METODOLOGIAS DE ANÁLISE FORENSE PARA AMBIENTES BASEADOS EM NTFS

RESUMO

A definição de técnicas para a identificação e coleta de evidências digitais é essencial para uma possível ação judicial contra o autor de um ataque bem sucedido. Este fato, faz com que a criação de metodologias para identificar evidências no sistema de arquivos se torne um dos pontos mais importantes de uma análise forense computacional.

ABSTRACT

The techniques' definition for digital evidences identification and acquisition is essential for a possible prosecution against the author of a successful attack. This fact implies that the methodology's creation for evidences identification in a file system becomes one of the most importants point in forensics analysis.

1 - INTRODUÇÃO

Atualmente, a grande maioria das instituições conectadas à Internet, conta com algum aparato de segurança com o intuito de evitar que sua rede se torne alvo fácil para toda sorte de atacante e bisbilhoteiro que se prolifera pela Rede. Entretanto, aparatos como *firewalls* e VPN's são compostos por peças de software que contém inúmeras linhas de código, e que por sua vez, não estão imunes a erros de programação. Logo, mesmo que uma organização tome todos os cuidados para manter a segurança dos seus dados, não está totalmente livre da possibilidade de ser vítima de um ataque bem sucedido.

A definição de uma política a ser adotada no caso de um incidente de segurança é essencial para que os danos sejam minimizados. É necessário que haja metodologias para que uma vez descoberta a invasão, seja possível identificar, coletar e manipular evidências sem distorce-las. Mantendo-se assim, a possibilidade de futuramente adotar-se medidas legais contra o invasor.

Nesse artigo, serão discutidas algumas metodologias e problemas na aquisição de evidências relacionadas ao sistema de arquivos de plataformas Windows 2K/NT (NTFS). Sistema esse, amplamente utilizado, porém, carente de estudos com tal abordagem.

O objetivo é expor algumas técnicas e aspectos que devem ser considerados durante uma análise forense do sistema de arquivos NTFS, a fim de evitar a dependência de elementos de software e metodologias que não sejam livres.

1.1 - Ciência Forense

Ao abordar o termo forense, se é automaticamente remetido ao meio policial, onde na tentativa de solucionar um mistério, policiais e peritos devem analisar minuciosamente todo tipo de objetos, sinais e marcas que estejam presentes na cena do crime.

A análise forense inicia imediatamente após a chegada dos policiais ao local, começando pelo isolamento eficiente do perímetro, evitando assim, a exposição excessiva e possível contaminação das evidências. Passando então, para a fase de identificação e coleta de todo tipo de dado e material que possa ter alguma relevância na resolução do caso em questão. Apenas após a realização dessas duas primeiras etapas, inicia-se uma análise laboratorial das possíveis evidências, tais como: análise balística e de DNA. Tal fato contraria o que muitas pessoas pensam ao considerar apenas a análise laboratorial como análise forense.[4]

O sucesso da análise está então, ligado diretamente o sucesso de todas as três etapas que constituem o processo. Pois caso haja contaminação ou falhas na aplicação de metodologias para aquisição ou manipulação de evidências, tem-se por sua vez, uma grande possibilidade de não conseguir-se resultados precisos, ou mesmo, resultado algum durante uma análise laboratorial. Isto avaliando apenas aspectos técnicos, uma vez que falhas desse tipo invalidam completamente uma evidência em um tribunal.

As técnicas envolvidas na análise laboratorial de uma evidência são divididas em uma série de etapas que dependem diretamente do tipo de material que se está analisando, o que pode variar desde um cadáver a uma minúscula mancha de tinta. Tome-se por

exemplo a análise feita no DNA recolhido de uma amostra de sangue na cena de um crime. É possível aplicar exatamente o mesmo protocolo a toda amostra de DNA recebida: elimina-se as impurezas e o reduz à sua forma elementar [6]. Todos estes procedimentos devem ser padronizados, gerar resultados reproduzíveis e serem aceitos pela comunidade científica internacional.

