UM FRAMEWORK PARA PREPARAÇÃO DE REDES WINDOWS 2000 PARA FUTURAS ANÁLISES FORENSE

Flávio de Souza Oliveira
Instituto de Computação
Universidade Estadual de Campinas
13083-970 Campinas - SP
flavio.oliveira@ic.unicamp.br

Célio Cardoso Guimarães
Instituto de Computação
Universidade Estadual de Campinas
13083-970 Campinas - SP
celio@ic.unicamp.br

Paulo Lício de Geus
Instituto de Computação
Universidade Estadual de Campinas
13083-970 Campinas - SP
paulo@ic.unicamp.br

RESUMO

Com o intuito de automatizar a implantação de técnicas para ampliar os traços deixados pela utilização ilegal de recursos em uma rede Windows 2000 e conseqüentemente fazer com que sejam gerados um número maior de evidências que possam vir a solucionar um incidente de segurança, desenvolvemos o PFSAF que busca fornecer um meio escalável para a preparação de redes W2k para futuras análises forense.

ABSTRACT

*This paper presents the PFSAF (*Pre-Forensic Setup Automation Framework*). PFSAF intends to automate the implementation of techniques to amplify tracks of illegal use in Windows 2000 networks and consequently make them to produce a greater number of evidences that could help to solve a security incident. The PFSAF main goal is supply a scalable environment to make Windows 2000 networks more forensic friendly.*

1 - INTRODUÇÃO

O estabelecimento de programas de resposta a incidente por parte das organizações conectadas à Internet, pode hoje ser considerado vital para a minimização dos prejuízos financeiros no caso de um ataque bem sucedido. Entretanto, ainda são poucas as instituições que contam com programas deste tipo devido aos custos e dificuldades técnicas de sua implantação.[19]

A preparação para uma resposta consistente começa bem antes do incidente propriamente dito, através da elaboração de procedimentos e políticas, definição de responsabilidades e treinamento de pessoal. Os procedimentos adotados durante um eventual incidente devem ser testados e ensaiados com antecedência em virtude da grande pressão envolvida durante este tipo de evento, o que possibilita a ocorrência de erros que podem inutilizar evidências e prejudicar o andamento da investigação.

Uma importante etapa da elaboração de um programa de resposta é a correta configuração do parque de máquinas, preparando-as para uma eventual análise forense durante a execução dos procedimentos previamente elaborados. Esta preocupação com configuração das máquinas se justifica principalmente nas plataformas Windows, onde as configurações *default* deixam escaços rastros das atividades dos usuários, o que pode retardar a obtenção de resultados e conseqüentemente atrasar o retorno da organização às suas atividades normais, agravando assim os prejuízos financeiros sofridos.

Quando uma máquina Windows é previamente configurada como possível alvo de uma análise forense, a busca por evidências é facilitada devido à utilização correta dos recursos que viabilizam o levantamento dos fatos ocorridos no passado próximo da máquina vítima [5]. Entretanto, a configuração de grandes redes com um sistema "*hands-on*" como o Windows 2000, que possui poucos mecanismos para automatização de tarefas pode ser uma tarefa extremamente laboriosa.

Com o intuito de automatizar algumas tarefas administrativas que podem influir significativamente para a agilidade de uma possível análise forense e fornecer mecanismos para ampliar traços da utilização ilícita de recursos computacionais em uma máquina Windows 2000, desenvolvemos PFSAF (*Pre-Forensic Setup Automation Framework*), sofware GPL (*GNU General Public License*) que procura viabilizar a preparação remota de máquinas W2k para futuras análises durante procedimentos de resposta a incidentes.

Este artigo tem como objetivo apresentar e discutir aspectos relevantes à configuração de máquinas W2k para uma futura análise forense e também apresentar este novo fremework que busca automatizar tarefas relativas a esta preparação, o que conseqüentemente acaba por evidenciar a importância da implantação de programas de resposta a incidente.

1.1 - Trabalhos Relacionados

Existem diversos trabalhos realizados na área de administração de grandes[1] redes Windows, contudo são escaços os que enfocam as necessidades da forense computacional. Além disso, a maioria dos softwares que objetivam a automatização de tarefas administrativas em redes Windows NT/2000 são fer-

1. Redes com algumas centenas de máquinas.

ramentas comerciais, com código fechado e cujos custos estão fora da realidade de inúmeras instituições e times de resposta a incidente.

Apesar de haver poucas aplicações que supram as necessidades que serão relatadas neste artigo, existem dois trabalhos que possuem semelhanças com o nosso *framework*. O primeiro é o conjunto de scripts apresentado por Harlan Carvey em [4], este poderia ser considerado o embrião do PFSAF, pois apresenta várias idéias em comum, dentre elas a consulta remota aos atalhos contidos na pasta Startup. Entretanto, os scripts apresentados em [4] dificilmente poderiam ser aplicados em uma rede real, uma vez que não são escaláveis para redes de grande porte. Obrigando o administrador a alterar e executar cada script individualmente para cada máquina, o que restringe sua utilização.

Outro trabalho que apresenta semelhanças é o DoIt4Me apresentado por Alessandro Augusto em [1]. Trata-se de um aplicativo voltado a automatização de tarefas administrativas em grandes redes Windows NT, contudo, suas funcionalidades não cobrem todas as necessidades aqui apresentadas (e.g. integridade de arquivos), além de ter seu foco voltado para uma outra plataforma.

