1 que está na extrema direita. Ao alterar o LSB, o valor do byte muda para mais um ou menos um, o que não altera muito, se comparado a mudança de qualquer um dos outros bits. E é por isso que muitos métodos de esteganografia alteram apenas os LSBs, assim o valor de cada byte não terá uma alteração significativa. Alterando os LSBs, a capacidade máxima para esconder uma informação é de 1/8 do tamanho do arquivo. Se uma imagem tem 152 Kilobytes, pode-se esconder 19 Kilobytes (152 Kilobits) de informação.

Figura 2 – Imagem de entrada para demonstração do plano de bits.

Fonte: retirado de https://www.quora.com/What-are-the-most-common-example-images-used-in-image-processing-and-computer-vision

Figura 3 – Plano de bits da imagem da Figura 2.



Fonte: elaborado pelo autor (2019).

A Figura 3 mostra os oito planos de bits de uma imagem de entrada em escala de cinza – o **cameraman** da Figura 2. A Figura 3 inicia, na primeira linha e primeira coluna com a camada LSB e vai apresentando cada plano de bits até chegar ao MSB (*Most Significant Bit* ou o bit mais significativo). Em cada plano, as células escuro e claro representam os valores binários 0's e 1's, respectivamente. Visualmente, é possível verificar que o MSB contém informações que mais se aproximam na imagem de entrada, e que o LSB parece com ruído. Mudanças no plano LSB, em geral, é imperceptível para o olho humano.

Pelo algoritmo do LSB, ele é aplicado sobre uma imagem plana, sem compressão. Fridrich, Goljan e Du (2001b) mostraram que imagens de cobertura no formato JPEG não são uma boa

escolha para métodos esteganográficos que funcionam no domínio espacial. Isso porque a quantização introduzida pela compressão JPEG serve como uma “marca d’água semi-frágil” ou uma impressão digital que pode ser usada para detectar pequenas modificações no arquivo de cobertura ao inspecionar a compatibilidade da estego-imagem com o formato JPEG. Mudanças pequenas de modificação do LSB podem ser detectadas, consequentemente, usar imagens JPEG descomprimidas como imagens de cobertura deve ser evitado para métodos de esteganografia no domínio espacial, como LSB e suas variantes.

Há várias propostas de melhoria sobre a proposta LSB. A técnica LSB clássica faz a mudança dos bits da imagem de cobertura na ordem por linha e por coluna. De acordo com Sujatha (2013), há propostas para que essa cobertura naõ seja tão simples: Matrix Embedding, Pixel-value-based image hiding, Difference Expansion (DE), Histogram modification e Predicted based image hiding. Outra possibilidade é usar mais planos de bits além do lsb, tais como propostas de rajan et al. (2013) e Choudhary (2012).

Neste trabalho, usaremos três programas de esteganografia LSB: Stepic, LSBSteg e o Steghide.

2.1.2 Esteganografia no domínio da frequência

Das possíveis transformações de frequência, a mais relevante para a esteganografia em imagens é o DCT, pois é usado no formato de imagem JPEG. No formato de imagem JPEG, para cada componente de cor, o formato usa uma DCT para transformar blocos de 8 × 8 píxeis em 64 coeficientes DCTs (PROVOS; HONEYMAN, 2003). O formato JPEG atraiu bastante atenção para o ramo da esteganografia por duas razões: ele é o formato mais comum para armazenar imagens e usado amplamente na internet, e o formato é usado quase que exclusivamente para armazenar imagens naturais (FRIDRICH; GOLJAN; HOGEA, 2002b). Métodos de esteganografia modernos usam o formato JPEG por oferecer uma boa capacidade esteganográfica sem necessariamente sacrificar a segurança. O algoritmo F5, proposto por Westfeld (2001) como um desafio a comunidade científica, é um bom exemplo disso.

Neste trabalho, serão usados dois programas de esteganografia que implementam algoritmos para embutimento de informação nos coeficientes DCTs de imagens JPEG: o F5 e o Outguess.

2.2 Técnicas de Esteganálise

A esteganálise é uma área de pesquisa que busca criar métodos para detectar quando a esteganografia foi aplicada em um arquivo ou não. Existem dois tipos principais de esteganálise:

visuais e estatísticos (FRIDRICH; GOLJAN, 2002). Quando disponíveis, ataques estatísticos são superiores aos visuais, pois são menos dependentes do arquivo original e podem ser completamente automatizados, o que permite processar vários itens em larga escala (WESTFELD; PFITZMANN, 1999).

De acordo com Boehm (2014), os principais métodos de esteganálise estatísticos são: Chi Square, RS Analysis, Primary Sets e Sample Pair. Todos são métodos de análise contra o método LSB. Neste trabalho, será usada uma ferramenta que reúne implementações destes quatro métodos, chamada StegExpose.

2.2.1 Chi Square

O Chi Square (WESTFELD; PFITZMANN, 1999) é uma análise estatística dos pares de valores, PoVs (*Pair of Values*), trocados durante a esteganografia LSB. Sobrescrever os LSB's de uma imagem transforma alguns valores em outros valores que apenas diferem pelo LSB. Esses pares de valores são chamados de PoV. Por exemplo, 0110111**0** forma um PoV com 0110110**1**. Se os LSB's sobrescritos forem uniformemente distribuídos, as frequências dos valores de cada PoV irá se igualar. Ou seja, 0110111**0** apareceria tantas vezes quanto 0110110**1** na imagem.

A ideia do Chi Square é comparar a distribuição de frequências teórica esperada de esteganogramas com algumas amostras da distribuição observada no item que está sendo analisado.

Um ponto crítico é como se pode obter a distribuição de frequências teoricamente esperada (i.e. a frequência de ocorrências esperada após aplicarmos a esteganografia). Essa frequência não pode ser derivada do item que está sendo analisado, porque o mesmo pode ter sido alterado por operações esteganográficas. Mas na maioria dos casos, não se tem o arquivo original disponível para comparação, ou para derivar a frequência esperada.

Na imagem original, a frequência teórica esperada é a média aritmética das duas frequências de um PoV. Porque a esteganografia sobrescreve os LSBs, não há mudança na soma dessas duas frequências. A quantidade tirada dos valores ímpares é transferida para os valores pares correspondentes de cada PoV, e vice versa. Como a soma das frequências de um PoV é constante, a média aritmética também é a mesma para cada PoV, tanto no arquivo original, quanto no arquivo alterado (o esteganograma). Isso nos permite obter a distribuição de frequências teoricamente esperada através do item que está sendo analisado (WESTFELD; PFITZMANN, 1999). Assim, o arquivo original não é necessário.

O grau de similaridade entre a distribuição observada na amostra e a distribuição de frequências teoricamente esperada é a medida de probabilidade de que métodos esteganográficos

foram aplicados. Esse grau de similaridade é determinado usando o teste Chi Square. Esse teste faz um mapeamento das observações em categorias.

2.2.2 RS Steganalysis

Fridrich, Goljan e Du (2001a) criaram um método que detecta esteganografia LSB, quando os bits da mensagem secreta são aleatoriamente espalhados na imagem de cobertura (que pode ser em tons de cinza ou coloridas). Os autores do RS Steganalysis exemplifica quatro programas vulneráveis a técnica: Steganos, Windstorm, S-Tools, e Hide4PGP. Apesar disso, hoje já existem métodos de esteganografia robustos ao RS Steganalysis, como os desenvolvidos por Nair et al. (2012) e Marçal e Pereira (2005). Os autores também definem que um limite máximo de 0,005 bits por píxel pode ser definido como seguro para esteganografia LSB.

Um conceito importante para o RS Analysis é o de embutimento sem perdas (*lossless embedding*), ou embutimento reversível (*invertible embedding*), e é tópico de um trabalho anterior dos mesmos autores Goljan, Fridrich e Du (2001).

Para a maioria das imagens, o plano LSB é essencialmente aleatório. Porém, ainda assim o plano LSB está relacionado aos outros planos de bits. Suponha uma imagem em tons de cinza de tamanho M × N píxeis, na qual cada píxel pode assumir um valor entre 0 e 255 pertencente a um conjunto **P** = {0, . . . , 255}. O processo de embutimento sem perdas começa com a criação de grupos disjuntos de **n** píxeis adjacentes $(x_1, ..., x_n)$. Para facilitar, vamos supor que n=4, então representada por um grupo **G**, como $G = (x_1, x_2, x_3, x_4)$. Deve-se definir uma função de discriminação **f** que designa um número real para cada um dos grupos pela Equação (1).

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

Uma forma simples de compreender a Equação (1) é entender que ela simplesmente soma a diferença entre os valores consecutivos de um grupo. O propósito da função de discriminação é capturar a regularidade do grupo de píxeis **G**. Quanto maior o resultado de **f**, mais irregular é o grupo. Por exemplo (1, 2, 3, 4) é um grupo um pouco irregular **f**(1, 2, 3, 4) = |1| + |1| + |1| = 3, enquanto (11, 22, 55, 33) é mais irregular **f**(11, 22, 55, 33) = 66, e (233, 233, 233, 233) é tão regular quanto possível **f**(233, 233, 233, 233) = 0.

Finalmente, define-se uma operação **F** em **P** chamada de inversão (*flipping*). O *flipping* será uma permutação dos níveis de cinza. **F** tem uma propriedade especial em que **F**(**F**(x)) = x, ou seja, para todo x ∈ P , se **F** for aplicada duas vezes seguidas a um valor, o resultado será o valor original.

