SEA – Sistema Esteganográfico de Arquivos

Fernando de O. Gil[1], Leandro José A. A. Malandrin[1], Roberto H. Morigaki[1], Paulo S. L. M. Barreto[1*].

[1]Departamento de Engenharia de Computação e Sistemas Digitais,
Escola Politécnica, Universidade de São Paulo, Brasil

{fernando.gil,leandro.malandrin,roberto.morigaki}@poli.usp.br,
pbarreto@larc.usp.br

Abstract. *This work describes a kind of file system based on a technique known as steganography. This file system, called SEA, tries to hide not only the information contents, but also its existence, therefore having the potential to become a tool used for privative information storage. Due to the fact that there is no way to detect the existence of data previously stored, the system uses cryptography and data dispersion in a way that makes possible the data recovery even when collision happens. The contributions of this work are the solution for security issues found when the two techniques mentioned are used together and the analysis of the results obtained with storage of large files.*

Resumo. *Nesse trabalho descreve-se um sistema de arquivos baseado em uma técnica conhecida como esteganografia. Esse sistema, chamado SEA, procura esconder tanto o conteúdo da informação como a própria existência desta, tendo assim o potencial de se utilizado no armazenamento de informações privativas. Devido à impossibilidade de detectar a existência de arquivos previamente armazenados, o sistema utiliza criptografia e dispersão de dados para permitir a recuperação de arquivos mesmo quando colisões acontecem. Este trabalho contribui com uma solução para problemas de segurança encontrados na utilização do conjunto das técnicas mencionadas e com a avaliação dos resultados para o armazenamento de arquivos grandes.*

1. Introdução

A esteganografia é a arte de esconder a existência de alguma coisa, de forma que a torne praticamente impossível de ser descoberta por uma pessoa que não conhece o segredo. Apesar do conceito simples de ser aplicado na vida real, dentro de sistemas computacionais a utilização da esteganografia é mais complicada. Baseando-se nas formas de uso dessa técnica no mundo real foram criadas as bases para o seu uso em sistemas computacionais. Assim, da mesma forma que uma pessoa pode utilizar as chamadas “tintas invisíveis” para escrever mensagens secretas em quadros, utilizam-se bits menos visíveis em imagens digitais para esconder informações.

---

* Orientador do trabalho. Bolsista de Produtividade em Pesquisa CNPq, processo 312005/2006-7.

Um sistema esteganográfico de arquivos é uma forma de se armazenar arquivos em um computador, de forma que a existência destes seja ocultada. Os dados são armazenados no disco rígido escondidos em uma partição escondida ou em um arquivo binário. No entanto, dadas as características desse sistema, os dados não podem ser identificados, mesmo que cada byte e porta lógica do hardware sejam investigados.

Quando um novo arquivo é armazenado, não devem existir informações relativas às posições dos dados gravados anteriormente. Com isso surge o problema da sobrescrita de arquivos, determinando um tempo de vida máximo para os arquivos.

O SEA, sigla de Sistema Esteganográfico de Arquivos, é uma aplicação desse tipo, baseada em idéias de trabalhos semelhantes feitos no exterior. O SEA foi desenvolvido como trabalho de conclusão de curso em 2007 resultando em um conjunto de aplicações para verificar e avaliar este uso da esteganografia. Uma das principais contribuições do trabalho é o levantamento de problemas de segurança que podem acontecer com a utilização da criptografia e dispersão de dados em conjunto, e a implementação de soluções para esse problema. Além disso, foi feita a análise da capacidade do sistema.

2. Esteganografia

Esteganografia é a arte de esconder informações de forma que nenhuma pessoa, fora o seu criador e a pessoa a qual a mensagem se destina, consiga detectar sequer a existência dessas informações. A palavra esteganografia vem do grego escrita escondida. Um dos primeiros casos conhecidos de uso da esteganografia é atribuído a um general grego chamado Histiaeus, que tatuou uma mensagem na cabeça de um escravo que teve seu cabelo raspado. Depois que o cabelo do escravo cresceu, ele foi enviado para entregar a mensagem.