O Tripwire, aplicação para verificação de integridade de arquivos, também pode ser considerado um trabalho relacionado. Embora sua versão para Windows seja proprietária, existe um projeto *open source* para a plataforma Linux [20].

2 - RESPOSTA A INCIDENTES

Em virtude do seu crescimento, atualmente pode-se afirmar que a preocupação com segurança é requisito essencial para a maioria das aplicações em rede. Um bom paralelo foi feito por Jeffrey J. Carpenter, engenheiro de segurança senior do CERT/CC (*Computer Emergency Response Team/Cordination Center*): "A história da segurança na Internet pode ser comparada à vida em uma cidade. Quando a cidade é pequena, as pessoas se conhecem e confiam umas nas outras, de modo que janelas e portas podem ser deixadas abertas (...). Contudo, quando a cidade cresce, crimes e segurança tornam-se preocupações mais comuns. A Internet pode hoje então ser comparada a uma metrópole, onde as portas e janelas devem permanecer fechadas a maioria do tempo"[3].

O problema é que mesmo tomando-se todas as medidas necessárias, falhas de segurança podem ocorrer, uma vez que alguma vulnerabilidade ainda não divulgada pode ser explorada ou um novo tipo de ataque pode ser utilizado. Dessa forma, não há como afirmar que um dado aparato de segurança (*Firewalls*, VPNs...) está isento de falhas. Isto se deve principalmente ao fato de que tais aparatos, bem como os serviços oferecidos através da Internet, são compostos por inúmeras peças de software, que por sua vez, possuem milhares de linhas de código que não estão imunes a erros de programação.

Tendo em vista que não há esquema de segurança imune a falhas, torna-se então necessária a definição de procedimentos a serem adotados no caso de um ataque bem sucedido, além da presença de pessoal capaz de executar tal função (Time de Resposta). No entanto, a preocupação com tal metodologia ainda é muito pequena dentro das organizações conectadas à Rede, o que pode agravar em muito os prováveis prejuízos, no caso de eventos como este.

3 - FORENSE COMPUTACIONAL

Ciência forense é a ciência exercida em favor da lei para uma justa resolução de um conflito [18]. Em outras palavras ciência forense seria, em sua origem, aquela que se baseia em procedimentos científicos para a obtenção de informações que possam ser úteis durante uma disputa judicial. Este termo esta relacionado ao meio policial e até bem pouco tempo atrás, não tinha qualquer relação com o meio computacional. Entretanto, o aumento do uso cotidiano do computador e da Internet, por parte das empresas e usuários domésticos, fez com que também surgissem crimes que explorassem esse novo tipo de corportamento. Crimes estes praticados por criminosos que apenas aprenderam utilizar uma nova ferramenta, ou, em sua maioria, por pessoas de conduta íntegra no mundo real, mas que buscam se beneficiar do virtual anonimato conferido pela Rede para praticar atos ilícitos.

Para que as agências legais (polícia, órgãos de defesa) pudessem lidar com este novo tipo de crime e ajudar a justiça a condenar este tipo de criminoso, criou-se a forense computacional. Segundo [13], a forense computacional pode ser definida como sendo a ciência de adquirir, preservar, recuperar e exibir dados que foram eletronicamente processados e armazenados digitalmente.

Assim como ocorre nas outras disciplinas forense o processo de análise no meio computacional é metódico e deve seguir procedimentos previamente testados e aceitos pela comunidade científica, de forma que todos os resultados obtidos durante uma análise sejam passíveis de reprodução.

4 - ANÁLISE FORENSE DE MÁQUINAS W2K

Para o melhor entendimento da importância de se efetuar configurações que considerem a necessidade de uma futura análise forense, é importante o entendimento básico de como se procede uma investigação em uma máquina W2k. Contudo, nesta seção são citados apenas alguns exemplos de procedimentos que podem, ou não, ser adotados em uma investigação, uma vez que cada caso possui suas necessidades peculiares, o que inviabiliza a definição de procedi-

mentos aplicáveis a todo universo de situações que podem ser encontradas durante um incidente de segurança.

4.1 - Live Analysis *no W2k*

Pode-se definir a live analysis como sendo aquela efetuada em um sistema vítima de algum incidente de segurança, sem que anteriormente tenha sido executado qualquer procedimento para seu desligamento. Esse tipo de análise é extremamente importante para a investigação, uma vez que é a única oportunidade de se coletar uma série de informações voláteis, que não estarão mais disponíveis quando a máquina reiniciar, tais como as conexões de rede atualmente ativas e os programas que estão sendo executados no momento.

O grande problema deste tipo de análise é a falta de domínio sobre a máquina analisada, uma vez que esta pode ainda estar sobre a ação do atacante, contendo programas e bibliotecas desenvolvidas para ocultar informações e ludibriar o investigador. Por este motivo, é necessário que a análise seja efetuada a partir de programas e bibliotecas originários de uma mídia confiável (eg. CD ROM), para que assim o examinador possa ter garantia da integridade dos binários que está utilizando.

Um possível problema com o qual o investigador irá se deparar durante a seleção das ferramentas que constituirão o seu CD de aplicativos (Kit de Resposta), é a necessidade de descobrir as dependências de bibliotecas de seus programas. Uma maneira de descobrir quais DLLs determinado programa necessita para ser executado é através da utilização do *Dependecy Walker* (`depends.exe`). Esta ferramenta vem com o Resource Kit do W2k e analisa toda a árvore de dependências de determinado software, podendo exibir inclusive quais funções são utilizadas e exportadas em cada biblioteca.

