Ministério da
Ciência e Tecnologia

INPE-15340-TDI/1376

APLICAÇÃO DE TÉCNICAS DE DATA MINING PARA A ANÁLISE DE LOGS DE TRÁFEGO TCP/IP

André Ricardo Abed Grégio

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Antonio Montes Filho e Rafael Duarte Coelho dos Santos, aprovada em 27 de fevereiro de 2007.

Registro do documento original:
<http://urlib.net/sid.inpe.br/mtc-m17@80/2007/04.04.18.29>

INPE
São José dos Campos
2008

PUBLICADO POR:

Instituto Nacional de Pesquisas Espaciais - INPE
Gabinete do Diretor (GB)
Serviço de Informação e Documentação (SID)
Caixa Postal 515 - CEP 12.245-970
São José dos Campos - SP - Brasil
Tel.:(012) 3945-6911/6923
Fax: (012) 3945-6919
E-mail: pubtc@sid.inpe.br

Ministério da
Ciência e Tecnologia

INPE-15340-TDI/1376

APLICAÇÃO DE TÉCNICAS DE DATA MINING PARA A ANÁLISE DE LOGS DE TRÁFEGO TCP/IP

André Ricardo Abed Grégio

Dissertação de Mestrado do Curso de Pós-Graduação em Computação Aplicada, orientada pelos Drs. Antonio Montes Filho e Rafael Duarte Coelho dos Santos, aprovada em 27 de fevereiro de 2007.

Registro do documento original:
<http://urlib.net/sid.inpe.br/mtc-m17@80/2007/04.04.18.29>

INPE
São José dos Campos
2008

Dados Internacionais de Catalogação na Publicação (CIP)

G861a Grégio, André Ricardo Abed.
Aplicação de técnicas de data mining para a análise de logs de tráfego TCP/IP)/ André Ricardo Abed Grégio. – São José dos Campos: INPE, 2008.
134p. ; (INPE-15340-TDI/1376)

1. Redes de computadores. 2. Data mining. 3. Segurança de sistemas de informação 4. Análise de logs. 5. Detecção de intrusão. I. Título.

CDU 004.7

Aprovado (a) pela Banca Examinadora em cumprimento ao requisito exigido para obtenção do Título de Mestre em

Computação Aplicada

Dr. José Demisio Simões da Silva

Presidente / INPE / SJCampos - SP

Dr. Antonio Montes Filho

Orientador(a) / INPE / SJCampos - SP

Dr. Rafael Duarte Coelho dos Santos

Orientador(a) / INPE / SJCampos - SP

Dr. José Carlos Becceneri

Membro da Banca / INPE / SJCampos - SP

Dr. Paulo Lício de Geus

Convidado(a) / UNICAMP / Campinas - SP

Aluno (a): André Ricardo Abed Grégio

São José dos Campos, 27 de Fevereiro de 2007

"I did have strange ideas during certain periods of time".

John Forbes Nash, Jr.

Àqueles que me incentivaram nesta caminhada.

AGRADECIMENTOS

Muitas pessoas têm de ser agradecidas neste trabalho. Meus pais, meu irmão, meus familiares, amigos e todas as pessoas que direta ou indiretamente contribuíram para que eu pudesse defender esta dissertação na data em que ela foi apresentada, seja me ajudando, seja compreendendo minha ausência.

Não há como citar todos que me deram apoio ou auxílio sem ser injusto e esquecer de alguém. Cada um que me deu força durante o trajeto do mestrado é também um pouco autor deste trabalho. Sem o apoio destas pessoas tão especiais certamente eu não teria conseguido.

Meu muito obrigado aos meus orientadores Prof. Montes e Prof. Rafael, que brilhantemente me conduziram e principalmente me suportaram e cuidaram de mim em minha passagem por São José dos Campos.

Aos professores e funcionários do INPE, pelos ensinamentos, ajuda, convívio e pelos momentos no café.

Aos meus amigos (e *co-workers*) que já têm me acompanhado há um tempo: Limão, Faiskaum e Gustavo (Oknawa).

Ao Adriano Cansian por ter me aconselhado no início do meu mestrado e por ter sempre me oferecido seu suporte e amizade.

Ao Benicio, por tudo. Nunca poderei agradecer adequadamente a oportunidade de compartilhar sua sabedoria.