O livro Steganographia, publicado em 1499 por Trithemius em três volumes, tratava sobre esteganografia e criptografia, só que disfarçado como um livro sobre magia, espíritos e assuntos do tipo. Os primeiros 2 volumes, aparentemente tratavam sobre o uso de espíritos para a comunicação em longas distâncias. Por volta de 1600, uma “chave” de decodificação do livro foi publicada e descobriu-se que esses volumes tratavam de esteganografia e criptografia. O terceiro volume, até recentemente era acreditado como sendo realmente sobre magia negra. Na verdade, tratavam apenas de mais conteúdo do mesmo tipo, codificado utilizando uma pequena variação da cifra utilizada nos dois primeiros volumes [Reeds 1998].

A criptografia utiliza ferramentas matemáticas para manipular os dados originais até se obter uma nova seqüência de dados, derivada a partir da original e de uma chave de criptografia, conseguindo assim, proteger o conteúdo da informação. Essas duas técnicas não são de forma alguma incompatíveis. De fato, muitas aplicações buscam utilizar ambas em conjunto, implementando assim um principio de segurança da informação conhecido como defesa em profundidade. No caso do SEA, a utilização da criptografia é fundamental para a viabilidade do sistema.

Em um trabalho recente, [Dusi *et al* 2008] apresentou uma técnica que permite identificar se uma conexão cifrada é uma conexão SSH que está sendo utilizada como túnel de outras conexões. Dessa forma, o administrador do servidor SSH pode bloquear

esta comunicação do usuário sem de fato saber que tipo de dados está sendo trafegado, apenas identificando que a comunicação existe.

A esteganografia aplicada a imagens busca transformar os bits de *pixels* menos significativos nos bits da mensagem que se deseja esconder. Logicamente, esse processo envolve uma perda da qualidade da imagem original. No entanto, dependendo do algoritmo utilizado, a imagem contendo a mensagem secreta e a imagem original não apresentam diferenças que possam ser identificadas a olho nu pelo ser humano.

3. Esteganografia em Sistema de Arquivos

As técnicas descritas anteriormente necessitam de um arquivo em um formato e com propriedades especificas no qual a mensagem será inserida. No entanto, para a proteção de grandes quantidades de dados é necessária uma nova abordagem.

A esteganografia pode ser aplicada em sistemas de arquivos, de forma que mensagens secretas de qualquer espécie possam ser escondidas em meio a dados também de qualquer espécie [Anderson *et al* 1998]. No sistema citado, inicialmente é realizada a gravação de dados aleatórios no disco rígido do usuário. Esse arquivo é criptografado e quebrado em blocos de dados. Esses blocos são gravados no espaço físico disponível nas posições indicadas por uma seqüência pseudo-aleatória derivada da chave do usuário e de algumas outras propriedades do arquivo e do sistema. Dessa forma, sem o conhecimento da chave do usuário, uma pessoa que está observando os dados no disco não consegue distinguir o que faz parte dos arquivos do usuário e o que é informação aleatória. Em [Hand e Roscoe 2002] é apresentado o sistema de arquivos *Mnemosyne*, que utiliza este conceito, porém utilizando uma plataforma distribuída de armazenamento.

Esses sistemas apresentam a vantagem de permitir a gravação de dados em qualquer formato e em qualquer quantidade sobre um substrato de dados aleatórios qualquer. Entretanto, o fato do sistema não poder armazenar informações sobre a localização dos arquivos (como, por exemplo, em uma tabela de alocação de arquivos) gera um novo problema. A cada arquivo gravado, e até durante a gravação de um mesmo arquivo, existe a possibilidade de ocorra uma colisão, ou seja, que dois blocos diferentes sejam escritos no mesmo lugar, corrompendo a integridade de um dos arquivos do disco.

A solução desse problema envolve a utilização de técnicas de redundância, e constituem um dos pontos mais interessante desses sistemas. A garantia da possibilidade de recuperação do arquivo fica associada a características do arquivo e do sistema. Não é possível garantir que um arquivo tem 100% de chance de ser recuperado com sucesso. Entretanto é possível calibrar variáveis do sistema tais como tamanho dos blocos, tamanho do disco e número de usuários de forma a garantir uma chance muito alta de sucesso. Entre as técnicas de redundância que podem ser utilizadas estão a cópia dos blocos em diversas posições do volume esteganográfico [Anderson *et al* 1998] e os algoritmos de dispersão de dados, apresentados em [Rabin 1989] utilizados no sistema *Mnemosyne* em [Hand e Roscoe 2002].