Outra alternativa é o `listdlls.exe`[2], desenvolvido por Mark Russinovich, esta ferramenta é capaz de listar todas as bibliotecas que estão sendo utilizadas por determinado processo. Desta forma, é necessário executar a ferramenta a ser analisada para posteriormente utilizar o `listdlls.exe` para obter-se a listagem de DLLs necessárias. O `listdlls` também pode ser útil durante a *live analysis*, na qual pode ser utilizado para investigar algum processo suspeito em uma máquina invadida.

A estratégia de incluir todas as bibliotecas necessárias à execução dos programas que serão utilizados durante a *live analysis* no Kit de Resposta, pode minimizar a utilização de código originário da máquina vítima, contudo apenas essa medida não é suficiente para assegurar que nenhuma DLL do sistema suspeito será consultada. Isto porque quando a máquina é abordada, ela está em funcionamento, logo, já possui diversas DLLs carregadas em sua memória. Caso uma das ferramentas a serem utilizadas durante a análise necessite de uma biblioteca que já esteja na memória, ela não será carregada novamente, fazendo com que a DLL presente no CD seja ignorada e abrindo uma brecha para a produção de falsos resultados.[2]

A solução mais apropriada para se evitar o acesso a bibliotecas dinâmicas inseguras é eliminar todo e qualquer acesso dinâmico através da compilação estáticas de todas as ferramentas necessárias para a análise. No entanto, em virtude da cultura comercial da família Windows, poucas ferramentas desenvolvidas para este sistema possuem código aberto, inviabilizando a re-compilação da maioria dos programas.

Outra solução que poderia ser adotada seria a limpeza de todas as DLLs presentes na memória durante a análise, contudo a quantidade de distúrbios que esta ação pode causar é imprevisível, uma vez que o W2k não apresenta mecanismos para efetuar esta operação de forma confiável. Tal medida pode causar inúmeras GPFs (*General Protection Fault*) nos programas em execução na máquina, podendo assim prejudicar em muito o andamento da análise forense, uma vez que um dos princípios básicos deste tipo de procedimento segundo [5], é justamente deturpar o sistema analisado o mínimo possível.

A *live analysis* do Windows 2000 ainda representa um grande problema para a forense computacional, em virtude dos problemas citados.

4.2 - Bases da Análise Forense de Máquinas W2k

Muitos dos principais passos de uma análise forense computacional podem ser portados para vários sistemas operacionais e o W2k não é diferente. Diversos procedimentos adotados no presente sistema são provenientes de outras plataformas, principalmente do mundo UNIX.

Após a execução de uma *live analysis* minuciosa, pode ser necessário o desligamento da máquina vítima para uma análise *postmortem*, que pode ser descrita como sendo uma análise efetuada em um ambiente controlado, onde possam ser executados procedimentos que atuam e coletam informações da máquina em mais baixo nível. Alguns exemplos deste tipo de análise são:

- **File Slack**: Os sistemas operacionais da Microsoft armazenam seus arquivos em disco utilizando blocos de dados de tamanho fixo chamados *clusters* (Figura 1), contudo os arquivos em um disco podem ter os mais variados tamanhos, dependendo do seu conteúdo. Desta forma, raramente o tamanho de um arquivo é múltiplo do tamanho de um *cluster*, o que impede o armazenamento ideal. Sendo assim, é comum que o último *cluster* associado a um arquivo não seja totalmente utilizado por ele (Figura 1), permitindo que dados excluídos

2. http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml

deste e de antigos arquivos possam ser capturados e analisados.

- **RAM Slack**: Além de dados de antigos arquivos do disco, o *file slack space* também pode conter conjuntos de bytes aleatoriamente selecionados da memória RAM. Isto ocorre porque o Windows normalmente efetua escritas no disco em blocos de 512 bytes, chamados setores. Como normalmente a quantidade de informação a ser gravada não pode ser dividida igualmente em blocos de 512, geralmente é necessário que o último bloco de informação tenha que ser completado de alguma forma (Figura 1). Neste caso, o Windows faz o complemento com os buffers de memória do sistema operacional.

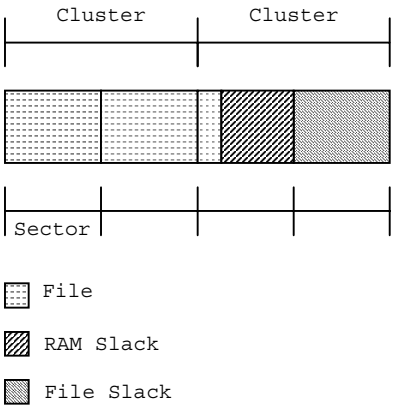

Figura 1: *Slack Space*

- **Alternate Streams**: Tecnicamente falando, podemos dizer que todo arquivo NTFS possui um outro arquivo sem nome embutido, chamado *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 os chamados *alternate streams*. O problema é que a detecção destes arquivos embutidos que possuem nomes, não pode ser efetuada por nenhum dos programas nativos do W2k, fazendo com que eles possam ser facilmente utilizados para ocultação de programas e outras informações. [15]

A análise *postmortem* pode ser extremamente demorada e trabalhosa. Ela deve ser conduzida a partir de cópias bit-a-bit dos discos da máquina vítima para que não haja o risco de que algum erro do examinador venha a comprometer a integridade dos arquivos originais, causando um dano irreparável ao andamento das investigações. O ambiente controlado citado anteriormente poderia ser, por exemplo, uma máquina com mais de um sistema operacional contendo ferramentas próprias para este tipo de análise, tais como o *Foundstone Forensic ToolKit*[3].