RESUMO

Com a popularização da Internet nos últimos anos, a variedade de serviços providos através de redes de computadores aumentou consideravelmente. Ao mesmo tempo surgiram novas formas de crimes usando estes serviços, muitos através da tentativa de invasão ou comprometimento de redes. O registro das atividades de redes e aplicações – através da coleta e armazenamento de registros de auditoria (*logs*) – é muito importante para que se possa analisar eventos e descobrir erros, anomalias ou até mesmo caracterizar ataques ou intrusões. Ainda hoje, a área de análise de *logs* carece de ferramentas que façam uma separação adequada dos *logs* interessantes para serem analisados por um humano. Isto torna a tarefa de análise muitas vezes impossível de ser cumprida, podendo levar ao comprometimento da rede de uma instituição sem que contra-medidas sejam tomadas em tempo hábil. Neste trabalho tem-se por objetivo expor o problema da análise de *logs*, discutir as abordagens para filtragem, tratamento de *logs* e detecção de intrusão por técnicas de mineração de dados, avaliar alguns algoritmos de mineração de dados para aplicá-los em separação de *logs* do tráfego de rede e construir um protótipo para realizar a redução de *logs* com uma taxa aceitável de falsos-positivos. Os testes feitos com os algoritmos dos vizinhos mais próximos, *perceptrons* de múltiplas camadas e árvores de decisão permitiram a implementação de um protótipo modular que utiliza árvores de decisão para automatizar a classificação de *logs* e reduzí-los para um pequeno conjunto de sessões suspeitas. É mostrado um estudo de caso com a aplicação deste protótipo, onde apresentam-se os resultados obtidos de mais de 90% de redução nos conjuntos de *logs* disponíveis.

APPLICATION OF DATA MINING TECHNIQUES TO TCP/IP NETWORK TRAFFIC LOGS' ANALYSIS

ABSTRACT

Since the popularization of the Internet in recent years, the amount and variety of computer network services has greatly increased. At the same time new methods to commit crimes using these services appeared, mostly by attempting intrusion or compromising networks. Logging network and application activities – through collection and storing of network logs – is a very important step to allow further event analysis in order to discover errors, anomalies or even to characterize attacks and intrusion. In spite of the importance of this task, even today log analysts suffer from a lack of tools to classify correctly interesting logs, making the analysis task impossible to be accomplished timely. This fact can lead to the compromise of an institution's network without counter measures being taken in time. In this work we aim to present the log analysis problems, to discuss log filtering, handling and intrusion detection approaches through data mining techniques, to evaluate some data mining algorithms to apply them on network traffic logs separation and to build a prototype to perform log reduction with an acceptable rate of false positives. Tests were done with some algorithms such as nearest neighbors, multilayer perceptrons and decision trees, which allowed the deployment of a modular prototype using decision trees to automatize log classification and reduce logs to a small set of suspicious sessions. A case study containing the prototype application and the results obtained with reduction rates in the log sets greater than 90% are also presented.

SUMÁRIO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE SIGLAS E ABREVIATURAS

ACK	–	*Acknowledgement*
ACL	–	*Access Control List*
ARFF	–	*Attribute-Relation File Format*
CBH	–	Consórcio Brasileiro de *Honeypots*
CERT	–	*Computer Emergency Response Team*
CERT.br	–	Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança do Brasil
DMZ	–	*De-Militarized Zone*
DNS	–	*Domain Name System*
DoS	–	*Denial of Service*
FTP	–	*File Transfer Protocol*
HTTP	–	*Hyper Text Transfer Protocol*
ICMP	–	*Internet Control Message Protocol*
IDS	–	*Intrusion Detection System*
IP	–	*Internet Protocol*
KNN	–	*K-Nearest Neighbors*
LAN	–	*Local Area Network*
MLP	–	*Multi Layer Perceptron*
SMTP	–	*Simple Mail Transfer Protocol*
SSH	–	*Secure Shell*
TCP	–	*Transmission Control Protocol*
UDP	–	*User Datagram Protocol*
WEKA	–	*Waikato Environment for Knowledge Analysis*
WWW	–	*World Wide Web*

CAPÍTULO 1

INTRODUÇÃO

No início dos anos 90, a Internet deixou de ser uma rede exclusivamente acadêmica e de pesquisas, passando a ser disponível para o público em geral. Isto possibilitou que diversos tipos de instituições e empresas disponibilizassem conteúdo e interagissem com usuários de todo o planeta através de novas formas de comunicação. Surgiu então a necessidade de uma interface amigável, com a possibilidade de prover conteúdo multimídia, para permitir a integração deste contingente de novos usuários e suas máquinas com a Internet. Tal aplicação foi inventada por Tim Berners-Lee e denominada *World Wide Web*, ou (WWW) (KUROSE; ROSS, 2002).