Uma vez descrita a origem do termo forense, bem como exemplificada a utilização e as etapas do processo de análise em um âmbito de investigação geral, segue na seção 2, uma introdução à aplicação desta ciência no meio computacional.

2 - FORENSE COMPUTACIONAL

Com o advento do computador e o surgimento dos primeiros casos envolvendo o meio computacional, tornou-se necessária a criação de uma nova disciplina forense. Disciplina essa, que devia preocupar-se em atuar nesse novo nicho, criando metodologias e acumulando conhecimentos para a aquisição, manipulação e análise de evidências digitais.

A resolução de um mistério computacional pode ser uma tarefa árdua e difícil. É necessário que se examine o sistema minuciosamente, assim como um detetive examina a cena de um crime [1]. Para isso, a pessoa que está realizando a análise deve conhecer profundamente o sistema operacional em que está trabalhando podendo então, identificar e entender as relações de causa e efeito de todas as ações tomadas durante a análise.

Tal nível de conhecimento é essencial, devido a tamanha volatilidade de certos tipos de evidências. Tome-se por exemplo os tempos de acesso, criação e modificação de um arquivo, conhecidos como *MAC-Times*. A simples seleção de um arquivo no *explorer* do Windows altera o tempo do último acesso, anulando assim um dos mais poderosos mecanismos de se reconstituir o que ocorreu no sistema em um passado recente. Voltar-se-á a falar de *MACTimes* nas seções que se seguem, bem como dos problemas de sua utilização em ambientes Windows.

Uma vez entendidas as relações de causa e efeito dentro do sistema, existe ainda a necessidade de uma série de habilidades, para que um perito possa conduzir uma análise forense de maneira eficaz. Segundo Venama e Farmer [1], felizmente muitas dessas habilidades são características aos programadores, tais como: raciocínio lógico, possuir uma mente aberta e o entendimento das relações de causa e efeito, como dito anteriormente. Tais habilidades são largamente utilizadas durante a busca de um erro em um programa. Contudo a depuração de um programa ainda está distante do desafio representado por uma análise forense, já que ao se depurar um programa está lutando-se contra si mesmo, enquanto em uma análise forense enfrenta-se outro programador que não tem o interesse de ser descoberto [1].

2.1 - Metodologias no Tratamento de Evidências Digitais

Apesar do atual estágio das pesquisas no campo da forense computacional, ainda existe muita carência de metodologias para o manuseio deste tipo de evidência. Tal carência pode ser explicada pelo fato de existirem inúmeras mídias e sistemas operacionais, além de diversas mudanças de versão. Todos esses fatores tornam difícil a definição de padrões e metodologias, pelo menos da forma como acontece com as outras disciplinas forenses [7].

Atualmente já existem padrões definidos e sendo aplicados de forma experimental [10]. Eles foram desenvolvidos pelo SWGDE (*Scientific Working Group on Digital Evidence*), que é o representante norte-americano na *International Organization on Computer Evidence* (IOCE). Tais padrões foram apresentados durante a *International Hi-Tech Crime and Forensics Conference* (IHCFC), realizada em Londres, de 4 a 7 de outubro de 1999.

Os padrões desenvolvidos pelo SWGDE seguem um único princípio: o de que todas as organizações que lidam com a investigação forense devem manter um alto nível de qualidade a fim de assegurar a confiabilidade e a precisão das evidências. Esse nível de qualidade pode ser atingido através da elaboração de SOPs (*Standard Operating Procedures*), que devem conter os procedimentos para todo tipo de análise conhecida e prever a utilização de técnicas, equipamentos e materiais largamente aceitos na comunidade científica internacional [10].

Após a compreensão da complexidade envolvida no manuseio de evidencias digitais, pode-se então identificar alguns métodos para manipulação de evidências relacionadas ao sistema de arquivos, como proposto na seção 1.