Através da utilização de uma máquina que possua outro sistema operacional com suporte ao sistema de arquivos NFS, tal como o Linux, pode-se aproveitar poderosas ferramentas desenvolvidas para o mundo Unix, como o TCT[4] (*The Coroners ToolKit*). Além disso, é possível evitar qualquer tipo de alteração, naturalmente efetuada pelo W2k, nos arquivos de índice da partição durante o boot, o que alteraria o estado original da evidência. Outro ponto positivo de se efetuar parte da análise *postmortem* em um sistema como o Linux, é a possibilidade de se visualizar as evidências sob um novo "ponto de vista", evitando assim possíveis vícios impostos pelo Windows.

Categoria de evento	Descrição
Account logon events	Ocorre quando a controladora de domínio recebe uma requisição de logon
Account management	Ativado quando um usuário ou grupo é criado ou modificado
Directory service access	Ativado quando um objeto do Active Directory é acessado
Logon events	Ativado quando um usuário efetua login/logoff
Object access	Ativado quando um objeto é acessado
Policy change	Ativado quando políticas envolvendo segurança, privilégios de usuários ou políticas de auditoria são modificados
Privilege use	Ativado quando um privilégio é utilizado para se efetuar uma ação
Process tracking	Ocorre quando uma aplicação efetua uma operação que está sendo monitorada
System events	Acontece quando o computador é reiniciado, desligado ou caso ocorra algo que afete a segurança

Tabela 1: Categorias de eventos auditáveis

Infelizmente o suporte do Linux ao sistema de arquivos NTFS não é completo, fazendo com que diversas estruturas, como as *alternate streams*, sejam ignoradas. Este fato impede que a análise seja totalmente conduzida neste sistema e implica que parte dos procedimentos devam ser efetuados no W2k.

3. http://www.foundstone.com/knowledge/proddesc/forensic-toolkit.html
4. http://www.porcupine.org/forensics/tct.html

Adotando	Problema em Potencial
falhas na categoria *logon/logoff*.	Ataques de dicionário
sucessos na categoria *logon/logoff*.	Utilização de senha roubada
sucessos na utilização de privilégio, no gerenciamento de contas de usuários, mudança nas políticas de segurança, reinicialização, *shutdown* e eventos do sistema.	Mal uso de privilégios
sucessos e falhas em eventos de acesso a arquivos e objetos. Sucessos e falhas de leituras e escritas em arquivos críticos.	Acesso indevido a arquivos críticos
Sucessos e falhas no acesso a impressoras e objetos.	Acesso indevido às impressoras
Sucessos e falhas na escrita em arquivos binários (.EXE e .DLL). Sucessos e falhas na auditoria de processos (*process tracking*). Execução de programas suspeitos, tentativas inesperadas de modificação de binários ou criação de processos.	Ataque de vírus

Table 2: Problemas em potencial, detectados com a auditoria dos eventos apropriados[10]

Em linhas gerais, pode-se dizer que a maioria das conclusões de uma análise forense são baseadas nos resultados obtidos durante a *live analysis*, seguida de análises nos eventos armazenados no *EventViewer* e checagem de integridade de arquivos. Poucas são as vezes em que é possível, ou necessária, a realização de uma análise *postmortem* minuciosa, isto porque muitas vezes é inviável a criação de imagens dos discos por questões de capacidade de armazenamento, ou pelo fato de que uma análise deste tipo implicarem na paralisação dos serviços oferecidos pela máquina atingida, o que muitas vezes não é possível [7].

5 - CONFIGURAÇÃO VISANDO FUTURAS ANÁLISES

Apesar de atualmente a preocupação com segurança ser requisito essencial para as mais variadas aplicações e serviços de rede, ainda é comum encontrar redes W2k com configurações minimalistas que se preocupam apenas com aspectos funcionais, deixando configurações de segurança renegadas a um segundo plano.

Uma das principais causas de escassez de informações fornecidas pelas máquinas W2k durante as análises é o fato de muitos administradores utilizarem configurações *default* do Windows, que geralmente tem pouca preocupação com estes aspectos. Tome-se por exemplo a configuração das políticas de auditoria que não são sequer habilitadas por *default*, fazendo com que deixem de ser registrados qualquer tipo de evento relacionado a segurança.

A configuração de uma máquina onde se leve em consideração a possibilidade de uma futura análise forense pode agilizar de forma substancial a resposta a um incidente de segurança, fazendo com que o problema possa ser mais rápidamente solucionado. Isto se deve ao fato de que uma correta configuração faz com que W2k forneça mais traços das ações de um possível atacante, ampliando o número de possíveis evidências, o que pode influir decisivamente na resolução do caso.

Abaixo temos exemplos de pontos cuja configuração ou implantação poderia ser de grande utilidade para o andamento de uma análise:

- **Audit Polices:** Auditar eventos é uma parte importante da administração de uma rede. Quando o administrador seleciona eventos para serem auditados, ele pode posteriormente descobrir padrões que caracterizam a utilização normal do sistema, possíveis problemas de segurança e observar tendências na utilização dos recursos da rede. Entretanto é necessário cuidado ao se escolher o que será auditado, em virtude do grande número de logs que alguns eventos podem gerar, além da sobrecarga adicionada a cada operação que envolva este tipo de registro [10][16]. Na Tabela 1 temos a descrição de todos os eventos auditáveis do W2k e na Tabela 2 temos algumas interpretações para os logs gerados por eles.