Em outras palavras, quando aplicada duas vezes, a segunda operação desfaz o efeito da primeira. A permutação $\mathbf{F}_1 : 0 \leftrightarrow 1,\ 2 \leftrightarrow 3,\ ...,\ 254 \leftrightarrow 255$ corresponde a inversão do LSB na escala de cinza. Também define-se F_{-1} como $-1 \leftrightarrow 0,\ 1 \leftrightarrow 2,\ ...,\ 255 \leftrightarrow 256$, ou

$$F_{-1}(x) = F_1(x+1) - 1 \text{ para todo } x \tag{2}$$

Para maior completude, define-se $\mathbf{F}_0(x) = x$ para todo $x \in P$. A partir de agora, quando **F** for usada sem especificar qual das variações, significa que se aplica a qualquer uma das três ($\mathbf{F}_{-1}$, $\mathbf{F}_0$, ou $\mathbf{F}_1$). Usa-se a função de discriminação **f** e a operação de *flipping* **F** para definir três tipos de grupos de píxeis:

- Grupos regulares: $G \in R \leftrightarrow f(F(G)) > f(G)$
- Grupos singulares: $G \in S \leftrightarrow f(F(G)) < f(G)$
- Grupos inutilizáveis: $G \in U \leftrightarrow f(F(G)) = f(G)$

Nas expressões acima, F(G) significa a aplicação da operação de *flipping* em cada um dos componentes do grupo $(F(x_1), \ldots, F(x_n))$. Para entender esses grupos, pode-se pensar da seguinte forma: os grupos regulares possuem um valor baixo quando passam pela função discriminante **f** (ou seja, são bem regulares). Porém, após aplicação da operação de *flipping* **F** no grupo, ele se torna mais irregular do que era anteriormente. Por exemplo, o grupo (1234) após passar por $\mathbf{F}_1$ se torna (0325), aplicando a função discriminante tem-se **f**(1234) = 3 e **f**(0325) = 7, por isso considera-se o grupo como regular. O contrário acontece para os grupos singulares, eles ficam mais regulares ao passarem pela operação de *flipping*. E os grupos inutilizáveis continuam tão regular quanto eram antes.

É possível que aplique-se diferentes operações de *flipping* em diferentes componentes de um grupo G. Para isso, usa-se uma máscara M que é uma n-tupla com valores −1, 0, ou 1 (equivalentes a $\mathbf{F}_{-1}$, $\mathbf{F}_0$ e $\mathbf{F}_1$). Define-se o grupo invertido **F**(G) como $(F_{M(1)}(x_1), F_{M(2)}(x_2), ..., F_{M(n)}(x_n))$, isto é, a mascara será aplicada para cada um dos seus respectivos elementos.

O propósito de **F** é modificar os valores dos píxeis sutilmente de uma forma reversível, para simular o ato da adição de ruído reversível. Em uma imagem típica, adicionar pequenas quantidades de ruído (invertendo os bits) leva a um aumento no resultado da função de discriminação (ou seja, ela fica mais irregular). E por isso, o número total de grupos regulares será maior do que o número de grupos singulares. E isso permite um embutimento sem perdas e imperceptível de uma quantidade possivelmente grande de informação.

Para se compreender a técnica de esteganálise, há ainda mais alguns conceitos. Define-se R_M para denotar a porcentagem relativa de grupos regularespara a máscara M (em porcentagem com

relação à todos os grupos). E da mesma forma S_M para a porcentagem de grupos singulares para a mesma máscara. Sabemos que ($R_M + S_M$) ≤ 1 e também que ($R_{-M} + S_{-M}$) ≤ 1 para a máscara negativa. O método RS Steganalysis afirma que em uma imagem típica, o valor esperado para R_M é igual ao de R_{-M} e que o mesmo é verdade para S_M e S_{-M}:

$$R_M \cong R_{-M} \text{ e } S_M \cong S_{-M} \quad (3)$$

Os autores do método afirmam que é possível justificar essa hipótese de forma heurística através da Equação (2). Em uma imagem típica, não há razão para que o número de grupos regulares R e singulares S devam ser modificados significativamente ao alterar os valores das cores por uma unidade apenas.

A chave para entender o método é que a relação demonstrada na Equação (3) é violada após randomizar o plano LSB, em função da esteganografia, por exemplo. Isso porque a randomização do plano LSB faz com que a diferença entre R_M e S_M caia para zero a medida que o tamanho *m* da mensagem secreta aumenta. Após o *flipping* de 50% do plano LSB (o que acontece se for inserido um bit de mensagem secreta em 100% dos píxeis da imagem), o resultado é que $R_M \sim = S_M$. E o efeito oposto acontece com R_{-M} e S_{-M} , a diferença entre eles aumenta a medida que o tamanho da mensagem secreta aumenta.

Para implementar o RS Steganalysis precisa-se definir mais alguns parâmetros: o tamanho dos grupos em píxeis e qual máscara usar. O tamanho da máscara e o tamanho dos grupos devem ser iguais, mas mesmo com o tamanho definido, há várias possibilidades de máscaras. Por exemplo, para grupos de três píxeis, pode-se definir uma máscara [−1, 1, −1] (representando F_{-1} , F_1 e F_{-1}).

Marçal e Pereira (2005) apresentam um método de esteganografia que é resistente ao RS Steganalysis. O método é baseado na aplicação de funções de transformação do histograma reversíveis às imagens, antes e depois de embutir a mensagem secreta.

2.2.3 Primary Sets

Inspirados pelo RS Steganalysis, Dumitrescu, Wu e Memon (2002) introduzem um novo método de esteganálise que pode detectar esteganografia LSB espalhadas aleatoriamente (assim como o RS Steganalysis) em imagens de tom contínuo.

O Primary Sets se baseia em uma identidade estatística relacionada a alguns conjuntos de píxeis em uma imagem. Uma identidade que é bem sensível a esteganografia LSB e às mudanças feitas a essa identidade podem ser mensuradas (nos dando o tamanho da mensagem secreta). Similar ao RS Steganalysis, formam-se grupos de píxeis na imagem, com a diferença que os grupos do Primary Sets são sempre formados por apenas dois píxeis adjacentes horizontalmente, isto é, um

par de píxeis (u, v). E considera-se P como o conjunto de todos esses pares de píxeis e definem-se os subconjuntos X e Y da seguinte forma:

- X é o conjunto de pares (u, v) ∈ P de forma que v é par e u < v, ou v é ímpar e u > v.
- Y é o conjunto de pares (u, v) ∈ P de forma que v é par e u > v, ou v é ímpar e u < v.

Os autores justificam a importância de X e Y para a esteganálise LSB, pois estatisticamente em imagens naturais, o número de elementos dos dois conjuntos é igual. Essa suposição é dada pela Equação (4). |X| é a cardinalidade do conjunto X, ou seja, quantos elementos existem dentro de X.

$$|X| = |Y| \quad (4)$$

Em seguida, define-se também Z, que é o conjunto de pares (u, v) em que u = v. E também define-se dois subconjuntos de Y, são eles W e V. W é o conjunto de pares (u, v) na forma de (2k, 2k + 1) ou (2k + 1, 2k) como os autores definem. Ou seja em todos os pares de W , u e v diferem por apenas uma unidade. Já o subconjunto V contém os restantes de pares pertencentes a Y, ou seja $V = Y - W$. Os conjuntos X, W , V e Z são chamados de Primary Sets (Conjuntos Primários) pelos autores e a união desses conjuntos forma P.

$$P = X \cup W \cup V \cup Z \quad (5)$$

O embutimento de bits no plano LSB, causados pela esteganografia modifica alguns valores de píxeis. E dessa forma alguns pares de píxeis vão mudar de categoria, seja X, V , W ou Z. Um par de píxel (u, v) poderá ser alterado segundo uma das seguintes situações:

1. Ambos os valores u e v continuarão sem modificação;
2. Apenas u é modificado;
3. Apenas v é modificado;
4. Tanto u quanto v são modificados.

Pode-se descrever cada um dos padrões de modificação acima usando dois bits. O primeiro bit do padrão representa se u foi modificado (1), ou não (0). E o segundo bit representa se v foi modificado. Então para cada um dos itens acima (1, 2, 3 e 4), associa-se um padrão de modificação 00, 10, 01 ou 11 respectivamente. Quando os pares de píxeis são modificados, transitam de um conjunto primário para outro.

Para os padrões de modificação, denota-se o símbolo π, que pode assumir um dos seguintes valores: 00, 01, 10, ou 11. E um subconjunto A que representa um dos Primary Sets, A ∈ {X, V, W, Z}. Repare que os Primary Sets V e W estão juntos no diagrama de estados. Os autores usam ρ(π, A) (letra grega ρ) para denotar a probabilidade que os pares de píxeis em A serão modificados pelo padrão π. Os autores fazem a seguinte suposição a respeito desses valores:

$$\rho(\pi, A) = \rho(\pi, P) \quad (6)$$

Ou seja, a probabilidade de um par de píxeis de qualquer um dos Primary Sets ser modificado por qualquer um dos padrões é a mesma probabilidade de qualquer outro par de píxeis na imagem ser modificado. A Equação (6) significa dizer que os bits da mensagem secreta estão aleatoriamente espalhados pelo plano LSB da imagem. E esta suposição é importantíssima para que o método esteganalítico do Primary Sets funcione. Se em uma imagem de cobertura, os bits da mensagem secreta foram sequencialmente embutidos, então outros métodos de esteganálise podem ser mais apropriados, como o Chi Square.

A partir do que foi definido, os autores conseguem derivar as seguintes equações:

$$\rho(00, P) = (1 - m/2)\ 2 \quad (2.13) \qquad (7)$$

$$\rho(01, P) = \rho(10, P) = m/2(1 - m/2) \quad (2.14) \qquad (8)$$

$$\rho(11, P) = (m/2)\ 2 \quad (2.15) \qquad (9)$$

Nas equações, perceba a diferença entre m e ρ. ρ representa a probabilidade dos pares de píxeis serem alterados (conforme discutido acima), enquanto m representa o tamanho da mensagem secreta embutida na imagem em bits com relação ao número de píxeis da imagem. E a fração de píxeis realmente modificados em função da esteganografia LSB é metade desse valor, isto é: m/2.