O fato de existir uma probabilidade de perda de blocos de um arquivo faz com que o arquivo armazenado tenha um tempo de vida dentro do sistema. Para estender este

tempo de vida, pode ser utilizado o processo de renovação, no qual o usuário recupera o arquivo e o armazena na seqüência. Esta é uma outra forma de redundância que precisa ser tratada nos sistemas de arquivos que implementam a esteganografia.

4. Implementação

Baseando-se nos conceitos apresentados, foram desenvolvidas duas aplicações. A primeira é um sistema esteganográfico de arquivos (SEA), que tem a capacidade de armazenar arquivos do usuário em um "volume esteganográfico". Esta aplicação possui uma interface gráfica e permite que o usuário oculte seus arquivos dentro de um grande arquivo binário de conteúdo aleatório. A segunda aplicação é uma plataforma para realização de testes do algoritmo de esteganografia. A plataforma possui uma interface através da linha de comando, permitindo a realização de testes em lote. Os resultados permitem a verificação da taxa de renovação necessária e da capacidade de armazenamento máxima do sistema. O SEA foi implementado utilizando-se a linguagem Java.

4.1. Arquitetura

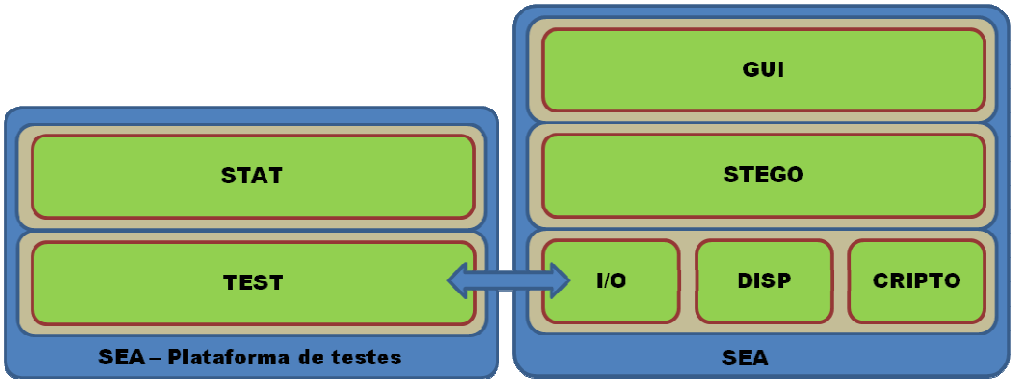

Figura 1. Arquitetura do SEA.

Para facilitar a criação das duas aplicações, o SEA foi estruturado em camadas, conforme apresentado na Figura 1. Cada uma das camadas da aplicação principal possui métodos para gravação e recuperação dos dados dentro do volume esteganográfico. Na figura, os módulos da camada 1 (I/O, Disp e Cripto) implementam o algoritmo de esteganografia e funcionam como suporte para as operações da camada 2 (Stego), que contém a lógica da aplicação principal. Esta por sua vez é utilizada pela interface gráfica (GUI) a qual é utilizada pelo usuário.

A plataforma de testes se integra com a aplicação principal na camada 1, possuindo uma camada de controle dos testes (Test) e uma camada para tratamento estatístico dos dados (Stat).

4.2. Interface Gráfica

A interface com o usuário foi planejada para ser semelhante ao de softwares gerenciadores de arquivos mais conhecidos do mercado, de modo a facilitar o uso por usuários que já estejam familiarizados com estes programas. Todas as funcionalidades do SEA podem ser acionados através de objetos gráficos. A Figura 2 apresenta o aspecto da interface desenvolvida.