- **ACLs:** Para cada arquivo ou pasta em um volume NTFS, existe uma ACL (*Access Control List*). A ACL contém uma lista de todos os usuários e grupos que possuem permissões para acessar o objeto, assim como o tipo de acesso permitido. Para que determinado usuário possa utilizar determinado objeto é necessário que exista uma ACE (*Access Control Entry*) para ele ou para um grupo ao qual ele pertença. A combinação entre a configuração das ACLs e uma correta configuração das políticas de auditoria podem revelar diversos traços de possíveis atividades ilegais de um usuário.

- **File Integrity Check:** É muito difícil atacar com sucesso um sistema sem alterar o seu sistema de arquivos, este fato faz com que a capacidade de

checar a integridade de arquivos se torne uma importante fonte de possíveis evidências. Através de um programa deste tipo pode-se armazenar os hashes criptográficos de todos os arquivos críticos do sistema podendo-se posteriormente calcular um novo hash e então compará-lo com o valor armazenado verificando assim se houve alguma alteração.[10][20]

6 - PRE-FORENSIC SETUP AUTOMATION FRAMEWORK

A popularização das redes corporativas e o seu espantoso crescimento de tamanho fez com que a criação de técnicas para a administração de redes com grande número de máquinas se tornasse um ramo promissor de pesquisas. Inúmeras técnicas surgiram dando prioridade ao mundo UNIX, bem mais comum em redes de grande porte até bem pouco tempo atrás. Entretanto, a partir do Windows NT 4.0 a Microsoft começou a atuar de forma incisiva neste mercado e atualmente é comum encontrar-se redes com algumas centenas de máquinas utilizando o Windows 2000.

Ao contrário dos sistemas UNIX, o Windows é considerado um sistema "*hands-on*", que exige a presença ou atuação do administrador em cada máquina para que sejam feitas a maioria das instalações, configurações e demais tarefas administrativas necessárias em uma rede. Este fato, torna a administração de uma rede Windows, com um grande número de máquinas, uma tarefa extremamente trabalhosa e pouco produtiva.

As dificuldades administrativas impostas pelas plataformas Windows em geral são um grande empecilho à implantação das recomendações citadas na Seção 5, fazendo com que a implantação de alguns destes requisitos consumam um tempo demasiado grande do Time de Resposta. Nesta seção apresenta-se a implementação de um *Framework* que busca automatizar as principais ações necessárias à preparação das máquinas de uma rede para a possibilidade de uma futura análise forense.

6.1 - Estrutura

O PFSAF foi estruturado de maneira a permitir a realização de todas as configurações necessárias a partir de uma única máquina, podendo ainda ter tarefas agendadas para o horário mais conveniente para o administrador, através do *Task Scheduler*. Esta máquina (*Forensic Station*) deve ser de acesso restrito e será responsável pelo armazenamento de todos os arquivos de configuração e bases de dados geradas pelo PFSAF (Figura 2).

As máquinas que serão monitoradas não necessitam de nenhuma configuração especial, exigindo apenas a existência dos *default shares* do W2k (Figura 3).

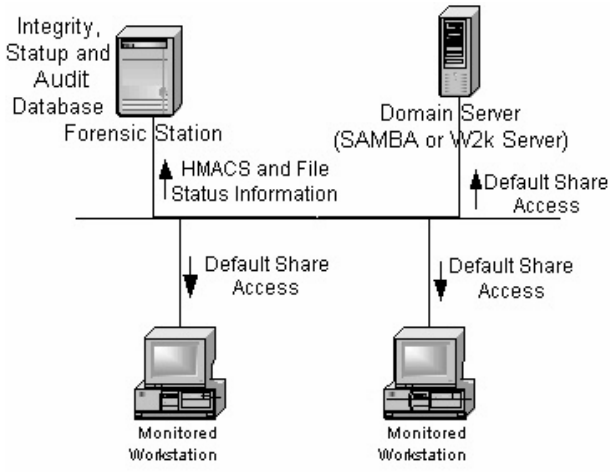

Figura 2: ***File Integrity Module Schema***

Atualmente as principais funcionalidades do PFSAF são:

- **Garantir a integridade de arquivos críticos**: o módulo para integridade de arquivos do PFSAF gera hashes criptográficos, registra a existência e o tamanho de alternate streams e armazena os extremamente voláteis MACTimes (tempos de acesso, modificação e criação de um arquivo);

- **Monitoramento dos programas com execução automática**: todos os programas automatimente executados após o logon dos usuários tem sua integridade monitorada, além disso qualquer alteração no número de programas a serem executados ou alteração em suas características pode ser detectada;

- **Configuração de políticas de auditoria**: PFSAF pode configurar políticas de auditoria remotamente e também checar se elas foram posteriormente alteradas.

6.2 - Implementação

O PFSAF foi implementado utilizando a linguagem PERL, que é extremamente versátil na manipulação de arquivos e *strings* (necessidade básica de nossa implementação), contando também com diversas bibliotecas[5] voltadas para a administração de máquinas Windows, apesar de ser oriunda das plataformas Unix. Além disso, possui ampla documenta-

5. A critério de notação as extensões da linguagem PERL, comumente chamadas de módulos, serão tratadas aqui como bibliotecas para não haver confusão com blocos do PFSAF, que também recebem a designação de módulos.

ção e é distribuída sob os termos da licença GNU GPL.