2.2 - Manipulação de Sistemas de Arquivos

O desenvolvimento de documentos como os SOPs dependem de técnicas específicas para cada tipo de situação e sistema operacional (SO). No entanto, alguns procedimentos relacionados à manipulação de sistemas de arquivos são gerais e não dependem do SO que está sendo analisado. Tais procedimentos já são adotados pela comunidade forense internacional. Dentre eles cita-se:

- *Análise sobre cópias*: Ao se iniciar uma análise deve-se considerar veementemente a atuação a partir de cópias dos dados originais, mas não cópias comuns (*copy*). O ideal é que as cópias sejam feitas bit a bit (imagem). Dessa forma, copia-se todos os blocos de dados, inclusive os de arquivos apagados, sem a deturpação dos tempos de acesso. Cópias desse tipo podem ser feitas através de ferramentas semelhantes ao comando *dd* do UNIX. A seção 6, aborda algumas ferramentas que podem

ser úteis durante a análise de sistemas baseados em NT.

- *Assinaturas digitais*: Para que não paire dúvidas sobre a confiabilidade das imagens e conseqüentemente das evidências obtidas, deve-se garantir que os dados analisados são exatamente iguais aos dados originais recolhidos da "cena do crime". Uma maneira de assegurar essa confiabilidade é através de assinaturas digitais, invariavelmente MD5.

- *Sem permissão de escrita e execução*: Um fato importante a se atentar antes de começar uma análise, é o de se examinar a imagem sem permissão de escrita e até mesmo de execução, a fim de evitar a alteração ou a execução involuntaria de algum arquivo. Muitos sistemas também permitem desabilitar a atualização dos tempos de acesso, que pode eventualmente ser interessante.

- *MACTimes*: São potencialmente uma das mais poderosas formas de se reconstituir o que ocorreu em um sistema de arquivos no passado [2]. Esse termo é usado para referênciar os três atributos de tempo presentes em todos os arquivos e diretórios da maioria dos SO's (mtime, atime e ctime). No caso do NT esses atributos são chamados de *LastWriteTime, LastAccessTime e CreationTime.* A análise desses atributos envolvem a utilização de ferramentas especiais que utilizem chamadas de sistema diretamente ou contornem de alguma forma os métodos convencionais de acesso a arquivos, com o intuito de evitar a perda de informações. Um problema apresentado por este tipo de evidência é a sua volatilidade e sua pouca utilidade em ambientes de grande atividade [2].

- *Arquivos excluídos*: Um mecanismo eficiente de recuperação de arquivos excluídos também pode ser decisivo durante uma análise forense, uma vez que na tentativa de esconder seus rastros, os invasores podem apagar arquivos que possam denunciar sua presença. O problema é que existem inúmeras ferramentas para se efetuar uma exclusão de forma segura: são as chamadas ferramentas de *wipping*.

3 - FORENSE EM AMBIENTE NT

A análise forense de sistemas baseados em Windows NT apresenta-se como um grande desafio para forense computacional, pois trata-se de um sistema operacional fechado, de documentação escassa e controversa, mas largamente utilizado.

Atualmente sabe-se que o ideal é que a análise seja feita em um ambiente controlado, isto é, isolado e com ferramentas confiáveis. De preferência utilizando um outro sistema operacional para que se evitem vícios comuns à ambientes Windows, tal como o principio de tornar tudo o mais fácil possível para o usuário. Todavia, o NTFS contém muitas peculiaridades não suportada por *drivers* de outros sistemas, entre eles o Linux.

Um problema a ser resolvido, é a não existência, no ambiente NT, de estudos sobre as ações tomadas por seus atacantes, como acontece no UNIX. Um exemplo desse tipo de documentação é o projeto honeynet [13], que conta com diversas descrições de incidentes, bem como a descrição das ações tomadas durante suas investigações [13].

Sistemas UNIX também contam com diversas ferramentas de código aberto, o que facilita o entendimento das metodologias adotadas. Dentre elas podemos citar o *The Coroners ToolKit* (TCT), conjunto de ferramentas reunidas por Dan Farmer e Wietse Venema que já está se tornando um padrão de fato na análise forense desse sistema [14].

4 - ESTRUTURA DO NTFS

O NTFS (*New Technology File System*) é o sistema de arquivos nativo do Windows NT/2K, embora mantenham suporte ao sistema FAT originário do DOS. Ele foi desenvolvido com o objetivo de suprir as necessidades do mercado coorporativo, tais como: maior capacidade de endereçamento, suporte a critérios de segurança aplicáveis a cada arquivo individualmente, cifragem de dados entre outros.