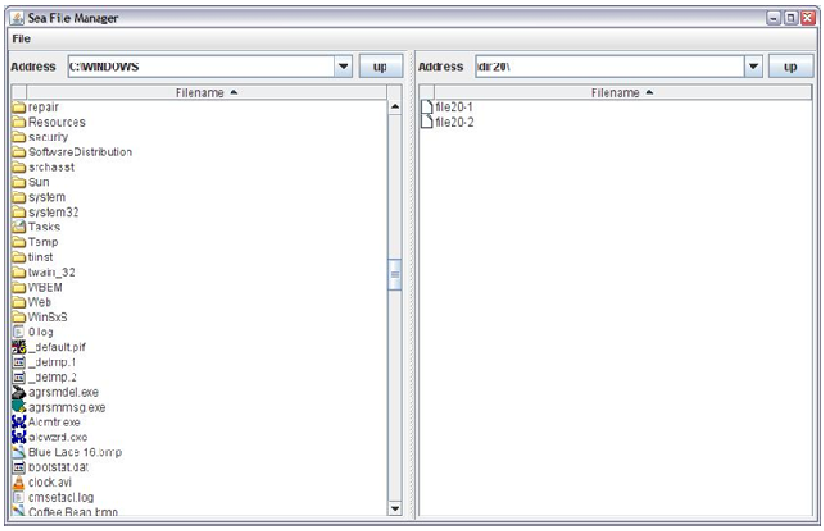

Figura 2. Interface gráfica do SEA.

4.3. Processo de Esteganografia

Conforme apresentado na Figura 3, o arquivo é criptografado por completo, no módulo de criptografia 1. Em seguida ele é dividido em blocos no módulo de empacotamento, onde cada bloco recebe um cabeçalho de identificação. Os blocos empacotados seguem para o módulo de dispersão, que adiciona uma redundância aos dados, pois alguns destes podem ser perdidos por colisões no volume esteganográfico. Os blocos dispersados seguem para o módulo de criptografia 2, onde são cifrados individualmente cada um com uma chave diferente, derivada da chave principal. Os blocos, agora, estão prontos para serem gravados no volume.

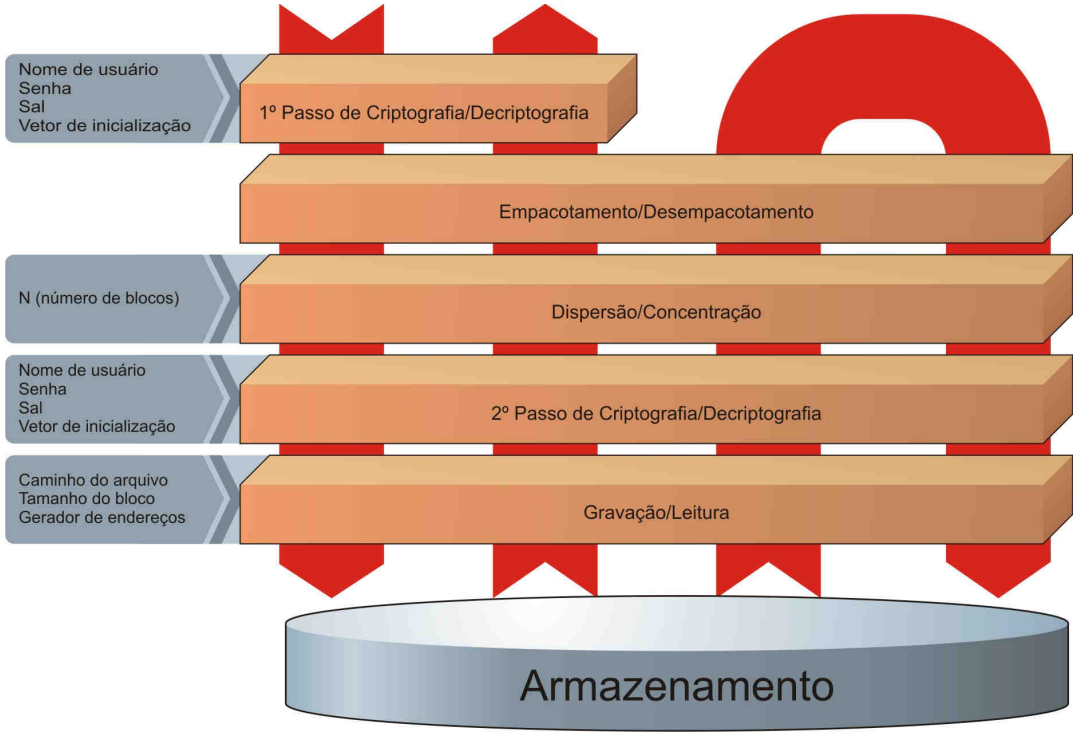

Figura 3. Processo de esteganografia, à esquerda, e de renovação, à direita.