O PFSAF foi testado com o interpretador ActiveState[6] 5.6.1 Build 631 em uma máquina com Windows 2000 Server e Service Pack 2, tendo como clientes máquinas Windows 2000 Server e Professional com variadas combinações de Sevice Pack.

Existem basicamente dois arquivos de configuração neste *framework*, o `files.cfg` (Exemplo 1) e o `audity.cfg` (Exemplo 2). O arquivo `files.cfg` armazena os nomes dos arquivos a serem incluídos na base dados de integridade (`hash.db`), que é responsável pelo armazenamento de hashes criptográficos e mais alguns dados que descrevem a estrutura e o estado do arquivo alvo.

```
c:\boot.ini
c:\io.sys
c:\config.sys
c:\autoexec.bat
c:\winnt\system32\fpnwclnt.dll
```

Exemplo 1: Arquivo files.cfg

```
#Event Order:
#
#server(Process Tracking,
#       Account Longon,
#       Restart and Shutdown,
#       User/Group Management,
#       Dir. Service Access,
#       File and Object Access,
#       Security Policy Changes,
#       Logon and Logoff,
#       Use of User Rights)
#
#Audit Values:
#
#        None                -> 0
#        Success             -> 1
#        Failure             -> 2
#        Success and Failure -> 3
#        Unchanged           -> 4
#
#Example:
#
#testserver(2,0,0,4,0,0,0,3,0)

obiwan(0,0,0,0,0,0,0,3,0)
jaba(0,0,0,0,0,0,0,2,0)
anakin(0,2,0,0,0,0,0,3,0)
```

Exemplo 2: Arquivo audit.cfg

O `files.cfg` é utilizado para todas as máquinas listadas no arquivo `audit.cfg`, a não ser que exista um arquivo especifico para determinada máquina, cujo nome siga a seguinte sintaxe: `<máquina>-files.cfg`. Neste caso, o arquivo `files.cfg` é preterido e a listagem de arquivos presente na configuração mais específica é utilizada.

Os nomes das máquinas a serem monitoradas são armazenadas no arquivo `audity.cfg` que, além disso, descreve quais políticas de auditoria devem ser aplicadas em cada máquina citada.

Nesta seção serão abordados detalhes de implementação de cada módulo citado na Seção 6.1.

6.2.1 - File Integrity Module

O módulo de integridade de arquivos utiliza os dois arquivos de configuração do PFSAF o `files.cfg`, que é específico para este módulo e o `audit.cfg` que contém a lista de máquinas a serem monitoradas.

O acesso aos arquivos remotos é feito através dos *default shares* do Windows (Figura 3), que embora combatido por alguns, tornou-se uma funcionalidade extremamente útil para a administração do sistema.

Para os hashes criptográficos, o PFSAF utiliza o algorítmo SHA1 em conjunto com a primitiva de chave simétrica HMAC [8], também chamado de HMAC-SHA1. A utilização HMAC-SHA1 inviabiliza um ataque de simples *cut-and-paste* ao arquivo `hahs.db`, no qual se poderia substituir o hash armazenado por um outro, validando assim, uma possível alteração nos arquivos monitorados.

O HMAC-SHA1 utiliza uma chave para calcular os hashes dos arquivos, desta forma não é possível a substituição dos valores contidos no `hashs.db`, a menos que se tenha conhecimento da chave secreta. No caso do PFSAF, a senha utilizada no ato da geração da base de dados é utilizada como chave e seu hash é armazenado no arquivo `passwd`. O suporte ao algoritmo HMAC-SHA1 no Perl é dado pela biblioteca `Digest::HMAC_SHA1`.

6.2.2 - Startup Integrity Database

O módulo *Startup Integrity* é responsável por monitorar e garantir a integridade de todos os programas executados automaticamente após o logon dos usuários. O objetivo é evitar que um programa maliciosamente alterado, ou mesmo um *backdoor* seja executado de forma automática em uma das máquinas monitoradas.

Existem basicamente duas formas de um programa ser executado automaticamente quando um usuário se "loga" na máquina. A mais simples é através da inclusão de atalhos na pasta *Startup* presente em todos os profiles de usuário. Para este caso o PFSAF faz acesso à pasta que contém os *profiles* dos usuários utilizando novamente o compartilhamento *default* do W2k. A análise dos atalhos é possível graças à utilização do `Win32::Shortcut`, biblioteca

6. http://www.activestate.com/Products/ActivePerl/

```
C:\>net share
Share name   Resource                        Remark
------------------------------------------------------------------
IPC$                                         Remote IPC
D$           D:\                             Default share
G$           G:\
ADMIN$       C:\WINNT                        Remote Admin
C$           C:\                             Default share
The command completed successfully.
```

Figura 3: *Default shares* em uma máquinaW2k

do Perl que fornece a interface para a manipulação deste tipo de arquivo.

Uma vez descoberto o binário ao qual o atalho se refere, o PFSAF adiciona dois registros à *Startup* Database. Um contendo o hash e o estado do arquivo de atalho e um outro contendo as informações referentes ao binário propriamente dito.