A formatação de uma partição NTFS resulta na criação da *Master File Table* (MFT) e de diversos arquivos de sistema. A MFT contém informações sobre todos os arquivos e diretórios de uma partição NTFS. A figura 1 ilustra o posicionamento padrão da MFT.

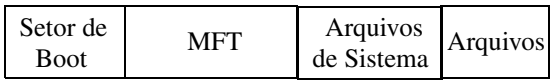

Figura 1: Estrutura de um Volume NTFS

Além da MFT, o processo de formatação ainda cria um conjunto de arquivos que contém meta informações usadas para implementar a estrutura do sistema de arquivos. Tais arquivos são mapeados nos primeiros registros da MFT, inclusive a própria. A tabela 1 mostra tais arquivos e seu mapeamento junto a MFT.

File Name	MFT Record	Descrição
$MFT	0	Master File Table
$MftMirr	1	Cópia dos 16 primeiros registros da MFT
$LogFile	2	Arquivo de log das trasações efetuadas no disco
$Volume	3	Número de série do volume, data de criação e o *dirty flag*
$AttrDef	4	Definição dos atributos
$.	5	Diretório raiz do volume
$Bitmap	6	Representação do disco indicando que clusters estão sendo utilizados
$Boot	7	Setor de *boot* do volume
$BadClus	8	*Clusters* defeituosos
$Secure	9	Contém *security descriptors* únicos para todos os arquivos do volume
$Upcase	10	Mapeia caracteres minúsculos em seus correspondentes maiúsculos
$Extend	11	Usado por várias extensões opcionais, como quotas e reparse points
	12-15	Reservados para uso futuro.

Tabela 1: Arquivos de Meta informação

Até o NT 4.0 tais arquivos podiam ser vistos através do comando *dir*, como no exemplo 1.

```
C:\> dir /ah <nome do arquivo>
```

Exemplo 1: Visulização de Arquivos de Meta informação no NT 4.0.

4.1 - Master File Table

Como dito anteriormente a MFT contém informações sobre todos os arquivos e diretórios do volume. Cada registro é composto por um pequeno cabeçalho que contém informações básicas descrevendo o próprio registro. tais informações são listadas abaixo:

- Números de seqüência, usados para verificação de integridade;
- Ponteiro para o primeiro atributo do registro;
- Ponteiro para o primeiro byte livre no registro;
- Número do registro em relação ao registro base da MFT, caso não seja o primeiro.

O cabeçalho inicial é seguido por um ou mais atributos que descrevem as características do arquivo. A figura 2 ilustra a estrutura de registro padrão.

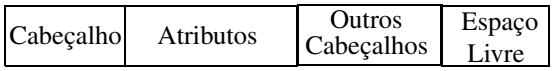

Figura 2: Registro da MFT

Cada atributo é dividido em dois componentes: um cabeçalho que guarda o tipo do atributo, nome, *flags* e a localização da parte de dados do registro, e uma parte de dados onde é armazenada a informação do registro. Existem uma série de possibilidades para o armazenamento de dados nos registros da MFT, mas tais detalhes estão fora do escopo desse artigo. Na figura 3 tem-se um exemplo de registro, onde pode-se observar seus primeiros atributos, dentre os quais, destaca-se o atributo *Standard Info*, que armazena os MACTimes. Na seção seguinte tem-se uma discussão da análise desse tipo de evidência.

4.2 - MACTimes

Uma vez compreendida a estrutura básica de um volume NTFS, pode-se então partir para uma análise mais detalhada a cerca dos *MACTimes* do ambiente Windows.

Um primeiro problema, que aflige a grande maioria dos SO's é a falta de um registro histórico, uma vez que os tempos registrados nos arquivos dizem respeito apenas a última modificação o que torna impossível a obtenção dos acessos anteriores utilizando-se apenas o sistema de arquivos [2]. Uma solução para esse problema poderia ser a habilitação de um log que registra-se os acessos aos arquivos críticos do sistema, o que além de fornecer um histórico dos acessos ainda forneceria outros dados como o usuário que fez o acesso. Mas, tal solução pode ter efeitos colaterais: como o possível excesso de *logs*, além do fato de que seria necessário um estudo para se descobrir quais arquivos monitorar.