A Web, como ficou conhecida a aplicação de Lee, acabou por revolucionar as formas de comunicação, por prover conteúdo sob a demanda do usuário (diferentemente do rádio e da televisão), e também a economia, pois a disseminação da Internet propiciou um novo ambiente para se fazer negócios. Isto, aliado à popularização dos computadores pessoais, modificou radicalmente o comportamento das pessoas, os níveis de interação social e as formas de se cometer crimes, uma vez que a Internet possibilitou o acesso remoto a computadores a partir de qualquer ponto do mundo, com um certo grau de anonimato.

O amplo acesso à Internet e a falta de controle do conteúdo tornou possível a disseminação de informações falsas, podendo levar usuários a acessar páginas Web ou mensagens de correio eletrônico com conteúdo malicioso. Mais grave ainda é a possibilidade de se realizar atividades ilegais de maneira anônima. Assim, programas maliciosos podem ser instalados no computador do usuário de forma oculta, permitindo a um invasor o roubo de informações pessoais como senhas e dados bancários, além desta máquina poder servir de trampolim para outras atividades de ataque, camuflando o real atacante.

1.1 Motivação e Objetivos

A grave situação relacionada aos incidentes de segurança no Brasil pode ser observada através das estatísticas contabilizadas pelo Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br)[1]. Estatísticas compre-

[1]Maiores informações sobre o CERT.br estão disponíveis em <http://www.cert.br>, acessado em janeiro de 2007.

endendo os meses de janeiro a dezembro de 2006 mostram que dos mais de 197000 incidentes reportados, 55% correspondem a *worms* e 21% à fraudes.

No fim da década de 80 e início da década de 90, os ataques lançados contra redes de computadores dependiam essencialmente das habilidades técnicas do atacante. O invasor em potencial tinha de conhecer profundamente o sistema operacional e a aplicação a ser explorada para ganhar acesso e então realizar suas atividades maliciosas. Nesta época ainda havia grande relação de confiança entre as máquinas interconectadas, visto que o objetivo principal era a efetivação da comunicação, facilitando a acessibilidade da rede como um todo. Um exemplo de ataque a este tipo de modo de operação ficou famoso e é descrito no livro *The Cuckoo's Egg: Tracking a Spy through the Maze of Computer Espionage* de Clifford Stoll (STOLL, 1989).

Mesmo hoje em dia, apesar de existir toda uma parafernália de dispositivos e *software* relacionados à segurança de sistemas de informação, esta ainda não é a preocupação primordial de muitas instituições. Aliado a isto, a facilidade (e automatização) da criação de ferramentas com a finalidade de atacar sistemas, os chamados *software* maliciosos (*malware*), fez com que ferramentas de exploração passassem a estar disponíveis para qualquer indíviduo com acesso à rede, aumentando consideravelmente o número de ataques não direcionados. Isto causou um efeito que pode ser observado na Figura 1.1, a qual mostra que, com o passar dos anos, os ataques tornaram-se cada vez mais sofisticados, apesar do conhecimento dos invasores ter diminuído drasticamente.

O objetivo final de um ataque, lançado diretamente por um invasor ou por meio de algum *malware*, é a apropriação do sistema de forma a conseguir acesso a informações privadas ou atacar outras máquinas. Embora os ataques sejam cada vez mais freqüentes, muitos deles são realizados por ferramentas automatizadas e, em muitos casos, não surtem efeito algum, pois a falta de definição de um alvo causa tentativas de se explorar qualquer máquina que for alcançada pelo mecanismo de busca do *malware* em questão, independentemente dela possuir ou não a vulnerabilidade que está sendo atacada.