A outra forma de execução automática de programas é através da inclusão de valores em algumas chaves de registro, são elas:

- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`: Os programas contidos nesta chave são executados simultaneamente após o login de qualquer usuário;

- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce`: Os valores pertencentes a esta chave são executados de forma seqüencial, obedecendo a ordem em que foram adicionados, que pode não ser a ordem em que aparecem no editor de registro;

- `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx`: Utilizada pelo *Windows Desktop Update*, esta chave é uma tentativa de tornar mais robusta a execução automática de programas, pois conta com mecanismos para manipulação e documentação de erros além de contar com meios para aumentar o desempenho das aplicações. Mais informações podem ser obtidas em [11];

- `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run`: Diferentemente da chave `Run` da hierarquia `LOCAL_MACHINE` que se refere a todos os usuários, esta chave traz configurações exclusivas do usuário que se "loga". Tais informações são instaciadas no ato do logon, a partir das informações contidas na hierarquia `HKEY_USERS`.

- `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Runonce`: Funcionamento semelhante ao de sua irmã pertencente à hierarquia `LOCAL_MACHINE`, contudo possui configurações exclusivas para cada usuário.

O PFSAF conta também com um *parser*, encarregado de analisar chamadas de interpretadores de scripts nativos do W2k, como o `wscript.exe` e o `cmd.exe`, que podem estar presentes tanto nos valores das chaves acima, quanto nos atalhos da pasta *Startup*.

A linha de comando para a execução de um *script* pode ser bastante complexa, em virtude do grande número de opções oferecidas pelos interpretadores. Este parser é utilizado na tentativa de encontrar o nome do arquivo fonte do *script* em meio a todas as opções possíveis, para que assim possa adicionado à base de dados de integridade. Os interpretadores suportados são:

- **wscript e cscript**: Ambos são parte do Microsoft® Windows® Script Host (WSH). O WSH funciona como um ambiente único para execução de todas as linguagens interpretadas compatíveis com sua estrutura (vbscript, javascript, etc.), permitindo a execução de scripts que contam com interface gráfica (wscript) e também os que possuem interface via linha de comando (cscript);

- **cmd**: é o interpretador de comandos padrão do W2k utilizado para executar scripts do tipo *batch*, comuns a todos os sistemas operacionais da Microsoft.

- **rundll**: O `rundll32.exe` permite a execução de funções exportadas por DLLs de 32 bits, contudo só é possível utilizar funções que forem explicitamente escritas para serem invocadas por ele. [12]

6.2.3 - Audit Policy Module

O objetivo do *audit policy module* é fornecer uma nova interface para configuração de políticas de auditoria, permitindo que PDCs que não compartilham as configurações do domínio, máquinas isoladas ou domínios gerenciados por máquinas Linux através do SAMBA, possam ser configuradas rápidamente de forma automatizada.

Através da utilização do `audit.cfg` (Exemplo 2) o *audit policy module* pode configurar as máqui-

nas nele presente com sua respectiva política. Este módulo também pode checar o estado atual das políticas em cada máquina e verificar se estas correspondem aos valores presentes no `audit.cfg`.

O acesso a esse tipo de configuração no Perl é feito através da biblioteca `Win32::Lanman` que funciona como um *wrapper* para a LANMAN API do Windows.

6.3 - Utilização

A utilização do PFSAF pode ser feita de forma interativa ou automática. No caso de se optar pela interatividade, o programa conta com o script main.pl (Apêndice A.1) que provê uma interface para a utilização de todo o *framework* (Figura 4), entretanto os arquivos de configuração já devem estar previamente preparados.

```
C:\> main.pl


        [1] Generate Integrity File Database.
        [2] Check Integrity File Database.
        [3] Set Audit Policy.
        [4] Check Audit Policy.
        [5] Query Autit Policy.
        [6] Generate Startup Database.
        [7] Check Startup Database.
        [8] Query Startup Database.
        [9] Help.
        [10] Quit.

                Choose one Option:
```

Figura 4: Opções oferecidas pelo main.pl

A utilização não interativa do PFSAF pode ser feita através da chamada de módulos individuais (Apêndice A.1), desenvolvidos separadamente para fornecer uma interface mais flexível ao *framework* e possibilitando assim, a execução de módulos independentes através do *Task Scheduler* do Windows.

6.4 - Trabalhos Futuros

O PFSAF está longe de estar terminado, ainda existem diversas funcionalidades que poderiam ser agregadas a fim de facilitar o trabalho dos Times de Resposta e administradores, tais como a configuração remota de ACLs e uma possível agregação de funções oferecidas pelo DoIt4Me [1], portando parte de seu código para o W2k.

Uma outra possível funcionalidade que poderia ser implementada seria a criação de uma interface para a sincronização dos relógios das máquinas monitoradas, o que torna a análise dos MACTimes bem mais amigável.

Algumas funções atuais também podem ser modificadas com o intuito de tornar o PFSAF mais flexível, tal como a ampliação da sintaxe dos arquivos de configuração com o intuito de condensar especificações semelhantes através da utilização de metacaracteres.

7 - CONCLUSÕES

A criação de ferramentas como o PFSAF podem incentivar a criação de programas de resposta a incidentes por parte das instituições, uma vez que reduzem a complexidade de sua implantação em redes baseadas em Windows 2000, através da automatização de tarefas. Além disso, tornam visíveis alguns requisitos básicos deste tipo de programa o que conseqüentemente ajuda a difundir a sua necessidade de sua implantação.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] AUGUSTO, Alessandro; GEUS, Paulo L.; GUIMARÃES, Célio C.; Administration of Large Windows NT Network with DoIt4Me; *The 10th International Conference on System Administration, Networking and Security*; Baltimore, MD, USA; May 2001;

[2] BREZINSKI, Dominique; Building a Forensic Toolkit That Will Protect You From Evil Influences; *The Black Hat Briefings '99;* Las Vegas, NV - USA;