Tendo $A \in \{X, Y, V, W, Z\}$ entende-se que A 0 é o conjunto definido pelos mesmos pares de píxeis que A, porém alterados pelo embutimento de bits. A Equação (4) nos permitem encontrar a cardinalidade (i.e. número de elementos em um conjunto) dos Primary Sets em função de *m* e as cardinalidades antes da esteganografia, através das seguintes equações (|A| é a cardinalidade do conjunto A):

$$|X'| = |X| \cdot (1 - m/2) + |V| \cdot m/2 \qquad (10)$$

$$|V'| = |V| \cdot (1 - m/2) + |X| \cdot m/2 \qquad (11)$$

$$|W'| = |W| \cdot (1 - m + m^2/2) + |Z| \cdot m(1 - m/2) \qquad (12)$$

Então, usando essas equações, e aquelas definidas anteriormente, e também $\gamma = |W| + |Z| = |W'| + |Z'|$, os autores chegam a uma equação simples para estimar o comprimento m da mensagem secreta, baseados nos valores de |X'|, |Y'| e $\gamma = |W' \cup Z'|$. O valor de *m* é a menor solução possível para a equação quadrática (13), dado que $\gamma \neq 0$.

$$0.5\gamma m^2 + (2|X'| - |P|)m + |Y'| - |X'| = 0 \qquad (13)$$

Repare que os valores X', Y', W', e Z' estão todos definidos na estego-imagem e pode-se obtê-los analisando a imagem. E então usar a Equação (13) para determinar o tamanho da mensagem secreta m. Se $\gamma = 0$, então a Equação (13) se torna uma função identidade. Uma função identidade é uma que sempre retorna o mesmo valor que recebeu como argumento. Consequentemente o método esteganalítico irá falhar. Quando $\gamma = 0$, então $|X'| = |X| = |Y| = |Y'| = |$

P|/2. Uma vez que γ = |W| + |Z| , γ é o número de pares de píxeis pertencentes a P que diferem apenas no LSB . Para imagens naturais a probabilidade de γ ser igual à 0 é muito pequena.

2.2.4 Sample Pair

Dumitrescu, Wu e Wang (2003) introduzem um novo método esteganalítico, chamado Sample Pair, para detectar esteganografia LSB em sinais digitais, como imagens e áudio. Os autores do método mostram que o tamanho da mensagem secreta pode ser medido com uma precisão relativamente alta. O método é baseado em medições estatísticas nos sample pairs mais sensíveis as operações de embutimento LSB.

O método é baseado em uma máquina de estados finitos cujos estados são multiconjuntos selecionados de pares de amostras chamados *trace multisets*. Alguns desses *trace multiset*s são iguais em suas cardinalidades esperadas, se os pares de amostras são tirados de um sinal contínuo digitalizado. O embutimento LSB aleatório causa transições entre esses *trace multisets*, e consequentemente alteram a relação estatística das cardinalidades dos *trace multisets*. Mesmo quando a mensagem embutida é muito pequena, a estatísticas dos sample pairs é extremamente sensível ao embutimento LSB. Ao analisar essas relações e modelá-las em uma máquina de estados finitos, os autores determinam uma função quadrática simples que pode estimar o comprimento da mensagem embutida com alta precisão. Para chegar à essa função quadrática é necessário uma certa suposição que geralmente é verdadeira para imagens e áudio natural. Adicionalmente, é possível estimar o grau em que essa suposição desvia da realidade.

2.3 Avaliação Quantitativa dos Resultados

Como descrito por Cancelli (2009), a estaganálise é um problema de classificação. Um modelo de classificação é um mapeamento de instancias para suas classes previstas (MENA, 2012). Por exemplo, em uma fábrica temos uma esteira na qual passam produtos continuamente. No final da esteira temos uma máquina que busca detectar produtos defeituosos e descartá-los. Assim, a máquina usa um modelo de classificação para classificar os produtos como “defeituosos” ou “não defeituosos”. “Defeituoso” e “não defeituoso” são as possíveis classes, e cada produto (instância) será mapeado para uma dessas classes.

Classificadores podem ser de dois tipos: contínuo, ou discreto. Classificadores do tipo contínuo produzem uma estimativa que diz a probabilidade de uma instância ter certa classificação. E a partir dessa estimativa, pode-se definir um *threshold* para dar a classificação final de cada instância. Enquanto um classificador discreto, apenas informa qual a classe prevista daquela instância.

Considerando apenas problemas de classificação que usam apenas duas classes (JAMES et al., 2013). Uma instância I é mapeada para a classe positivo (p) ou para a classe negativo (n). E um classificador tentará predizer a classe dessa instância. Quando ele prevê que uma instância é positiva, usa-se o label Y. E quando ele prevê que a instância é negativa, usa-se o label N . Então {p, n} se refere a classificação real da instância, e {Y, N}, a classificação prevista. Quando uma instância é classificada existem quatro possibilidades:

- Se a instância for positiva e é classificada como positiva, tem-se um verdadeiro positivo.
- Se a instância for positiva e é classificada como negativa, tem-se um falso negativo.
- Se a instância for negativa e é classificada como positiva, tem-se um falso positivo.
- Se a instância for negativa e é classificada como negativa, tem-se um verdadeiro negativo.

Um gráfico ROC (*Receiver Operating Characteristics*) é uma técnica de visualização, organização e seleção de classificadores baseado em sua performance (FAWCETT, 2006). Gráficos ROC são gráficos de duas dimensões, nos quais o eixo Y representa a taxa de verdadeiros positivos (tp rate) e o eixo X representa a taxa de falsos positivos. O gráfico retrata uma balança entre verdadeiro positivo e falso positivo.

Conforme definido anteriormente, um classificador discreto é aquele que apenas produz uma label que define a classe da instancia, no gráfico ROC, cada classificador discreto corresponde a um único ponto no gráfico. O ponto (0, 0) representa a estratégia de um classificador que nunca produzirá uma classificação positiva; e o ponto (1, 1), a estratégia que sempre produzirá classificações positivas. O ponto (0, 1) representa um classificador perfeito. Classificadores no triângulo superior e próximos do eixo X fazem classificações positivas apenas com fortes evidências. Assim eles cometem poucos falsos positivos, porém também têm baixa taxa de verdadeiros positivos. Classificadores localizados no canto superior direito fazem classificações positivas sem muitas evidências e com isso eles classificam todos os positivos corretamente, mas frequentemente classificam negativos incorretamente como positivos, o que resulta em uma alta taxa de falsos positivos.

Pontos dispostos na linha x = y representam estratégias de classificação aleatórias. E para se afastar dessa linha e se posicionar na área do triângulo superior esquerdo, um classificador precisa verdadeiramente analisar dados das instâncias para produzir uma classificação que mais se aproxime da realidade.

Qualquer classificador que apareça na área do triângulo inferior direito tem uma performance pior do que uma classificação aleatória. Porém tais classificadores podem ser negados (i.e. invertemos as suas decisões) e assim estarão posicionados no triângulo superior esquerdo. E por isso o triângulo inferior direito está frequentemente vazio em gráficos ROC.

Alguns classificadores produzem uma probabilidade ou *score*, que é um número que representa o grau em que uma instância se aproxima de uma classe. Então pode-se definir um *threshold* para o classificador, que é um limite que determina como as instâncias serão classificadas de acordo com o seu score individual. Uma instância que possui um *score* acima do *threshold* é classificada como positiva; e uma instância que possui um *score* abaixo do *threshold* é classificada como negativa.

Dessa forma, esse tipo de classificador pode ser usado para produzir um classificador discreto. Cada *threshold* diferente produz um classificador discreto diferente, que corresponde a um único ponto no gráfico. Se variarmos o *threshold* de um extremo ao outro, produzimos uma série de pontos no gráfico que compõe uma curva, chamado de curva ROC.

Ao comparar classificadores é comum o desejo de se usar um simples valor escalar para representar seus desempenhos. Isso é possível ao analisar a área embaixo da curva ROC (*Area Under the Curve*, ou AUC). Qualquer valor de AUC sempre estará entre 0 e 1, pois a área total do gráfico ROC é 1. E como a estratégia de classificação aleatória (r*andom guessing*) corresponde à diagonal de (0, 0) à (1, 1), que tem uma área de 50%, não se pode esperar que um bom classificador tenha um AUC menor que 0,5. Uma propriedade importante do AUC é que este valor é equivalente à probabilidade do classificador dar um score maior para uma instância positiva qualquer do que para um instância negativa qualquer.

3. MATERIAIS E MÉTODOS

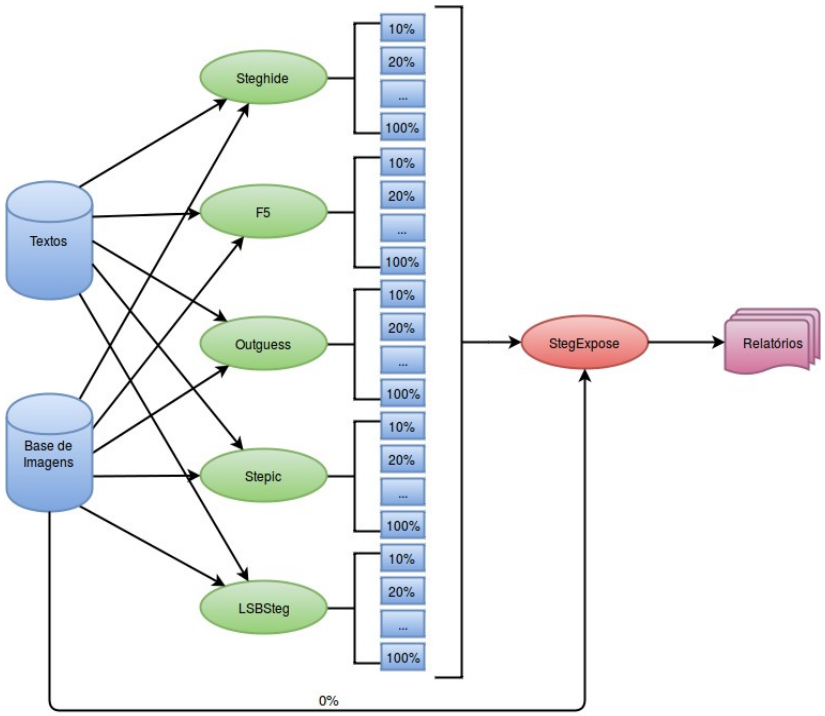
A partir de um banco de imagens de teste, deseja-se embutir mensagens secretas de tamanhos variados usando cada uma das ferramentas de esteganografia selecionadas. O fluxo da Figura 4 mostra que uma imagem passa por cinco sistemas de esteganografia diferentes, para cada sistema, embutiu-se mensagens de tamanhos diferentes, de 10 em 10%, equivalentes à 10%, 20%, até 100% da capacidade esteganográfica da imagem. Finalmente, após todos os embutimentos serem realizados, o StegExpose irá analisar cada uma das imagens (incluindo as imagens limpas) e avaliar se a imagem contém uma mensagem secreta ou não. O StegExpose irá gerar um relatório detalhado para análise.