Grande parte das redes institucionais possui um segmento composto por máquinas que interagem tanto com a rede interna como com a rede externa. Esta área é denominada DMZ (*De-Militarized Zone*) e contém provedores de serviços diversos,

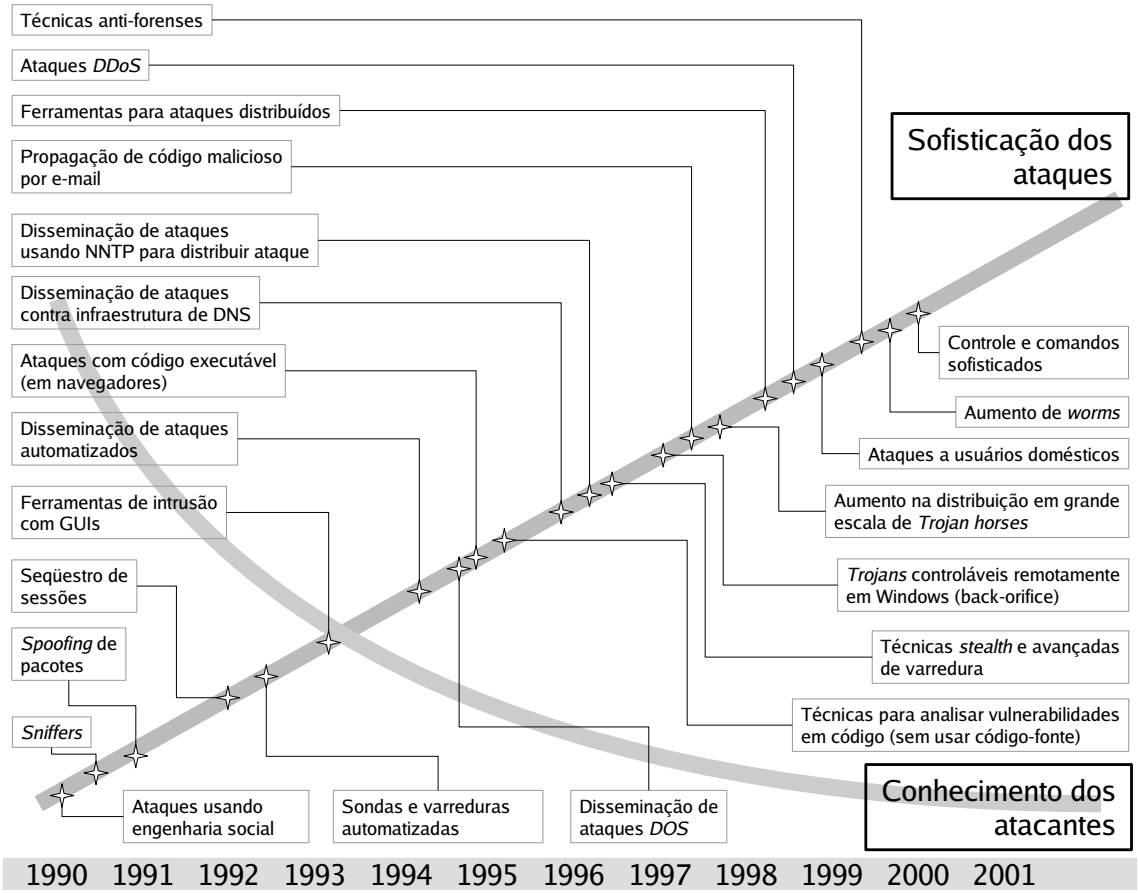

FIGURA 1.1 - Conhecimento dos atacantes vs. Sofisticação dos ataques.
Fonte: Adaptado de (CERT, 2006).

como por exemplo, o servidor de páginas Web da instituição. Por estes serviços estarem expostos à Internet, os ataques têm mais chances de serem bem sucedidos, tornando tais servidores mais visados por invasores e alcançáveis por *malware*. Então, além de uma preocupação maior com a configuração e segurança destes servidores, é necessário que sejam coletados registros de auditoria (*logs*).

Os registros de auditoria podem ser de vários tipos: registros do tráfego de rede, do sistema operacional, das aplicações, entre outros. Cabe ressaltar que as informações geradas e armazenadas são volumosas, pois toda a sorte de eventos pode ser registrada, como a inicialização de determinado aplicativo, erros ocorridos durante sua execução, momento e causa de sua parada, cada qual em registros específicos. Para posterior auditoria, um registro deve conter informações que permitam reconstituir os eventos ocorridos, as quais são dependentes do tipo de evento, tais como data,

hora, endereço de origem e destino, aplicação que gerou o registro e o evento em si. O objetivo disto é a criação de evidências que levem à descoberta do comprometimento de alguma máquina da instituição. Entretanto, a já citada falta de alvos específicos para os ataques, bem como o volume de informações referentes às atividades normais dos sistemas, costumam gerar uma grande quantidade de informações irrelevantes (ruído), tornando maior a massa de dados a ser analisada pelo administrador de redes ou analista de segurança em busca de evidências.