Outro problema é que para efeitos de desempenho, o *LastAccessTime* tem resolução de uma hora, logo, todos os acessos a um arquivo em um intervalo menor de tempo aparentemente não é registrado. No entanto, existem relatos de que é possível se verificar

Header	Name	Arquivo.txt	Standard Info	January 3, 2000 8:30 pm January 3, 2000 8:46 pm December 9, 1999 5:03 pm	Attribute List Data	Data (non-resident)

Figura 3: Exemplo de Registro da MFT

tempos de acesso com resolução de segundos, entretanto, tal possibilidade foi anunciada por membros de uma empresa da área que não divulgaram os detalhes da operação.

A análise de *MACTimes* no Windows ainda reserva uma série de anomalias, tais como:

- Quando copia-se um arquivo para um outro de nome diferente, a data da última modificação continua igual a do arquivo original, enquanto as datas do último acesso e criação se comportam normalmente, dando a impressão que o arquivo foi modificado antes de ser criado;

- Recentemente na lista de discussão sobre forense da Security Focus [15], foi descrito o seguinte problema: ao se copiar um arquivo para um outro com o mesmo nome de um recém excluído, a data de criação do novo arquivo assume a do arquivo antigo, excluído anteriormente. Neste caso, provavelmente trata-se de um erro de programação e não de projeto. Entretanto, tal anomalia poderia causar deturpações em um processo de análise.

5 - ALTERNATE STREAMS

As *alternate streams* são sem sombra de dúvida um dos pontos cruciais da análise de um sistema NTFS. Elas foram tratadas como falhas de segurança primeiramente pelos colunistas da InfoWorld Security, Stuart McClure e Joel Scambray em julho de 98. Em linhas gerais, trata-se de um mecanismo para embutir um arquivo dentro de outro, sem que seu conteúdo ou tamanho seja alterado. Tecnicamente falando, podemos dizer que todo arquivo NTFS possui um outro arquivo sem nome embutido, a este dase o nome de *default stream* ou *unnamed stream*, onde os dados convencionais, como texto e programas, são armazenados. Todavia, existe a possibilidade de criar-se arquivos embutidos com nomes diferentes, são as chamadas *alternate streams* [9] [3].

No início dos anos 90 a Microsoft introduziu essa funcionalidade com o intuito de tornar o NT um servidor de arquivos para computadores que utilizassem o Mac OS. O objetivo era simular o *resource forks* do HFS (*Hierarchical File System*) usado para armazenar dados como ícones e outros tipos de metainformação.

O Windows utiliza arquivos *alternate streams* para armazenar informações fornecidas na *summary tab* do explorer, como mostrado na figura 4. Tais informações são armazenadas em uma stream chamada ?SummaryInformation, onde o ponto de interrogação indica um caracter especial que não pode ser impresso.

Figura 4: Summary Tab

5.1 - Acesso ao Conteúdo

O acesso aos arquivos embutidos pode ser feito através da notação do exemplo 2. Entretanto, poucos comandos podem utilizar tal notação, entre ele podemos destacar o *more* cuja utilização é demonstrada no exemplo 3.

```
-> <nome_arquivo>:<nome_stream>

-> leiame.txt:cmd
```

Exemplo 2: Notação.

Com a execução do comando *echo* cria-se um arquivo embutido chamado *otr* e com o *more* podemos verificar seu conteúdo. Note que o acesso é feito

```
C:\> echo Testando > arq.txt:otr

C:\> more < arq.txt:otr
```

Exemplo 3: Acesso.

através de redirecionamento, uma maneira de contornar a falta de suporte dos comandos.