Assim, para cada imagem na base de imagens, foram embutidas mensagens de 50 (cinquenta) formas diferentes: cinco diferentes ferramentas de esteganografia mencionadas (Steghide, Outguess, F5, Stepic e LSBSteg), e com dez diferentes tamanhos de mensagem secreta dependentes da capacidade esteganográfica da imagem (percentuais de 10% a 100%). Com isso, serão 1.500 imagens a serem testadas pelo StegExpose, mais as 30 (trinta) imagens sem mensagens escondidas.

O processo para verificar a capacidade esteganográfica de uma imagem é da seguinte forma: para os programas Outguess e F5 embutiu-se uma pequena mensagem de teste na imagem. Esses dois programas imprimem na tela uma série de informações úteis, dentre elas a capacidade esteganográfica da imagem. Para o Steghide, usou-se o comando **steghide info** que imprime a capacidade esteganográfica da imagem. Para os algoritmos LSBs (LSBSteg e Stepic), foi feito um cálculo simples da quantidade de bytes existentes na imagem. Mas é interessante notar que se o plano LSB for esgotado, o LSBSteg continuará embutindo nos outros planos de bit, se houverem mais bits para serem embutidos.

O Outguess tem uma capacidade esteganográfica que, além de variar com a imagem, também varia de acordo com o conteúdo da mensagem secreta. Por isso, durante os testes, era importante observar quando algum dos embutimentos estourasse a capacidade de uma imagem e falhasse. E de fato, alguns embutimentos de 100% da capacidade usando o Outguess falharam, como veremos nos resultados.

Figura 4 – Fluxo dos experimentos.



Fonte: elaborado pelo autor (2017).

Nas próximas seções serão descritos como a base de dados foi gerada, e sobre as cinco ferramentas de esteganografia e a ferramenta de esteganálise StegExpose.

3.1 Materiais

A base de dados foi construída a partir de imagens de na internet. Todas as imagens são de domínio público, e em sua grande maioria são fotografias. Como fonte de imagens, decidiu-se usar o Pixabay[1] como fonte exclusiva para as imagens, pois todas as imagens hospedadas no site são de domínio público, licenciadas sob a Creative Commons CC0[2]. Além disso, o Pixabay possui uma API[3] de fácil uso, o que eliminou a necessidade de construir um Web Crawler.

Para encontrar imagens no Pixabay usamos nomes de países como termos de pesquisa, assim como foi feito também por Boehm (2014). Os nomes dos países usados foram: Russia, Canada, China, United, Brazil, Australia, India, Argentina e Kazakhstan (sem os acentos mesmo). Ao todo, foram coletadas 30 imagens em JPEG para compor o banco de imagens. Houve padronização para que todas as imagens ficassem com resolução de 400 × 400.

A base de textos foi composto por vários livros do Project Gutenberg[4]. Todos os arquivos são em formato de texto puro (.txt), codificados em UTF-8. Todos os livros são obras literárias que estão em domínio público. Todos os livros foram compilados para um único arquivo (aproximadamente 8 MB). A partir deste arquico único, extraía-se a quantidade exata de bytes necessária para ser usada pelos programas de esteganografia.

Os títulos dos livros usados são: Alice’s Adventures in Wonderland; Pride and Prejudice; Emma; The Adventures of Sherlock Holmes; The Jungle Book; Grimms’ Fairy Tales; Moby Dick, or The Whale; Dracula; Metamorphosis; A Man from the North; When Sarah Saved the Day; Adventures of Huckleberry Finn; Frankenstein; The Importance of Being Earnest e A Tale of Two Cities.

3.2 Métodos

Nas próximas subseções serão descritas as ferramentas: Steghide, Outguess, F5, Stepic e LSBSteg, além da ferramenta de esteganálise: o StegExpose.

3.2.1 Stepic

O Stepic é um programa de esteganografia LSB desenvolvido na linguagem Python que se encontra nos repositório oficiais do Ubuntu (DOMNITSER, 2007). Em uma imagem RGB, o Stepic usa os três bytes que codificam as cores de um píxel para embutir dados. O *byte* que codifica a opacidade da imagem (*alpha*) é ignorado pelo Stepic, quando presente na imagem. O algoritmo atua de três em três píxeis. Ele seleciona um grupo de três píxeis, totalizando 9 *bytes*, e usa os LSBs dos

1 Site do Pixabay: <https://pixabay.com/>
2 Creative Commons CC0 : <https://creativecommons.org/publicdomain/zero/1.0/deed.pt_BR>
3 Documentação da API do Pixabay: <https://pixabay.com/api/docs/>
4 <http://www.gutenberg.org/>

8 primeiros *bytes* para embutir 8 bits da mensagem secreta. E então ele usa o LSB do nono *byte* restante destes três píxeis como uma *flag* para sinalizar que a mensagem secreta chegou ao fim.

3.2.2 LSBSteg

O LSBSteg é um ferramenta de esteganografia LSB (DAVID, 2015). Sua implementação é feita em Python usando as funcionalidades oferecidas pelo OpenCV (TEAM, 2017). O algoritmo do LSBSteg embute os bits da mensagem secreta nos LSBs dos componentes de cor dos píxeis da imagem de cobertura. Se for usado uma mensagem secreta grande demais e que ultrapasse a capacidade do plano LSB, o algoritmo usará os outros planos de bit para embutir os dados, conforme for necessário.

3.2.2 Steghide

O Steghide é um algoritmo de esteganografia que permite esconder dados em arquivos de imagem e também áudio usando um método LSB não linear (HETZL, 2002). O programa usa uma abordagem baseada em teoria de grafos para a escolha de quais píxeis serão alterados via LSB.

O algoritmo funciona da seguinte forma: primeiramente, os dados secretos são comprimidos e encriptados. Então uma sequência de posições de píxeis no arquivo de cobertura é criada baseada em um gerador de números pseudo-aleatórios inicializado com a senha (os dados secretos serão embutidos nos píxeis, nessas posições). Destas posições, aquelas que não precisam ser alteradas (porque já possuem o valor correto por acaso) são separadas. Então um algoritmo de seleção baseado em teoria de grafos encontra pares de posições de forma que trocando seus valores temos o efeito de embutir a parte correspondente dos dados secretos. Se o algoritmo não conseguir encontrar mais nenhum par, todas as trocas são realizadas. Os píxeis nas posições restantes (as posições que não são parte desses pares) são também modificadas para conter os dados secretos (mas isso é feito através de sobrescrita, não apenas trocando com outros píxeis).

O fato de que a maior parte do algoritmo de embutimento ser feito através da troca de valores de píxeis implica que estatísticas de primeira ordem (i.e. o número de vezes que uma cor ocorre na imagem) não são alteradas. Para arquivos de áudio o algoritmo é o mesmo, exceto que amostras de áudio são usadas no lugar de píxeis.

3.2.3 Outguess

O Outguess está disponível nos repositórios oficiais do Ubuntu. O algoritmo esteganográfico do Outguess foi proposto por Provos (2001) para ser resistente ao ataque do Chi Square. O método

altera o LSB dos coeficientes DCT. Altera no máximo a metade dos coeficientes e o restante dos coeficientes são ajustados para que o histograma permaneça inalterado.

O algoritmo funciona em duas etapas. Primeiramente, ele embute bits nos coeficientes DCTs da imagem JPEG saltando coeficientes 0 e 1, porém ele faz isso caminhando aleatoriamente pelos coeficientes. Na segunda etapa, o Outguess processa a imagem novamente fazendo correções nos coeficientes para que o histograma da estego-imagem seja equivalente ao histograma da imagem de cobertura. Como o Chi Square é um ataque estatístico de primeira ordem (ataques baseados no histograma) ele não consegue detectar o Outguess.

O Outguess foi desenvolvido em duas versões principais. A primeira versão usa um gerador de números pseudo aleatórios (*pseudo-random number generator*, ou PRNG) para selecionar coeficientes DCTs aleatórios e embutir bits da mensagem secreta em seus LSBs. Porém os autores notaram que o algoritmo fazia alterações no histograma dos coeficientes DCTs. E essas alterações eram detectáveis usando uma versão estendida do Chi Square. O teste Chi Square original (WESTFELD; PFITZMANN, 1999) não pode detectar o algoritmo do Outguess, pois o teste não detecta embutimento de bits espalhados pelo esteganograma. Porém os próprios autores do Outguess criam um teste Chi Square estendido que pode detectar sim o algoritmo. E em seguida, aperfeiçoam o algoritmo de embutimento para não ser detectado pela versão estendida do teste.

O teste original do Chi Square usa uma amostra que cresce continuamente em tamanho e sempre começa do início da imagem, e por isso apenas detecta mudanças que distorcem o histograma continuamente a partir do início da imagem. Este não é o caso do Outguess, que distribui suas mudanças espalhadamente pelo esteganograma.

O teste estendido difere em dois aspectos do teste original: (1) ao invés de aumentar gradualmente o tamanho da amostra, o novo teste usa um tamanho de amostra fixo; (2) ao invés de usar uma posição fixa (início da imagem), o novo teste desliza a posição da amostra por toda a imagem. Usando o teste estendido é possível detectar a versão inicial do Outguess.

A abordagem do algoritmo original tenta preservar a razão entre bits 1 e 0 da imagem original. Para isso, sempre que o algoritmo altera um bit de 0 para 1, ele tenta mudar um bit próximo de 1 para 0. Isso ajuda a prevenir o aumento da entropia nos dados redundantes. O aumento da entropia significa que a taxa de bits 1 e 0 tende à 50% para cada. Porém essa estratégia não previne uma distorção no histograma das DCTs.