Em uma rede com um sistema de detecção de intrusão, este ruído causa ainda mais problemas. Estes sistemas podem operar basicamente de duas formas: comparando o conteúdo de cada pacote recebido com uma base de dados em busca de assinaturas de ataque (detecção por abuso) ou comparando o fluxo de dados com um perfil normal estabelecido para verificar anomalias no tráfego (detecção por anomalia). Em ambos os casos, o ruído faz com que os sistemas de detecção de intrusão emitam uma infinidade de falsos alertas. Mesmo com um bom sistema de registro e armazenamento de *logs*, sistemas de detecção de intrusão e *scripts* desenvolvidos para automatizar tarefas de pré-análise, o processo sempre culmina na atuação final do analista de segurança, seja para reconfigurar um *firewall* ou inserir alguma regra de controle de acesso (ACL) nos roteadores para tentar bloquear ataques, seja para auditar um sistema comprometido.

Apesar de o número de tentativas de ataque aumentar consideravelmente a cada ano, a quantidade de incidentes de segurança reportados também aumenta, conforme pode-se observar na Figura 1.2. O reporte destes incidentes só é possivel se os incidentes forem detectados, e a detecção e o consequente aviso da tentativa (fracassada ou não) de ataque para o órgão responsável envolve a análise de *logs*. Por isso, de nada adianta dispor de procedimentos elaborados de armazenamento remoto de registros de auditoria e emissão de alertas, se não houver um administrador de redes ou analista de segurança treinado para avaliá-los e tomar providências, ou ainda, se houver tanto ruído no objeto da análise que torne tal tarefa impraticável em tempo hábil para evitar comprometimentos maiores ou ataques a outras redes.

Todavia, existem várias técnicas destinadas a reduzir a quantidade de informações a ser analisadas (SCHMIDT, 2006) (BABBIN *et al.*, 2006) e, em particular, várias técnicas de mineração de dados têm sido utilizadas com relativo sucesso na área de análise de tráfego de rede e detecção de intrusos, como pode ser visto em (STOLFO *et al.*, 2001). A aplicação de métodos de mineração de dados têm a vantagem de permitir a auto-