Uma vez demonstrada o acesso a um arquivo embutido, uma pergunta é quase imediata: "É possível embutir executáveis?". A resposta a essa pergunta é sim, como pode ser visto no exemplo 4.

```
C:\>type nc.exe > otk.exe:a.exe
```

Exemplo 4: Embutindo Executáveis

Um programa localizado em uma *alternate stream* pode ser executado de 5 formas diferentes no Windows 2000 [3]:

- Através da opção *run* do menu *start* do Windows utilizando-se a notação do exemplo 5.

```
-> file:\\<dir>\<arq>:<stream>

-> file:\\c:\otk.exe:a.exe
```

Exemplo 5: Execução Através do Run

- Com a utilização de atalhos, inclusive na pasta *startup*.

- Inclusão de uma chave na pasta do registro que gerencia os programas executados automaticamente durante o *logon*.

- Para *vb scripts* existe uma possibilidade extra, através comando wscript (exemplo 6).

```
-> wscript <arquivo>:<stream>

C:\> wscript otk.exe:s.vbs
```

Exemplo 6: Execução Através do Wscript

- E por fim, através de um programa criado com esse objetivo. (exemplo 7)

5.2 - Perigos e Possibilidades

Em agosto de 2000, dois *hackers* tchecos criaram o W2K.Stream [10], um vírus que utilizava os arquivos embutidos do Windows 2000. Apesar de não causar maiores danos à máquina infectada o W2K chamou novamente a atenção para o problema.

No meio forense o perigo está na possibilidade de se ocultar um programa malicioso como o *netcat* (veja seção 6). Dessa forma, tem-se *backdoor* que não poderia ser localizada no disco através das ferramentas convencionais.

Apesar de tratar-se de uma funcionalidade bem conhecida, arquivos embutidos não criam alterações nas assinatura gerada pela maioria dos elementos de software para assinatura digital. Sendo assim arquivos com *alternate streams* só podem ser descobertos através da utilização de programas específicos, como o streams do Mark Russinovich citado na seção *6.*

6 - FERRAMENTAS ÚTEIS

Nesta seção estão relacionadas algumas ferramentas que podem ser úteis ao se efetuar uma análise. Todas são grátis, mas nem todas de código aberto. Seguem-se:

- CygWin Tools: Versão para Windows de diversas ferramentas GNU, inclusive o *dd*. Isto é possível, graças às bibliotecas cygwin que simulam as chamadas de sistema e geram o ambiente que tais ferramentas necessitam. Endereço: http://sources.redhat.com/cygwin/;

- MS-DiskEdit: Ferramenta acidentalmente inclusa no SP4 do Windows NT 4.0. Aparentemente era uma ferramenta de depuração interna, usada pelos desenvolvedores do NTFS. O importante é que ela pode acessar o disco em baixo nível (hexadecimal); e entende a estrutura da MFT, interpretando e exibindo o conteúdo dos registros de forma organizada e legível. O DiskEdit pode ser obtido através do seguinte endereço: http://www.informatik.fh-hamburg.de/pub/nt-service/sp4en-ex/i386/diskedit.exe; [8]

- MD5summer: Um exemplo dos inúmeros elementos de software para criação de assinaturas digitais. Uma das suas vantagens é a compatibilidade com o *md5sum* do Linux, em relação ao formato do arquivo de saída. Endereço: http://homepages.ihug.co.nz/~floydian/md5/md5v11.zip;

- NetCat: Versão para NT do *netcat* original do UNIX. Muito usado, como *backdoor,* pelos atacantes. Endereço: http://www.l0pht.com/users/10pht/nc11nt.zip;

- Streams: Programa que detecta a existência de *alternate streams* em um arquivo. Endereço: http://www.sysinternals.com/files/streams.zip;

```
#include <windows.h>
#include <stdio.h>

void main( )
{
 HANDLE hFile, hStream;
 DWORD dwRet;

 hFile = CreateFile("testfile",
                    GENERIC_WRITE,
                    FILE_SHARE_WRITE,
                    NULL,
                    OPEN_ALWAYS,
                    0,
                    NULL );
 if( hFile == INVALID_HANDLE_VALUE )
     printf( "Cannot open testfile\n" );
 else
     WriteFile( hFile, "This is testfile", 16, &dwRet, NULL );

 hStream = CreateFile("testfile:stream",
                      GENERIC_WRITE,
                      FILE_SHARE_WRITE,
                      NULL,
                      OPEN_ALWAYS,
                      0,
                      NULL );
 if( hStream == INVALID_HANDLE_VALUE )
     printf( "Cannot open testfile:stream\n" );
 else
     WriteFile(hStream, "This is testfile:stream", 23, &dwRet, NULL);
}
```