Para evitar uma distorção no histograma, os autores propõe uma nova versão do Outguess que faz uma compensação nos coeficientes DCT para que o histograma da imagem modificada seja idêntico ao da imagem original. Por exemplo: toda vez que um coeficiente DCT de valor 4 é

alterado para 5, o algoritmo altera um coeficiente de valor 5 para o valor 4. Ele faz isso para todas alterações que ele causa aos coeficientes. E assim preservar as características do histograma.

O Outguess permite embutir mais de uma mensagem em um mesmo esteganograma. Ele tenta embutir novas mensagens sem sobrescrever as anteriores. Isso dá suporte a algo chamado "negação plausível" (*Plausible Deniability*). Imagine que alguém (o remetente) faz o embutimento de mais de uma mensagem secreta em uma imagem, e que o remetente seja forçado a revelar a mensagem secreta. Neste caso, o remetente pode revelar apenas uma das mensagens secretas (aquela de menor prejuízo) e afirmar que era a única mensagem escondida no esteganograma e sair ileso.

3.2.4 F5

O F5 é um algoritmo de esteganografia baseado em esteganografia no domínio de frequência (WESTFELD, 2001). Para executarmos a implementação do F5 precisamos do Java instalado, e baixarmos a implementação do site oficial[5]. Para desenvolver o F5, seu autor se inspirou no método usado pelo Jsteg (QIAO et al., 2015) e criou um algoritmo que ele chamou de F3. O Jsteg é resistente a ataques visuais e apresenta uma boa capacidade esteganográfica. Porém não é resistente a ataques estatísticos. Porém, o F3 continuava vulnerável a ataques estatísticos e por isso ele criou o F4 que corrige as vulnerabilidades de F3. Em seguida, criou F5 que melhora F4 em dois aspectos: espalhando as alterações feitas por todo o esteganograma usando uma técnica de permutação (*permutative straddling*); e também usando *matrix encoding* para melhorar a eficiência do embutimento. Para entendermos o F5, precisamos entender a evolução desses algoritmos (como o algoritmo anterior é melhorado pelo posterior). Por isso vamos ver superficialmente cada um desses.

O que deixa o Jsteg vulnerável a ataques é o fato de que ele substitui bits. Em função disso ele cria uma dependência entre os coeficientes que apenas diferem no LSB, igualando suas frequências. O Chi Square pode ser usado para detectar essa substituição de bits. O algoritmo F3 corrige essa falha do Jsteg ao evitar completamente a substituição de bits. Ao invés de sobrescrever os bits, ele decrementa o valor absoluto do LSB do coeficiente caso o LSB não coincida com o bit que deseja-se embutir. Então, por exemplo, ao tentar embutir o bit 1 no coeficiente 0000111, o algoritmo não fará nada. Porém ao tentar embutir o bit 1 no coeficiente 0000110, ele irá decrementar o coeficiente para 0000101, assim o coeficiente estará carregando o LSB com valor 1. O único coeficiente que não é possível decrementar seu valor absoluto é o coeficiente zero. E isso traz um problema para o algoritmo que o autor chama de *shrinkage*.

5 Página do F5: <https://code.google.com/archive/p/f5-steganography/>

Shrinkage acontece sempre que o algoritmo decrementa o valor absoluto dos coeficientes −1 e 1, resultando em zero. O receptor do esteganograma, não consegue diferenciar entre um zero que foi resultado de embutimento (produzido pelo decremento de -1 ou 1) e um zero que não foi usado. Dessa forma, o receptor salta todos os coeficientes de valor zero. É por isso o algoritmo que produz o esteganograma precisa repetir a operação de embutimento no coeficiente seguinte, quando ele percebe que produziu um zero e causa um problema que é revelado no histograma, em que haverá um adicional nas frequencias dos números pares, que é resultado da repetição do embutimento devido a *shrinkage*.

O algoritmo F4 melhora o F3, corrigindo essa fraqueza. Para isso, ele faz um mapeamento dos coeficientes negativos com o valor esteganográfico invertido. Isto é, coeficientes negativos pares representam um bit 1, e coeficientes negativos ímpares representam 0 (o contrário de F3). O F4 age da mesma maneira que o F3 sobre os coeficientes positivos. Assim, o histograma produzida pelo F4 não difere de forma significativa de um histograma de uma imagem original.

O F5 melhora dois aspectos do F4: a eficiência de embutimento, e o espalhamento uniforme das alterações pelo esteganograma. Ao melhorar a eficiência do embutimento, menos alterações são feitas nos coeficientes da imagem. E o espalhamento uniforme das alterações dificulta ataques de esteganálise, uma vez que as alterações não estão concentradas em uma única porção da imagem. O algoritmo do F5 primeiramente permuta todos os coeficientes. E então o algoritmo inicia o embutimento na sequência de coeficientes permutados. A permutação depende de uma chave que é derivada da senha utilizada. Com a chave correta, o receptor pode repetir a permutação e ler a mensagem. A permutação é eficiente pois é de complexidade linear.

A técnica *matrix encoding* também é utilizada no F5 para aumentar a eficiência do embutimento. O F5 infere de que, em geral, não se usa toda a capacidade esteganográfica possível da imagem. Quando isso é verdade, é possível usar a técnica de *matrix encoding* para diminuir o número de alterações nos coeficientes necessários. Podemos considerar que a metade da mensagem secreta causa mudanças. Isso equivale a uma eficiência de 2 bits para cada alteração. E em função do *shrinkage* causado pelo F4 esse valor é até um pouco menor. Mas usando matrix encoding, o F5 consegue alcançar uma eficiência de 3 bits por alteração ou mais, dependendo do caso.

Para entendermos como o matrix encoding funciona, vamos a um exemplo: suponha-se que queiramos embutir 2 bits x 1 e x 2 em 3 bits do arquivo de cobertura a 1 , a 2 e a 3 , alterando um único bit. Temos as quatro seguintes situações (⊕ é um símbolo para ‘ou exclusivo’):

- $x_1 = a_1 \oplus a_3$, $x_2 = a_2 \oplus a_3 \Longrightarrow$ Nenhum bit muda
- $x_1 \neq a_1 \oplus a_3$, $x_2 = a_2 \oplus a_3 \Longrightarrow a_1$ é alterado
- $x_1 = a_1 \oplus a_3$, $x_2 \neq a_2 \oplus a_3 \Longrightarrow a_2$ é alterado

- $x_1 \neq a_1 \oplus a_3$, $x_2 \neq a_2 \oplus a_3 \Longrightarrow a_3$ é alterado

Repare que nas quatro situações acima no máximo um bit é alterado. Ao mesmo tempo em que isso trás uma eficiência maior no embutimento, não é necessário que a mensagem secreta ocupe toda a capacidade esteganográfica do arquivo.

Westfeld (2001) criou o F5 como um desafio a comunidade científica, de um algoritmo de esteganografia resistente a ataques da época. Porém com o passar do tempo surgiram novos ataques e que são capazes de detectar imagens que passaram pelo F5. Um deles é o método descrito por Fridrich, Goljan e Hogea (2002b).

3.2.5 StegExpose

O StegExpose (BOEHM, 2015) é uma ferramenta que reúne a implementação de quatro diferentes técnicas de esteganálise: Primary Sets, Chi Square, Sample Pair, e RS Steganalysis. O StegExpose usou a implementação do RS Analysis e Sample Pair do site Digital Invisible Ink Toolkit[6].

A proposta do StegExpose é não somente agregar a implementação dos quatro métodos em um único programa, mas também de usar técnicas de fusão desses quatro métodos. Isso quer dizer que eles buscam construir algoritmos (fusores) que reúnam o resultado dos quatro métodos e dê um veredito final combinando os resultados individuais. A fusão padrão usa a média aritmética dos retornos dos detectores individuais.

Todos os detectores usados são automáticos e retornam uma porcentagem que refletem a probabilidade de o arquivo estar carregando uma mensagem secreta ou não.

4. RESULTADOS E DISCUSSÃO

Nesta seção serão apresentados e discutidos os resultados obtidos através da aplicação da metodologia apresentada no capítulo anterior. Na Seção 5.1 serão apresentados os resultados focando nas ferramentas de esteganografia e, na Seção 5.2, no desempenho dos métodos de esteganálise contra cada uma das ferramentas de esteganografia.

5.1 Testes por ferramenta de esteganografia

Primeiramente, aplicamos a esteganografia a todas as imagens do banco, conforme descrito no capítulo anterior. E analisamos todas as imagens usando o StegExpose com todos os seus parâmetros padrões, no qual o recomendado pelos autores é o uso de *threshold* de 0,2.

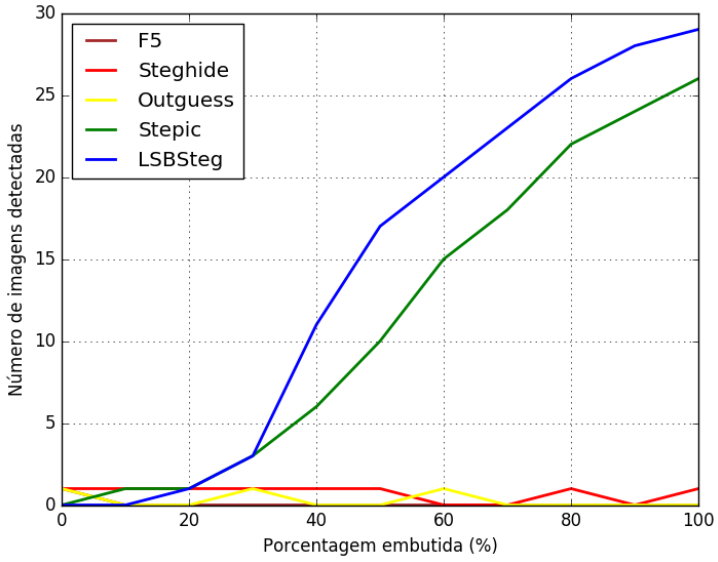
6 <http://diit.sourceforge.net/>

Na Tabela 1 podemos ver o número de imagens detectadas pelo StegExpose, para cada porcentagem embutida e para cada programa. Cada célula da tabela corresponde a um total de 30 imagens, exceto para a célula do resultado do Outguess com 100% (indicado com o *). A execução do Outguess para 100% falhou para algumas imagens, das 30 (trinta) imagens, 6 (seis) falharam, pois o tamanho da mensagem secreta excedeu o limite de embutimento. Por isso apenas 24 estego-imagens foram produzidas para 100% usando o Outguess. Usando os dados coletados, foi gerado o gráfico da Figura 5.