A camada de gravação inicia o gerador de números pseudo-aleatórios com as informações do arquivo e do usuário, e então dá início à gravação dos blocos armazenando cada um em um endereço obtido a partir do gerador. Dessa forma o arquivo do usuário está agora “esteganografado” no volume.

Para efetuar a recuperação de um arquivo, o usuário deve fornecer a semente utilizada. A primeira etapa do processo é, na camada de leitura, iniciar o gerador de números pseudo-aleatórios com as mesmas informações utilizadas para a gravação. Feito isto, a camada passa a ler os blocos no volume seguindo a mesma seqüência de gravação. Os blocos passam pela camada de decriptografia 2, e então são agrupados na camada de concentração.

Este segundo passo de criptografia tem como objetivo impedir que um atacante descubra, através do processo de concentração, que um conjunto de dados pertence a um arquivo válido. O atacante poderia tentar realizar a restauração de blocos aleatórios até obter um trecho válido do arquivo. Apesar de não conseguir recuperar o arquivo, ele pode destruir a informação e atestar existência de dados, comprometendo assim a segurança oferecida pela esteganografia.

A camada de concentração testa cada bloco até obter o número mínimo de blocos válidos para a recuperação da informação dispersada e a recupera. Em seguida a camada de desempacotamento verifica se a ordem dos blocos está correta, retira o cabeçalho e os agrupa, formando novamente a unidade do arquivo. A camada de decriptografia 1 decifra o arquivo inteiro obtendo a sua forma original. O SEA então grava o arquivo de volta no sistema de arquivos convencional, conforme especificado pelo usuário no início do processo.

4.4. Renovação

Uma das medidas de avaliação da confiabilidade do SEA é o tempo de vida do arquivo armazenado. Uma vez que colisões provavelmente irão ocorrer, com o passar do tempo existe uma maior probabilidade de um arquivo ser corrompido. Assim, é interessante permitir que seja feito uma regravação dos blocos relativos a um mesmo arquivo.

O processo de renovação dos arquivos (Figura 3) é feito pelo usuário, uma vez que os dados precisam ser decifrados para que seu conteúdo seja verificado. A renovação segue as mesmas etapas apresentadas no item anterior, com exceção da primeira etapa de criptografia. Dado que existe um processo de redundância, não é necessário revelar totalmente o conteúdo dos dados, bastando apenas que o arquivo completo possa ser recuperado na sua forma cifrada para que este seja renovado.

4.5. Criptografia

Para a maior parte dos sistemas não faz sentido exigir que os usuários se lembrem dos valores das chaves de criptografia utilizadas. Dessa forma, foi utilizado um algoritmo de derivação para as chaves de criptografia, a partir das senhas utilizadas pelo usuário durante a esteganografia. O algoritmo escolhido foi o PBKDF2 [Kaliski 2000]. O modo utilizado para a criptografia, em cadeia, é o CFB (Cipher Feedback), que permite que dados sejam criptografados sem que o resultado apresente um tamanho maior que os dados originais. Essa é uma característica desejada, uma vez que o controle do tamanho dos dados de entrada e saída de cada módulo precisava ser feito.

O segundo passo de criptografia não precisa ser executado com chaves tão fortes quanto o primeiro, uma vez que só visa acabar com as chances de um atacante descobrir os blocos pertencentes ao mesmo usuário dentro do volume esteganográfico. Sendo assim para o primeiro passo de criptografia utilizam-se chaves de 256 bits, enquanto que para o segundo passo, 128.

4.6. Empacotamento

Uma vez que para a recuperação do arquivo só é necessária a seqüência de endereços utilizados para armazenar cada um dos blocos, foi necessária a criação de uma forma de identificação da quantidade de blocos vinculada a cada arquivo. A camada de empacotamento insere o tamanho do arquivo e um código de verificação em cada bloco, para que o sistema reconheça o final do arquivo. O armazenamento de um simples bloco de fim poderia realizar esta função, porém, como pode ocorrer perda de blocos, optou-se por colocar o critério de parada em todos os blocos, de forma que a perda do último bloco não acarrete na perda total do arquivo.