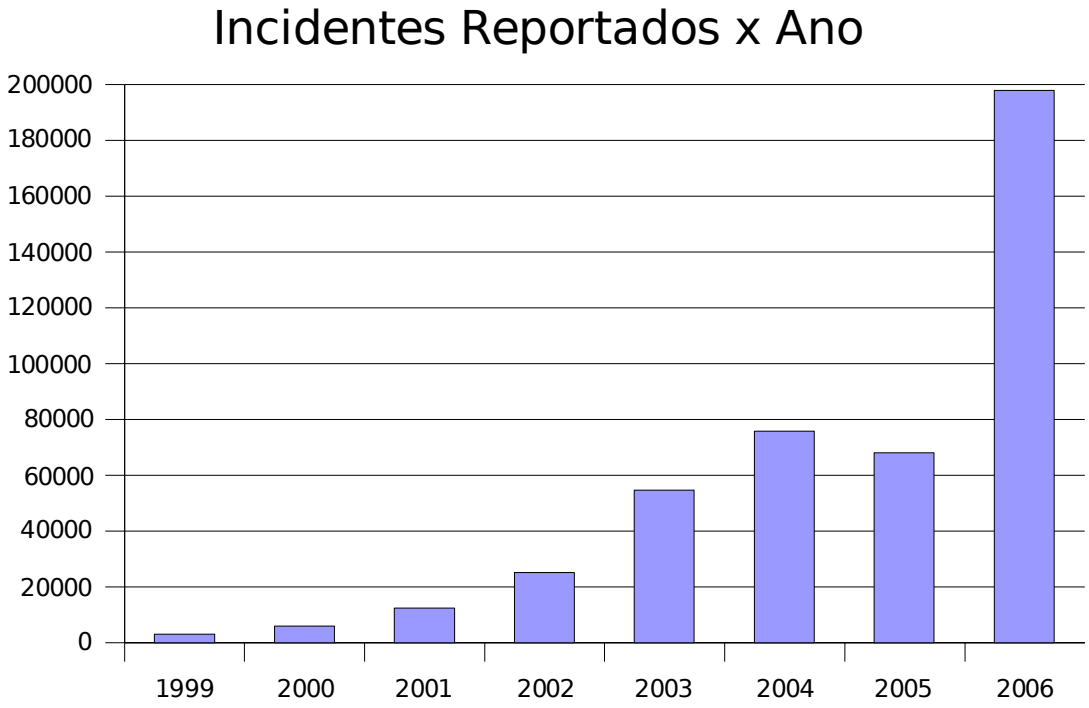

FIGURA 1.2 - Número total de incidentes reportados ao CERT.br por ano.
Fonte: Adaptado de (CERT.BR, 2007b).

matização da modificação de modelos gerados, através de métodos supervisionados ou não supervisionados, ou seja, algoritmos que adaptam o modelo com ou sem a interferência de um agente externo (por exemplo, humano), respectivamente. Com isto, dados alguns critérios explicados nos próximos capítulos desta dissertação, é possível separar adequadamente os *logs* de forma a prover um conjunto reduzido de informações que sejam realmente úteis para o analista de segurança coletar evidências e compreender melhor as atividades de sua rede.

O objetivo inicial deste trabalho é avaliar algumas técnicas de mineração de dados (K vizinhos mais próximos, redes neurais artificiais e árvores de decisão) e comparar seu desempenho em amostras de tráfego real do Instituto Nacional de Pesquisas Espaciais (INPE), em São José dos Campos. Após a avaliação, será feita a escolha de um destes algoritmos, cujos resultados tenham se mostrado mais adequados (de acordo com certos parâmetros estabelecidos adiante neste trabalho) para minimizar a quantidade de falsos positivos e classificar *logs* em tempo hábil, reduzindo o conjunto a ser passado para o analista e promovendo meios de interpretação dos resultados. Passada a fase de análise, parte-se para a construção de um protótipo que se utilize

de perfis normais e suspeitos aplicados no treinamento do algoritmo escolhido, e que irá classificar os *logs* e apresentar um conjunto de sessões consideradas suspeitas para posterior análise humana.

Os perfis suspeitos serão criados através dos *logs* obtidos do *honeypot* do INPE (MONTES *et al.*, 2003), que é uma máquina que pode emular diferentes sistemas operacionais e serviços e para a qual foi atribuída uma faixa de endereços IP não anunciados para estudo de incidentes de segurança. Este tipo de perfil proposto pode ser realimentado constantemente, tanto com a adição de novos *logs* do *honeypot*, quanto com *logs* do tráfego da DMZ classificados como maliciosos por um sistema de detecção de intrusão por abuso, como o Snort[2] configurado com regras específicas para a rede em questão. Ao final, espera-se otimizar a tarefa de análise de *logs* a fim de diminuir o espaço de tempo compreendido entre a descoberta de uma atividade intrusiva e a tomada de contra-medidas.

1.2 Disposição deste Trabalho

Este trabalho está dividido em seis capítulos. No segundo capítulo, os problemas encontrados na análise de *logs* serão colocados, mostrando as implicações do grande volume de *logs* gerados em instituições de médio e grande porte no trabalho do analista de segurança. Será fornecido o embasamento sobre alguns conceitos necessários para a compreensão do trabalho e técnicas convencionais para análise de *logs* serão discutidas. Serão também apresentadas algumas dificuldades enfrentadas por estas técnicas.

No terceiro capítulo será feita uma revisão das técnicas de mineração de dados e algumas aplicações destas à análise de *logs* por grupos de pesquisa ao redor do mundo. Os resultados obtidos com estes trabalhos serão discutidos embasados pelas informações publicamente disponíveis de cada um dos projetos.

No quarto capítulo é apresentado o cenário com o qual trabalhou-se durante a dissertação, a metodologia utilizada para criar perfis de tráfego malicioso baseados nas tendências de ataques contra a DMZ do INPE, como procedeu-se para efetuar os testes com os algoritmos de mineração de dados, e os resultados provenientes da avaliação destes algoritmos. Ainda neste capítulo, será discorrida a construção de um protótipo para filtragem e redução automatizada de *logs* de tráfego de rede.