Exemplo 7: Exemplo de Código para Acessar Alternate Streams [5]

- Strings: Recupera todas as *strings* contidas em um arquivo executável. Muito útil durante a identificação da função de um dado software. Endereço: http://www.sysinternals.com/files/strings.zip;

7 - CONCLUSÕES

A análise forense de um sistema proprietário como o Windows NT/2000, torna difícil a definição de metodologias totalmente confiáveis. Uma vez que, não se tem a exata noção das conseqüências de ações tomadas durante a análise. Torna-se então, indispensável a troca de informações e experiências a fim de se conseguir um conjunto de metodologias amplamente aceitas. Dessa forma, não se corre o risco da dependência de metodologias fechadas e elementos de software privados.

8 - REFERÊNCIAS BIBLIOGRÁFICAS

[1] FARMER, Dan; VENEMA, Wietse; *Forensic Computer Analysis: An Introduction*; Dr. Dobb's Journal; setembro 2000; http://www.ddj.com/articles/2000/0009/0009f/0009f.htm

[2] FARMER, Dan; *What are MACTimes?*; Dr Dobb's Journal; outubro 2000;

[3] KARSPERSKY, Eugene; ZENKIN, Denis; *NTFS Alternate Data Streams*; Windows 2000 Magazine; Storage Admin; spring 2001; http://www.storageadmin.com/Articles/Index.cfm?ArticleID=19878

[4] LEE, Henry; PALMBACH, Timothy; MILLER, Marilyn; *Henry Lee's Crime Scene Handbook*; Academic Press;

[5] MICROSOFT, Corp; *Windows 2000 Server Resource Kit*; TechNet Julho 2000

[6] NOBLETT, Michael G.; POLLITT, Mark M.; PRESLEY, Lawrence A.; *Recovering and Examining Computer Forensic Evidence*; Forense

Science Communications, outubro 2000, Vol. 2 N. 4; Federal Bureau of Investigation;

[7] OLIVEIRA, Flávio; GUIMARÃES, Célio; REIS, Marcelo; GEUS, Paulo; *Forense Computacional: Aspectos Legais e Padronização*; Anais do Wseg'2001 (I Workshop de Segurança em Informática); Florianópolis - Brasil;

[8] RUSSINOVICH, Mark; *Exploring NTFS On-disk Structures*; Windows 2000 Magazine; Focus; november 2000;

[9] RUSSINOVICH, Mark; *Inside Win2K NTFS, Part 2*; Windows 2000 Magazine; Focus; winter 2000; http://www.win2000mag.com/Articles/Index.cfm?ArticleID=15900

[10] SMIT, Thomas A.; *An In-depth Look at W32.Winux and W2K.Stream and the Ever Growing "Proof of Concept" Virus*; http://www.sans.org/infosecFAQ/malicious/w2kwinux.htm

[11] SWGDE, Scientific Working Group on Digital Evidence; IOCE, International Organization on Digital Evidence; *Digital Evidence: Standards and Priciples*; Forense Science Communications, abril 2000, Vol. 2 N. 2; Federal Bureau of Investigation;

[12] VENEMA, Wietse; *File Recovery Techniques*; Dr. Dobb's Journal; dezembro 2000; http://www.ddj.com/articles/2000/0012/0012h/0012h.htm

[13] Projeto Honeynet; http://project.honeynet.org/

[14] The Coroner's Toolkit; http://www.porcupine.org/forensics/tct.html

[15] Lista de discussão sobre forense; http://www.securityfocus.com/forums/forensics/intro.html

[16] Página criada pelos desenvolvedores do W2K.Stream; http://viry.bonusweb.cz/kniha_o_virech/ntfs.html