Tabela 1 – Número de imagens detectadas para cada ferramenta de esteganografia e porcentagem de embutimento para threshold de 0,2.

	F5	**Steghide**	**Outguess**	**Stepic**	**LSBSteg**
0%	1	1	1	0	0
10%	0	1	0	1	0
20%	0	1	0	1	1
30%	0	1	1	3	3
40%	0	1	0	6	11
50%	0	1	0	10	17
60%	0	0	1	15	20
70%	0	0	0	18	23
80%	0	1	0	22	26
90%	0	0	0	24	28
100%	0	1	0*	26	29

Figura 5 – Gráfico com o número de imagens detectadas (eixo y) para cada porcentagem embutida (eixo y), threshold de 0,2.



É possível se dizer que apenas os programas de esteganografia LSB (Stepic e LSBSteg) foram realmente detectados. Isso já era esperado, pois o StegExpose foi feito para detectar especificamente esteganografia LSB. Podemos ver que quase nenhuma estego-imagem gerada pelo Outguess, F5 e Steghide foram detectas.

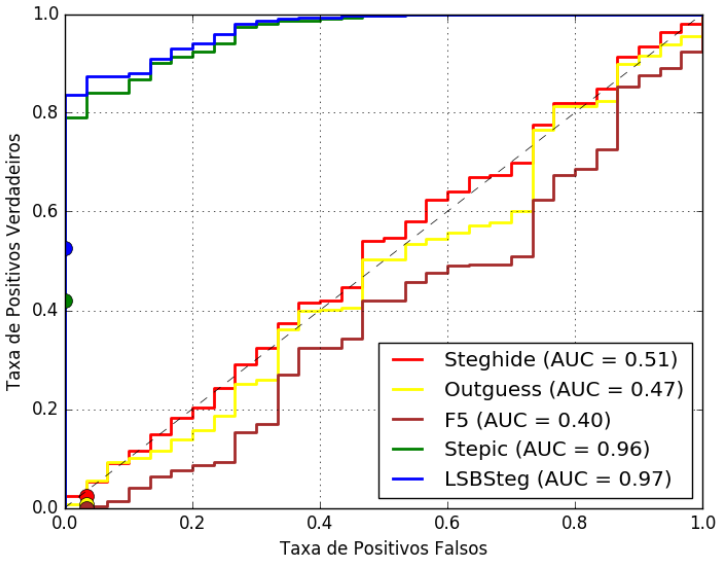
Podemos também notar que as estego-imagens produzidas pelo LSBSteg foram mais facilmente detectas do que aquelas produzidas pelo Stepic. Acreditamos que isto se deve ao embutimento do LSBSteg ser mais sequencial (contínuo) que o do Stepic. Observamos também que apenas a partir das estego-imagens com 60% de embutido é que o StegExpose foi capaz de detectar pelo menos metade das imagens. Assim, para as imagens estego-imagens produzidas pelo LSBSteg e o Stepic, quanto maior a quantidade de informação escondida, mais assertivo foi a detecção de que é uma estego-imagem.

Para a elaboração da curva ROC, varia-se o *threshold* do StegExpose. O StegExpose dá um *score* para cada imagem, que diz a probabilidade da imagem conter conteúdo esteganográfico. E somente após uma imagem receber seu *score*, ele compara tal *score* com o *threshold* fornecido para dar uma classificação binária da imagem (se ela contém uma mensagem secreta ou não).

Por isso, se o *threshold* é alterado, obtém-se classificações finais diferentes. Diminuindo o *threshold*, classifica-se mais imagens como positivas, porém ao custo de mais falsos positivos. E aumentando o *threshold*, classifica-se menos imagens como positivas, porém ao custo de menos verdadeiros positivos.

O StegExpose gera um relatório com cinco scores diferentes, para a curva ROC desta subseção, Figura 6, usou-se apenas o score da fusão dos quatro métodos de esteganálise utilizados. E com isso nós produzimos um gráfico ROC que representa o desempenho geral do StegExpose contra cada um dos programas de esteganografia utilizado.

Figura 6 – Curvas ROC para o desempenho da fusão do StegExpose contra cada um dos programas usados. Com pontos destacando o threshold de 0,2

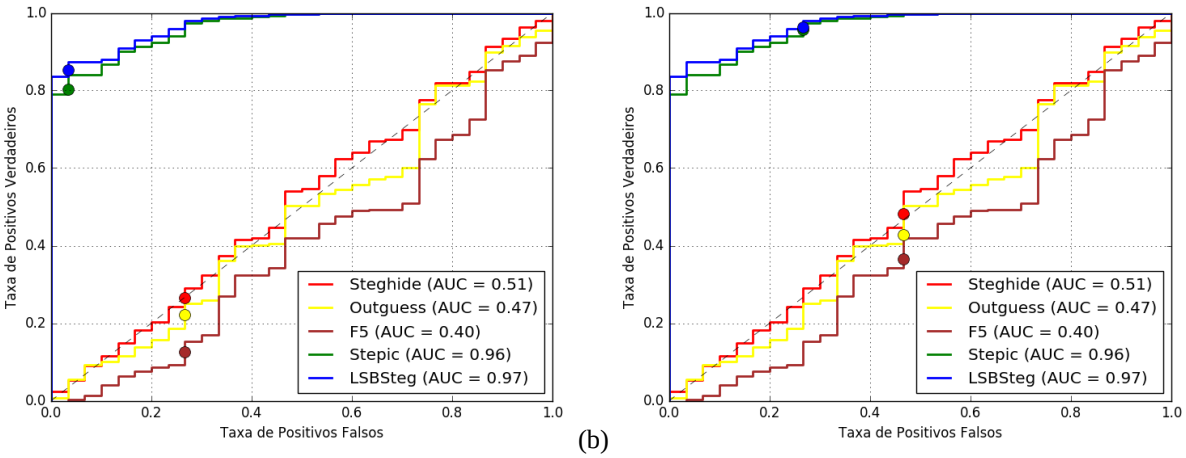


A curva é uma função escada, pois a avaliação foi feita para valores discretos de *threshold* (FAWCETT, 2006). E para visualizarmos no gráfico ROC o desempenho do StegExpose para o *threshold* padrão, plotou-se os 5 (cinco) pontos no gráfico da Figura 5.2 que correspondem ao *threshold* de 0,2 para cada um dos programas.

Observando os pontos marcados na Figura 6, vemos que o desempenho do StegExpose para o *threshold* de 0,2 não foi o melhor possível para o banco de imagens utilizado. No gráfico ROC, quanto mais próximo do ponto (0, 1), melhor é o desempenho daquele classificador. Se tivéssemos diminuído um pouco o *threshold*, teríamos obtido classificações melhores. O melhor resultado é obtido para o valor de *threshold* de 0,1, na (Figura 7a) mostram-se os pontos correspondentes.

Poderia haver uma dúvida se diminuíssemos mais o *threshold*, por exemplo, para 0,05, o resultado melhoraria. Analisando o resultado da curva ROC para 0,05 mostrado na Figura 7b. Observando as Figuras 7a e 7b, o valor de threshold 0,05 resulta em muitos falsos positivos.

Figura 7 – Curvas ROC para o desempenho da fusão do StegExpose contra cada um dos programas usados. (a) com pontos destacando o threshold de 0,1 e (b) com pontos destacando o threshold de 0,05



(a) (b)

Os três gráficos confirmam as nossas espectativas quanto ao gráfico ROC. Quanto maior o threshold, menos imagens são classificadas como positivas, o que ocasiona em uma menor taxa de verdadeiros positivos. Quanto menor o threshold, mais imagens são classificadas como positivas, e consequentemente teremos uma maior taxa de falsos positivos.

A Tabela 2 usa os dados obtidos para a classificação binária usando o threshold de 0,1, bem como o gráfico da Figura 8. Observando a tabela e o gráfico, vemos que conseguimos detectar muito mais estego-imagens para o Stepic e LSBSteg. Para o melhor *threshold*, o StegExpose detecta todas as imagens do LSBSteg com mais de 60% de embutimento de dados e do Stepic com mais de 80%.

Figura 8 – Gráfico com o número de imagens detectadas (eixo y) para cada porcentagem embutida (eixo y), threshold de 0,1.

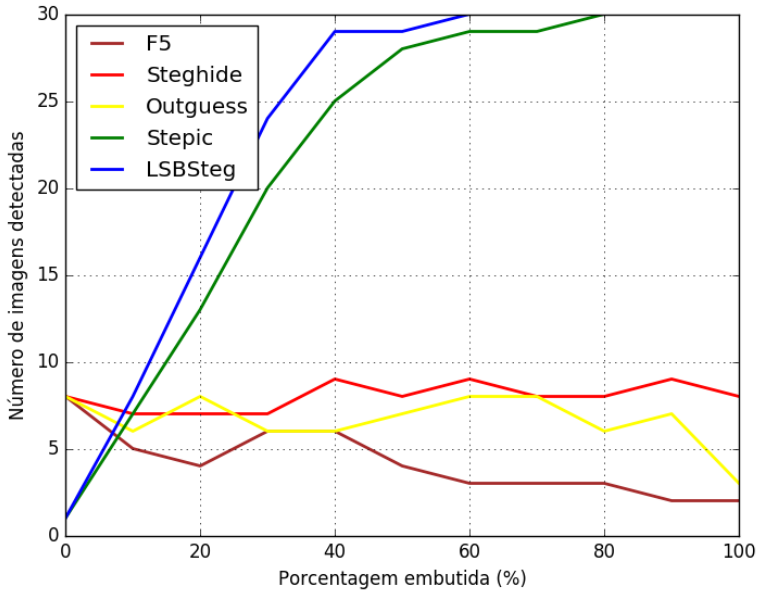


Tabela 2 – Número de imagens detectadas para cada ferramenta de esteganografia e porcentagem de embutimento, threshold de 0,1.