[2]Maiores informações sobre a ferramenta Snort estão disponíveis em <http://www.snort.org>, acessado em janeiro de 2007.

O quinto capítulo trata de um estudo de caso aplicando o protótipo em dados da DMZ do INPE e observando sua eficiência em reduzir a quantidade de *logs* que serão submetidos para análise, enquanto identifica atividades suspeitas na rede.

As conclusões extraídas deste trabalho são apresentadas no sexto capítulo, bem como direcionamentos para trabalhos futuros.

CAPÍTULO 2

DESCRIÇÃO DO PROBLEMA

Com o advento da Internet e disseminação de serviços via Web, empresas e pessoas passaram a trabalhar e a se organizar (economica e socialmente) de maneira diferente, criando uma sociedade virtual. Diversas características da dita sociedade real foram transpostas ao mundo virtual com algumas modificações, possibilitando uma grande interação sob a forma de *chats*, compras *on-line* e acesso a bancos virtuais.

Por outro lado, surgiram novos modos de se cometer crimes, alimentados pela facilidade de acesso à rede e baixo risco aos criminosos. Estes cibercriminosos se utilizam de técnicas não tão novas, derivadas de ilegalidades já há muito cometidas na sociedade tradicional estabelecida, tais como falsificação de identidade, atos de vandalismo, estelionato e roubo de informações, para obtenção de algum tipo de vantagem.

Na sociedade virtual, o ciberespaço, conceitos como privacidade são difíceis de serem mantidos, considerando o aspecto de acesso global trazido pela Internet. Este aspecto cria condições ideais para a ocorrência de crimes eletrônicos, pois permite que um atacante realize suas atividades de forma anônima e à distância, podendo inclusive burlar as leis de seu país, cometendo atividades ilícitas através de máquinas comprometidas em redes de países com leis menos restritivas. Além disso, os desdobramentos de uma invasão podem não ser imediatos, pois o invasor tem a possibilidade de implantar mecanismos de ataque, ocultá-los e controlá-los remotamente a qualquer tempo.

Entretanto, como qualquer crime, a realização de um ataque no ciberespaço deixa rastros que podem ser utilizados para eventualmente se chegar até o invasor, ou para analisar como o ataque foi efetuado. Estes rastros correspondem às atividades efetuadas pelo invasor em uma rede para alcançar seu objetivo, e armazenam informações diversificadas, tais como a data e a hora do ocorrido, o endereço IP de origem e de destino, serviços acessados/atacados, o conteúdo do fluxo de dados que resultou no comprometimento de uma rede de computadores ou de um sistema computacional, etc.

Toda esta interação via rede – legítima ou com intuito malicioso – gera dados que podem (e devem) ser registrados. O registro das informações de tráfego é chamado

de *logging* no jargão da computação, e consiste no armazenamento de dados relacionados à comunicação em rede das partes envolvidas. A atividade de *logging* pode ser útil no monitoramento de falhas ou na identificação de eventos que têm a possibilidade de ser ataques contra os serviços de rede.

O *logging* é feito por intermédio da coleta de *logs* de vários tipos (Seção 2.1.1), sendo esta atividade de vital importância para que o histórico dos eventos seja mantido. Nas próximas seções serão colocados assuntos relativos aos *logs*, seus tipos, problemas de coleta, armazenamento e análise, além de assuntos correlatos a esta última.

2.1 *Logs*

Entende-se por *log* um registro de transação ou auditoria que consiste de um ou mais arquivos do tipo texto ou em formatos específicos gerados por certas aplicações, que permite a um analista visualizar as atividades que ocorrem em seus sistemas ou redes. Os *logs* são gerados por dispositivos computacionais relacionados aos componentes de um sistema, tais como aplicativos, o sistema operacional ou dispositivos de *hardware*, bem como a interação entre sistemas formando redes de computadores. *Logs* podem representar avisos de atividades normais, alertas ou erros em algum dos componentes do sistema.