4.7. Dispersão

A dispersão é a etapa de redundância de dados do sistema. O algoritmo utilizado na dispersão de dados foi baseado no apresentado em [Rabin 1989], no entanto foi adaptado para sua utilização dentro do SEA. A idéia básica da dispersão é a visualização do conteúdo de entrada como um elemento geométrico, tal como uma reta ou um plano. Por exemplo, para se armazenar uma reta, precisa-se de dois pontos. Porém, caso sejam armazenados mais pontos, existirá informação mais do que suficiente para a reconstrução da reta. Dessa forma, duas informações (os coeficientes da reta) são armazenadas em n valores (pontos).

No SEA, essa técnica é feita de maneira semelhante, são utilizados 5 blocos do arquivo para gerar novos blocos redundantes, em geral são gerados 10, porém este número varia de acordo com o configurado, e será variado para a realização dos testes. Assim, dentro do módulo Disp, existem dois algoritmos principais implementados: o primeiro é utilizado durante o armazenamento dos dados no volume esteganográfico, chamado de algoritmo de dispersão, enquanto que o segundo é utilizando durante a recuperação desses dados, chamado algoritmo de recuperação. A recuperação consiste na resolução de um sistema linear, tal como para a recuperação dos coeficientes de uma reta dados seus pontos. Vale ressaltar que são necessários apenas o número de blocos de entrada para a recuperação do arquivo, podendo os demais terem sido perdidos.

Pelo fato dos dados armazenados precisarem ser decifrados pela primeira camada de criptografia, o resultado da solução do sistema linear precisa ser exato. Para resolver este problema foi utilizada aritmética modular e um campo de verificação dos dados em cada bloco, esta verificação é feita através do valor do hash do bloco.

4.8. Armazenamento

Quando um novo arquivo está para ser gravado no SEA é gerado um identificador único, chamado FID, utilizando o algoritmo SHA-1 [National Institute of Standards and Technology 2002] que recebe como semente o nome de usuário, sua senha, o nome do arquivo a ser gravado, o sal e o tempo atual no formato Unix Timestamp. O sal é uma

cadeia de bytes utilizados para diferenciar textos cifrados com a mesma chave em instalações diferentes do SEA, assim como utilizado no protocolo PKCS #5 [Kaliski 2000]. O FID é utilizado como semente para o gerador pseudo-aleatório.

A seqüência pseudo-aleatória é utilizada como endereço para a gravação dos blocos em um arquivo binário. Apesar da existência deste arquivo indicar a presença de dados escondidos, esta implementação é suficiente para o objetivo da pesquisa.

O usuário tem a possibilidade de criar uma árvore virtual de pastas e arquivos armazenados no volume esteganográfico. No sistema, esta árvore é armazenada em um arquivo chamado FAT, escrito no padrão XML. O arquivo contém a estrutura da árvore, os nomes e FIDs dos arquivos do usuário. A FAT também é armazenada no volume esteganográfico. Dessa forma, o usuário precisa somente fornecer o FID da FAT para ter acesso aos seus arquivos. Vale ressaltar que o usuário não é obrigado a utilizar a FAT, e também que este pode criar diversas FATs.

5. Testes

A segunda aplicação do SEA é a plataforma de testes. Os testes foram desenvolvidos para verificar ocorrência de colisões e assim determinar a capacidade de armazenamento do sistema. A plataforma de testes criada consiste em dois módulos separados. O primeiro deles, chamado Test é responsável pela execução dos testes. O segundo, chamado Stat, é responsável pela obtenção da média e desvio padrão de execuções dos testes. O módulo de Test realiza a gravação e recuperação de arquivos fictícios no volume esteganográfico e determina onde ocorrem as colisões.