	F5	Steghide	Outguess	Stepic	LSBSteg
0%	8	8	8	1	1
10%	5	7	6	7	8
20%	4	7	8	13	16
30%	6	7	6	20	24
40%	6	9	6	25	29
50%	4	8	7	28	29
60%	3	9	8	29	30
70%	3	8	8	29	30
80%	3	8	6	30	30
90%	2	9	7	30	30
100%	2	8	3*	30	30

Vale a pena esclarecer que para obtermos os pontos mostrados nessas figuras, nós não executamos a análise do StegExpose novamente usando thresholds diferentes. Ao invés disso, nós usamos os scores que já temos nos relatórios do StegExpose, e refizemos a classificação binária para cada uma das imagens através de um pequeno script. Assim obtivemos os dados necessários para criarmos esses gráficos.

5.2 Testes por método de esteganálise

Nesta seção, analisa-se o desempenho de cada um dos métodos de esteganálise individualmente contra cada uma das ferramentas de esteganografia utilizadas. Para isso, gráficos ROC foram gerados, onde cada gráfico mostra o desempenho dos métodos de esteganálise usados contra aquela ferramenta de esteganografia específica. Os métodos de esteganálise que serão comparados aqui são: Chi Square, RS Analysis, Primary Sets, Sample Pair e também a técnica de fusão padrão do StegExpose (“Fusion” nos gráficos). Nas Figuras 8 e 9, apresentam-se o desempenho dos métodos contra o Stepic e LSBSteg respectivamente.

Figura 8 – Desempenho dos métodos de esteganálise contra o Stepic.

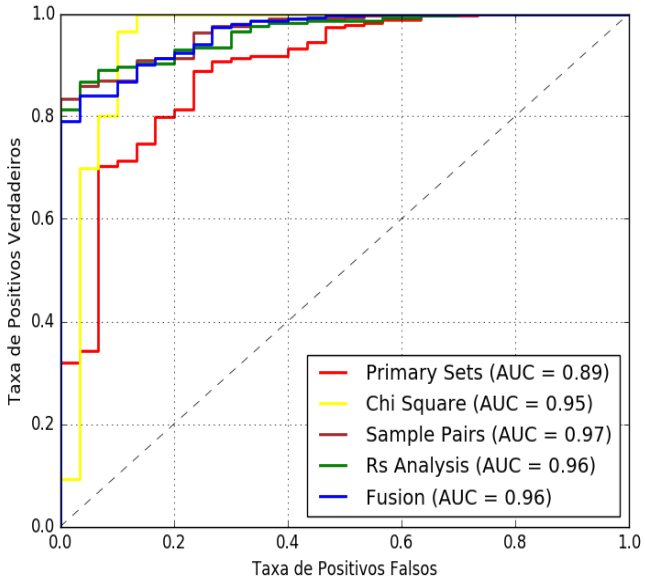
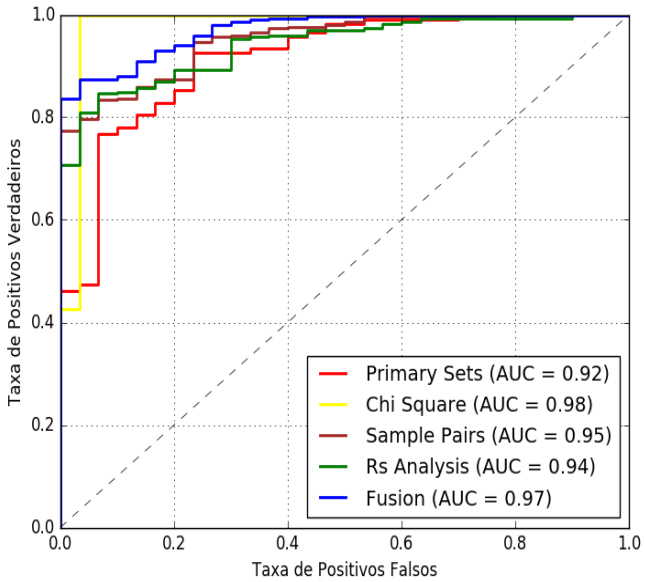


Figura 9 – Desempenho dos métodos de esteganálise contra o LSBSteg.



Com o método de esteganografia do Stepic, o melhor resultado, AUC de 0.97, foi o Sample Pairs, seguido do método RS e Fusion, ambos com AUC de 0.96. O pior desempenho foi o Primary Sets, com AUC de 0.89.

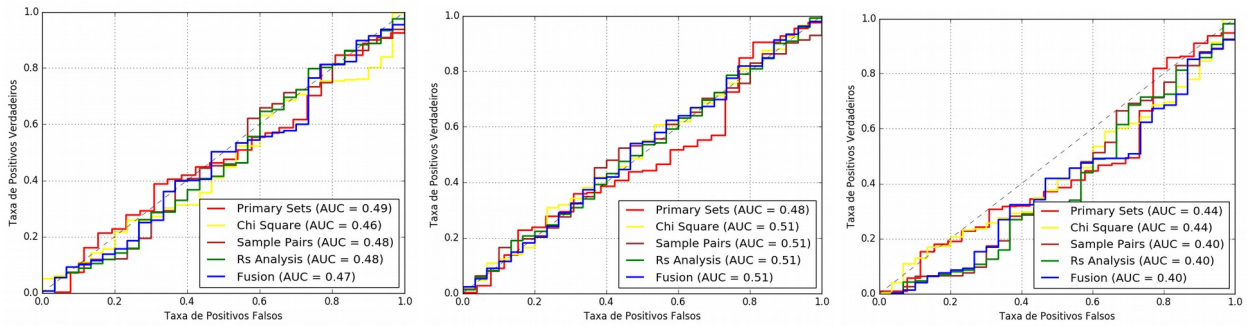
Para o método de esteganografia do LSBSteg, o melhor resultado, AUC de 0.98, foi o Sample Pairs, seguido do método do Fusion com AUC de 0.97. O pior desempenho foi o Primary Sets, de novo.

O Chi Square, de acordo com a literatura, realmente apresenta bom desempenho contra programas de esteganografia sequencial. Mas dificilmente detecta esteganografia LSB espalhada aleatoriamente pelo esteganograma, exceto quanto embutimos mensagens secretas muito grandes e

o esteganograma fica saturado. E veja que tanto o Stepic quanto o LSBSteg são programas de esteganografia LSB sequencial. Se tivéssemos usado algum programa de esteganografia LSB aleatório, os resultados para o Chi Square nesta seção provavelmente seriam diferentes. Essa questão é bem discutida em Kharrazi, Sencar e Memon (2006), no qual algoritmos específicos de esteganálise são desenvolvidos para métodos específicos de esteganografia.

Também geramos as curvas ROC para as imagens geradas pelas ferramentas Steghide, o F5 e o Outguess como pode ser visto na Figura 10 (a), (b) e (c). respectivamente. E como podemos constatar, a detecção de cada um dos métodos é completamente aleatória para estes programas, dado que suas curvas são bem próximas à reta de 45 o dos gráficos.

Figura 10 – Curvas ROC para o desempenho dos métodos de esteganálise contra Steghide (a), o F5 (b) e Outguess (c).



(a) (b) (c)

Os resultados deste trabalho são diferentes dos resultados do trabalho de Boehm (2014), que mostra que o método de fusão do StegExpose é sempre melhor. Um dos motivos é que as bases de dados de imagens são diferentes e as ferramentas de esteganografia são diferentes.

5. CONSIDERAÇÕES FINAIS

A esteganografia é uma técnica usada para esconder a comunicação, e pode ser usada em conjunto com a criptografia. Diferente da criptografia, que busca tornar a comunicação ilegível para qualquer um que não seja o receptor, a esteganografia busca ocultar que qualquer comunicação sequer tenha ocorrido. Ainda assim, muitos sistemas esteganográficos criptografam a mensagem secreta para posteriormente embutí-lo no arquivo de cobertura.

A esteganálise é o estudo que busca detectar mensagens escondidas através da esteganografia. Em geral, a esteganálise não busca descobrir o conteúdo da mensagem escondida, mas busca detectar se um arquivo passou por esteganografia ou não, e qual poderia ser o tamanho da mensagem secreta.

Para aplicarmos as técnicas de esteganografia e esteganálise, foram selecionados cinco programas de esteganografia, três deles de esteganografia LSB (Stepic, LSBSteg e Steghide) e outros dois que utilizam algoritmos próprios (Outguess e F5) baseados em transformadas de

frequencia. E selecionamos um programa de esteganálise chamado StegExpose que reuni quatro importantes métodos de esteganálise LSB (Chi Square, RS Analysis, Primary Sets e Sample Pair)

Elaborou-se um banco de imagens contendo 30 imagens coloridas na resolução de 400x400 em JPEG, e com cópias em PNG, para servirem de arquivos de cobertura para a esteganografia. E coletamos alguns livros do Project Gutenberg para formar um banco de texto para ser usado como mensagem secreta. Então usou-se um ambiente automatizado para embutir mensagens nas imagens limpas usando cada um dos programas de esteganografia mencionados, e usando também diferentes tamanhos de mensagem secreta. Ao todo foram produzidas quase 1.500 estego-imagens. Usamos o StegExpose para analisar cada uma das imagens (limpas e estego), usando o seu *threshold* padrão (0,2).

Coletamos as informações geradas pelo StegExpose e foi possível notar que o StegExpose detectou um percentual de estego-imagens produzidas pelo Stepic e LSBSteg, mas não detectou praticamente estego-imagem alguma produzida pelo Steghide, Outguess e F5. Algo que era esperado, pois o StegExpose é uma ferramenta de esteganálise específica para detectar esteganografia LSB.

Produzimos curvas ROC demonstrando o desempenho do StegExpose contra cada um dos programas de esteganografia usados, e constatamos que a detecção do Steghide, Outguess e F5 é completamente aleatória, enquanto a detecção do Stepic e LSBSteg é muito boa. Observando as curvas ROC pudemos ver que o desempenho do StegExpose contra o Stepic e o LSBSteg poderia ser melhor se tivéssemos usado um *threshold* de 0,1, diferente do valor padrão recomendado pelos autores da ferramenta.