[3] CARPENTER, Jeffrey J.; Welcome to The Big City; *;login: The Magazine of USENIX & SAGE*; Novembro 1999;

[4] CARVEY, Harlan; System Security Administration for NT; *LISA-NT -- The 3rd Large Installation System Administration of Windows NT Conference*; Seattle, Washington - USA; 2000

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

[6] LEE, Henry; PALMBACH, Timothy; MILLER, Marilyn; *Henry Lee's Crime Scene Handbook*; Academic Press; 2000;

[7] MANDIA, Kevin; PROSISE, Chris; *Incident Response: Investigating Computer Crime*; Osborne/McGraw Hill; 2001;

[8] MENEZES, Alfred J.; van OORSCHOT, Paul C.; VANSTONE, Scott A.; *Handbook of Applied Cryptography*; CRC Press; 1996;

[9] MICROSOFT, Corp.; *MCSE Training Kit – Microsoft Windows 2000 Active Directory Services*; Microsoft Press; 2000;

[10] MICROSOFT, Corp.; Securing Windows® 2000 Network Resources; *Microsoft TechNet*; Microsoft Corporation; July 2000;

[11] MICROSOFT, Corp; Description of RunOnceEx and RunEx Registry Keys [Q232487]; *Microsoft TechNet*; Microsoft Corporation; July 2000;

[12] MICROSOFT, Corp; The Windows 95 Rundll and Rundll32 Interface [Q164787]; *Microsoft TechNet*; Microsoft Corporation; July 2000;

[13] NOBLETT, Michael G.; POLLITT, Mark M.; PRESLEY, Lawrence A.; Recovering and Examining Computer Forensic Evidence; *Forense Science Communications*, Vol. 2 N. 4 october 2000; Federal Bureau of Investigation;

[14] OLIVEIRA, Flávio de Souza; GUIMARÃES, Célio Cardoso; REIS, Marcelo; GEUS, Paulo Lício de; Forense Computacional: Aspectos Legais e Padronização; *Wseg'2001 - I Workshop de Segurança em Sistemas Computacionais Procedings*; Florianópolis, SC - Brazil; pp. 80-85

[15] OLIVEIRA, Flávio de Souza; GUIMARÃES, Célio Cardoso; REIS, Marcelo; GEUS, Paulo Lício de; Metodologias de Análise Forense para Ambiente Baseados em NTFS; *SSI'2001 - III Simpósio de Segurança em Informática Procedings*; São José dos Campos, SP - Brazil; pp. 83-90;

[16] REIS, Marcelo A.; GEUS, Paulo L.; Stardardization of Computer Forensic Protocos and Procedures; *14th Annual FIRST Computer Security Incident Handling Conference*; Hawaii; June 2002

[17] RUSSEL, Charlie; CRAWFORD, Sharon; *Microsoft® Windows® 2000 Server Administrator's Companion*; Microsoft Press; 2000;

[18] THORTON, J.; The general assumptions and rationale of forensic identification; *Modern Scientific Evidence: The Law and Science of Expert Testimony*; West Publishing Co.; Volume 2;

[19] WYK, Kenneth R. van; FORNO, Richard; *Incident Response*; O'Reilly & Associates, Inc.; 2001;

[20] Tripwire Open Source's Home Page: http://www.tripwire.org;

APÊNDICE A.1 - ARQUIVOS

- `audit.cfg`: arquivo que contém a lista de máquinas a serem monitoradas pelo PFSAF e suas respectivas políticas de auditoria (Exemplo 2);

- `files.cfg`: relação de arquivos que serão monitorados pelo file integrity module. Este arquivo será utilizado para todas as máquinas contidas no `audit.cfg`, a não ser que exista uma configuração especifica para determinada máquina em um arquivo cujo nome siga a seguinte sintaxe, `<maquina>-files.cfg`. (Exemplo 1);

- `main.pl`: script responsável pela interface interativa do PFSAF. Exibe um menu de opções e executa os scripts apropriados em função da escolha do usuário;

- `genintdb.pl`: responsável pela geração da base de dados de integridade de arquivos, faz acesso às máquinas contidas no arquivo `audit.cfg` e coleta informações dos arquivos contidos no `file.cfg`;

- `chkintdb.pl`: faz a checagem da base de dados de integridade de arquivos e gera relatório;

- `adscount.pl`: script auxiliar encarregado de fazer a contagem das alternate streams de um arquivo e calcular seu tamanho total;

- `genstartdb.pl`: faz acesso à pasta Startup de todos os usuários e às chaves de registro utilizadas para a inicialização automática de programas, analisa todas as entradas e gera uma base de dados com as informações coletadas;

- `chkstartdb.pl`: efetua a checagem das informações contidas na startup database e gera relatório das possíveis ocorrências;

- `keytime.pl`: script auxiliar responsável pela coleta de informações referente às chaves de registro, tais como número de valores e data da última alteração;

- `parser.pl`: responsável por analisar linhas de comandos que envolvam a chamada de interpretadores de scripts nativos do W2k;

- `auditpol.pl`: faz a configuração das políticas de auditoria contidas no arquivo `audit.cfg` em suas respectivas máquinas;

- `chkaudit.pl`: verifica se as políticas de auditoria nas máquinas monitoradas estão de acordo com as configurações no arquivo `audit.cfg`;

- `hashs.db`: file integrity database;

- `start.db`: startup database;

- `passwd`: chave secreta utilizada pelo PFSAF durante a geração dos hashes criptográficos;