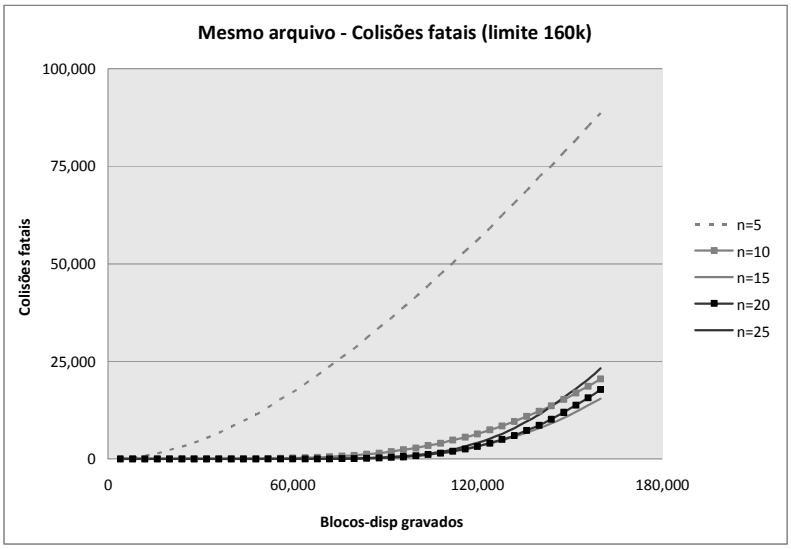

Figura 4. Número de colisões fatais entre blocos de um mesmo arquivo.

O primeiro teste realizado efetuou a gravação contínua de blocos, simulando um arquivo muito maior do que a capacidade de armazenamento sem que ocorram colisões fatais. Foi medida a quantidade de colisões fatais, que acarretam a perda do arquivo. Dessa forma foi possível detectar a eficiência da redundância dos dados. Foram utilizados 5 blocos de entrada para o algoritmo de dispersão, de forma que a curva para n=10 representa uma redundância de 2 vezes o arquivo. A Figura 4 apresenta o resultado para a gravação seqüencial de blocos até 160 mil blocos. Pode ser observado que

qualquer nível de redundância reduz significativamente a ocorrência de colisões fatais. Para um volume esteganográfico de 2 milhões de blocos, usado nos testes, não temos colisões fatais para arquivos de até 386 mil blocos. Esse tamanho corresponde a aproximadamente 19% do volume total do disco.

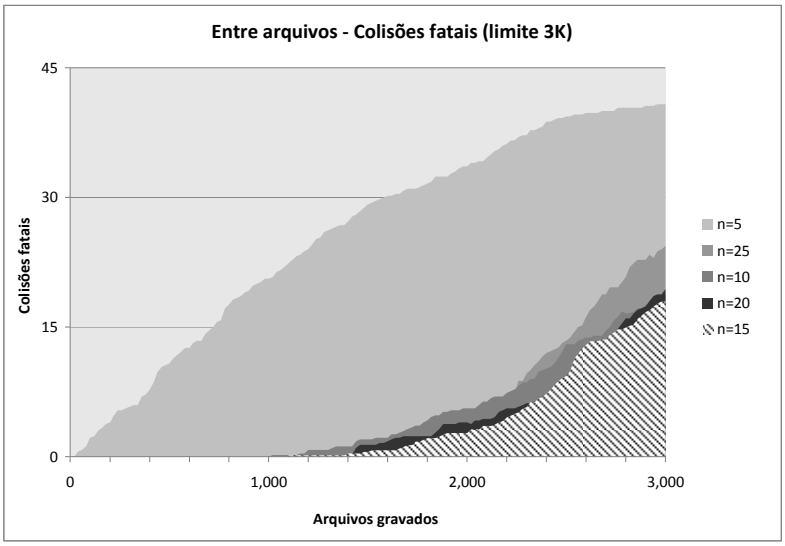

Figura 5. Número de colisões fatais entre arquivos.

O segundo teste mede a interação entre os arquivos. Observou-se que os resultados independem se os blocos gravados pertencem a arquivos diferentes ou a um mesmo arquivo (Figura 5). Neste caso o eixo das abscissas retrata o número de arquivos gravados, sendo todos de mesmo tamanho. Neste teste, o tempo de vida alcançou 338 mil blocos, ficando bem próximo do teste anterior.