Também foram elaboradas curvas ROC para análise de desempenho das ferramentas e métodos de esteganálise. Não foi possível indicar uma única ferramenta que tenha tido sempre o melhor desempenho, e sim, verificar que o desempenho de uma ferramenta de esteganálise depende do método de esteganografia usado. Este resultado está de acordo com a bibliografia, dado que foram utilizadas técnicas de esteganálise específicas.

Tambpem foi verificado que quanto maior o texto embutido, aumenta a acurácia de acerto na avaliação de uma estego-imagem.

O objetivo deste trabalho foi alcançado. Porém algumas alterações podem ser feitas para alcançar novos resultados, ou melhorar os resultados atuais:

- Repetir os testes com um banco de imagens maior e mais variado, usando imagens com diferentes espaços de cores, resoluções e formatos.
- Aplicar esteganografia em outros tipos de dados como, por exemplo, áudio, vídeo e texto, explorando diferentes tipos de mídia.

- Usar outras programas e métodos de esteganálise e fazer um estudo comparativo dos mesmos. É possível que outros programas possam ter desempenho melhor que o StegExpose para diferentes sistemas esteganográficos.
- Utilizar outras ferramentas de esteganografia em imagens e repetir os testes (FRIDRICH; GOLJAN; HOGEA, 2002a). Pode ser interessante usar esteganografia LSB não-sequencial, para ver a diferença no desempenho do Chi Square comparado ao restante dos métodos de esteganálise.
- Pesquisar correlações entre características estatísticas globais de uma imagem com a sua capacidade de ocultar mensagens secretas. Por exemplo, uma imagem muito simples que contém apenas uma única cor possui capacidade praticamente zero para ocultar mensagens, por outro lado, uma imagem com muitos objetos num ambiente com texturas variadas como grama ou areia podem esconder mensagens. Assim, dado uma imagem, seria possível estimar o tamanho de mensagem a ser escondida.

REFERÊNCIAS

BOEHM, B. (2014). Stegexpose - A tool for detecting LSB steganography. CoRR, abs/1410.6656, 2014. Disponível em: <http://arxiv.org/abs/1410.6656>. Acesso em <01 de jan. de 2018>.

BOEHM, B. (2015). StegExpose. Disponível em: <https://github.com/b3dk7/StegExpose>. Acesso em <01 de jan. de 2018>.

CANCELLI, G. (2009). New techniques for steganography and steganalysis in the pixel domain. Tese (Doutorado) — Technische Universität Dresden.

CHANDRAMOULI, R.; KHARRAZI, M.; MEMON, N. (2003). Image steganography and steganalysis: Concepts and practice. In: Springer. International Workshop on Digital Watermarking. [S.l.], 2003. p. 35–49.

CHEDDAD, A. et al. (2010) Digital image steganography: Survey and analysis of current methods. Signal processing, Elsevier, v. 90, n. 3, p. 727–752. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0165168409003648>.

CHOUDHARY, K. (2012). Image steganography and global terrorism. International Journal of Scientific & Engineering Research, v. 3, n. 4, p. 12.

DAVID, R. (2015). LSB-Steganography. Disponível em: <https://github.com/RobinDavid/LSB-Steganography>. Acesso em <01 de jan. de 2018>.

DOMNITSER, L. (2007) Spy vs. Spy. Disponível em: <http://domnit.org/stepic/doc/>. Acesso em <01 de jan. de 2018>.

DUMITRESCU, S.; WU, X.; MEMON, N. (2002). On steganalysis of random LSB embedding in continuous-tone images. In: IEEE. Image Processing. 2002. Proceedings. 2002 International Conference on. [S.l.]. v. 3, p. 641–644.

DUMITRESCU, S.; WU, X.; WANG, Z. (2003). Detection of LSB steganography via sample pair analysis. Signal Processing, IEEE Transactions on, IEEE, v. 51, n. 7, p. 1995–2007, 2003.

FAWCETT, T. (2006). An introduction to ROC analysis. Pattern recognition letters, Elsevier, v. 27, n. 8, p. 861–874,.

FRIDRICH, J.; GOLJAN, M. (2002). Practical steganalysis of digital images: state of the art. In: International Society for Optics and Photonics. Electronic Imaging. [S.l.], 2002. p. 1–13.

FRIDRICH, J.; GOLJAN, M.; DU, R. (2001a). Reliable detection of LSB steganography in color and grayscale images. In: ACM. Proceedings of the 2001 Workshop on Multimedia and security: new challenges. [S.l.]. p. 27–30.

FRIDRICH, J.; GOLJAN, M.; DU, R. (2001b). Steganalysis based on JPEG compatibility. In: International Society for Optics and Photonics. ITCom 2001: International Symposium on the Convergence of IT and Communications. [S.l.]. p. 275–280.

FRIDRICH, J.; GOLJAN, M.; HOGEA, D. (2002a). Attacking the Outguess. In: Proceedings of the ACM Workshop on Multimedia and Security. Vol. 2002. Juan-les-Pins, France.

FRIDRICH, J.; GOLJAN, M.; HOGEA, D. (2002b). Steganalysis of JPEG images: Breaking the F5 algorithm. In: SPRINGER. International Workshop on Information Hiding. [S.l.]. p. 310–323.

GOLJAN, M.; FRIDRICH, J. J.; DU, R. (2001). Distortion-free data embedding for images. In: Srpinger. International Workshop on Information Hiding. [S.l.]. p. 27–41.

HETZL, S. (2002). Steghide. Disponível em: <http://steghide.sourceforge.net/index.php>. Acesso em <01 de jan. de 2018>.

NSTC, (2006). Federal Plan for Cyber Security and Information Assurance Research and Development. National Science and Technology Council, Washington DC. Disponível em: http://www.au.af.mil/au/awc/awcgate/nitrd/fed_plan_csia_rese.pdf. Acesso em <01 de jan. de 2018>.

JAMES, G. et al. (2013). An introduction to statistical learning. [S.l.]: Springer. New York.

JOHNSON, N. F. (2012). Steganography Software. Disponível em: <http://www.jjtc.com/Steganography/tools.html>. Acesso em <01 de jan. de 2018>.

JULIO, E. P.; BRAZIL, W. G.; ALBUQUERQUE, C. V. N. (2007). Esteganografia e suas aplicações. Livro de Minicursos do SBSEG. Rio de Janeiro: Sociedade Brasileira de Computação, v. 7, p. 54–102, 2007.

KHARRAZI, M.; SENCAR, H. T.; MEMON, N. (2006). Improving steganalysis by fusion techniques: a case study with image steganography. In: Transactions on Data Hiding and Multimedia Security I. [S.l.]: Springer. p. 123–137.

MARÇAL, A. R.; PEREIRA, P. R. (2005). A steganographic method for digital images robust to RS steganalysis. In: Springer. International Conference Image Analysis and Recognition. [S.l.], 2005. p. 1192–1199.

MAZURCZYK, W. et al. (2016). Information Hiding in Communication Networks: Fundamentals, Mechanisms, Applications, and Countermeasures. Wiley, 2016. IEEE Press Series on Information and Communication Networks Security. ISBN 9781118861691. Disponível em: <https://books.google.com.br/books?id=KPhcCgAAQBAJ>.

MENA, M. E. Emotion recognition from speech signals. Tese (Doutorado) — Universitat Politècnica de Catalunya, 2012.

MICHE, Y. (2010). Developing fast machine learning techniques with applications to steganalysis problems. Tese (Doutorado) — Institut National Polytechnique de Grenoble-INPG, 2010.

NAIR, T. et al. (2012). Genetic algorithm to make persistent security and quality of image in steganography from RS analysis. arXiv preprint arXiv:1204.2616, 2012.

PETITCOLAS, F. A.; ANDERSON, R. J.; KUHN, M. G. (1999). Information hiding-a survey. Proceedings of the IEEE, IEEE, v. 87, n. 7, p. 1062–1078, 1999.

PROVOS, N. (2001). Defending against statistical steganalysis. In: Usenix security symposium. [S.l.: s.n.], 2001. v. 10, p. 323–336.

PROVOS, N.; HONEYMAN, P. (2003). Hide and seek: An introduction to steganography. IEEE Security & Privacy, IEEE, v. 1, n. 3, p. 32–44, 2003.

QIAO, T. et al. (2015). Steganalysis of Jsteg algorithm using hypothesis testing theory. EURASIP Journal on Information Security, v. 2015, n. 1, p. 2, 2015. ISSN 1687-417X. Disponível em: <http://dx.doi.org/10.1186/s13635-015-0019-7>.

RAJAN, A. N. et al. (2013). Block mean modulation: An effective and robust image steganographic technique in the spatial domain. International Journal of Computer Applications, Foundation of Computer Science, v. 73, n. 19.

SHACHTMAN, N. (2010). FBI: Spies Hid Secret Messages on Public Websites. 2010. Disponível em: <https://www.wired.com/2010/06/alleged-spies-hid-secret-messages-on-public-websites/>. Acesso em <01 de jan. de 2018>.

SIMMONS, G. J. (1984). The prisoners' problem and the subliminal channel. In: . Advances in Cryptology: Proceedings of Crypto 83. Boston, MA: Springer US, 1984. p. 51–67. ISBN 978-1-4684-4730-9. Disponível em: <http://dx.doi.org/10.1007/978-1-4684-4730-9_5>.

SUJATHA, P. (2013). Image steganalysis using artificial neural networks. Tese (Doutorado) — Vels University.

TEAM, O. D. (2017). OpenCV (Open Source Computer Vision Library). 2017. Disponível em: <http://opencv.org/>. Acesso em <01 de jan. de 2018>.

WENDZEL, S. et al. (2014). Hidden and uncontrolled–on the emergence of network stegaographic threats. In: ISSE 2014 Securing Electronic Business Processes. [S.l.]: Springer. p. 123–133.

WESTFELD, A. (2001). F5 - a steganographic algorithm. In: Springer. International Workshop on Information Hiding. [S.l.]. p. 289–302.

WESTFELD, A.; PFITZMANN, A. (1999). Attacks on steganographic systems. In: Springer. Information Hiding. [S.l.]. p. 61–76.