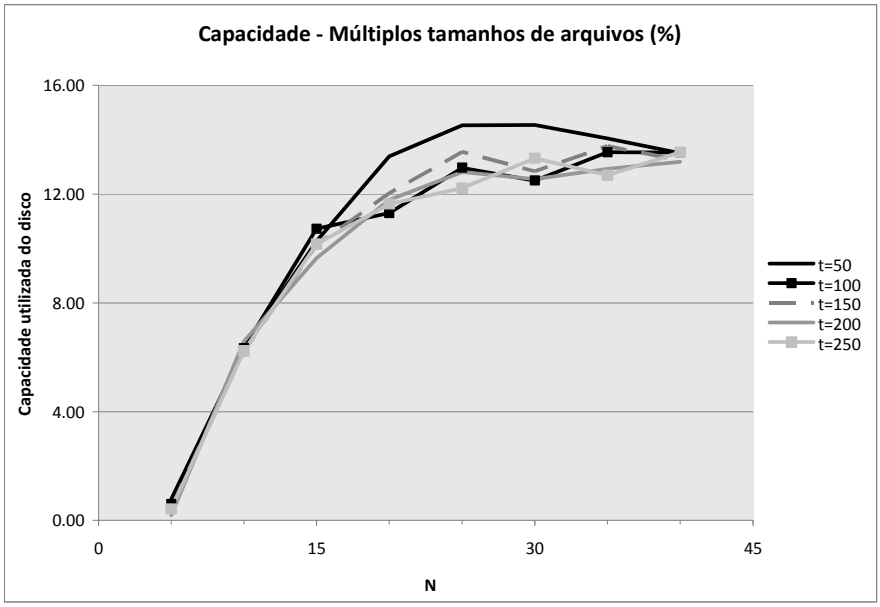

Figura 6. Capacidade relativa de armazenamento.

O terceiro teste tem como foco descobrir qual é a capacidade máxima do sistema, de forma a criar uma estimativa para o dimensionamento de sistemas reais. Foi

identificado que o número de colisões depende do valor de N e do número de blocos gravados, limitando a quantidade de arquivos íntegros no sistema até o momento que ocorre uma colisão fatal. No teste, para cada tamanho e N, foram gravados arquivos até que ocorresse pelo menos uma colisão fatal. Nesse ponto, a capacidade do sistema é considerada sua capacidade total, uma vez que uma colisão fatal foi encontrada. No sistema *Mnemosyne* [Hand e Roscoe 2002], por exemplo, a capacidade máxima alcançada gira em torno de 50%.

Podemos observar na Figura 6 que o pico de capacidade do sistema fica em torno de 14% do total de espaço do volume esteganográfico. Esses valores estão de acordo com os obtidos no segundo teste. Pode-se observar também que arquivos de menor tamanho otimizam o uso do volume.

6. Conclusão

O objetivo do desenvolvimento do SEA foi criar um conjunto de aplicações para verificar e avaliar o uso da esteganografia em sistema de arquivos. Através da aplicação principal, foi possível observar a usabilidade do sistema e entender os detalhes necessários para a aplicação da técnica.

Com a plataforma de testes, foi possível determinar a capacidade máxima do sistema e a eficiência da redundância na proteção dos dados. Também foi possível determinar a proporção ideal desta redundância que maximiza a capacidade.

Foram realizadas adaptações no algoritmo de dispersão de dados para se adequar ao SEA e foi proposta a inclusão de um passo adicional de criptografia, com o intuito de se reduzir a possibilidade de ataques ao sistema.

Referências

Anderson, R; Needhan, R; Shamir, Adi. (1998) “The Steganographic File System”, In: *IWIH: International Workshop on Information Hiding*.

Dusi, M.; Crotti, M.; Gringoli, F.; Salgarelli, L. (2008) “Detection of Encrypted Tunnels Across Network Boundaries” In: *ICC ‘08. IEEE International Conference on Communications 2008*, pages 1738–1744.

Hand, S.; Roscoe, T. (2002) “Mnemosyne: Peer-to-peer Steganographic Storage.” In: *Proceedings of the 1st International Workshop on Peer-to-peer Systems (IPTPS’02)*. Boston, USA.

Kaliski, B. (2000) "*PKCS #5: Password-Based Cryptography Specification Version 2.0*", RFC Editor.

National Institute of Standards and Technology. (2002) “Secure Hash Signature Standard”. In: *Federal Information Processing Standards Publication*.

Rabin, M. O. (1989) “Efficient dispersal of information for security, load balancing and fault tolerance”, In: *Journal of the ACM*, page 335.

Reeds, J. (1998) “Solved: The Ciphers in Book III of Trithemius’s Steganographia”. In: *Cryptologia*. United States Military Academy, West Point, USA, pages 291-317.