Embora a atividade de *logging* seja aparentemente trivial, vários outras aspectos devem ser levados em conta conforme apareçam mais serviços e com maior diversificação. Isto acarreta em uma quantidade maior de *logs* sendo gerados e no aumento da dificuldade de análise. Os aspectos a se considerar englobam a disponibilidade de espaço para armazenamento, a utilidade dos *logs* armazenados, a carga de trabalho do analista, a dificuldade do processo de análise e os requisitos de tempo para realização da tarefa de análise. Com exceção dos requisitos de espaço para armazenamento, que não vêm a ser um problema real nos dias de hoje, todos os outros quesitos citados são centrados no elemento humano. Em outras palavras, o processamento, a análise e as ações resultantes desta devem ser realizados por um especialista humano, que é inerentemente lento e custoso.

2.1.1 Tipos de *Logs*

Os dados ou mensagens de *log* podem variar conforme suas fontes geradoras, que vão desde o sistema operacional instalado, até a finalidade do sistema em si (servidor, computador de usuário, roteador), não sendo limitados a um sistema específico. A

seguir, apresenta-se uma lista adaptada de (SCHMIDT, 2006), contendo classes de sistemas ou dispositivos capazes de gerar *logs*:

- Sistema operacional;
- Aplicações;
- *Firewall*;
- Sistema de Detecção de Intrusão;
- Anti-vírus;
- Dispositivos de rede

Cada um destes tipos de *logs* tem o seu formato específico, definido pelo fabricante ou mantenedor do *software*/dispositivo gerador do *log*. Neste trabalho serão tratados os *logs* relativos ao tráfego da rede, também conhecidos por *dumps*. Todos os serviços e ferramentas utilizados são de código-fonte aberto e baseados em sistemas operacionais `Linux`. Nas subseções a seguir são descritos e exemplificados alguns dos tipos de *logs* acima mencionados.

2.1.1.1 *Logs* de Tráfego na Rede

Os *logs* do tráfego em uma rede, ou simplesmente *dumps*, contêm informações úteis para se identificar ocorrências relacionadas às conexões entre máquinas. Estas informações podem ser (mas não se limitam a) os endereços de rede das máquinas de origem e destino do tráfego, os serviços que serão executados durante a comunicação, os protocolos utilizados e os dados transferidos. Isto permite a reconstituição dos eventos de rede, podendo servir para se reconstruir partes de tráfego que antecederam um ataque, bem como o que ocorreu durante o mesmo (se foi bem-sucedido ou não, se houveram tentativas de conexão para outras redes com o objetivo de realizar varreduras, atacá-las ou obter ferramentas).

Um exemplo de *dump* pode ser observado na Figura 2.1, que representa um pacote de início de conexão e contém muitas informações importantes. Os campos são: *timestamp* (marca o tempo em que o pacote foi capturado pela ferramenta), protocolo da camada de rede, endereço IP e porta de origem, direção do tráfego, endereço IP e porta de destino, *flag* habilitada, número de seqüência, de `ACK` e *bytes* no pacote, tamanho da janela (win), e tamanho máximo do segmento (mss) e outras opções.

```
23:25:11.526803 IP 192.168.1.100.56017 > 192.168.1.1.80: S 313767689:3137
67689(0) win 5840 <mss 1460,sackOK,timestamp 3146475 0,nop,wscale 2>
23:25:11.529348 IP 192.168.1.1.80 > 192.168.1.100.56017: S 2712524221:271
2524221(0) ack 313767690 win 8192 <mss 1460,nop,wscale 0,nop,nop,timestamp
 2251583 3146475>
23:25:11.529400 IP 192.168.1.100.56017 > 192.168.1.1.80: . ack 1 win 1460
 <nop,nop,timestamp 3146476 2251583>
23:25:16.343467 IP 192.168.1.100.56017 > 192.168.1.1.80: P 1:4(3) ack 1
win 1460 <nop,nop,timestamp 3147679 2251583>
```

FIGURA 2.1 - Parte de um *dump* gerado pela ferramenta *tcpdump*.

2.1.1.2 *Logs* da Aplicação

Os *logs* das aplicações registram informações a respeito do funcionamento destas, tais como mensagens de início, finalização ou reinicialização de um serviço, acessos a recursos da aplicação e erros ocorridos. Muitas aplicações e ferramentas de segurança se utilizam das rotinas de geração de relatórios do próprio sistema operacional para registrar seus eventos.

Na Figura 2.2, pode-se ver um *log* da aplicação `ssh` contendo data e hora, *hostname*, o serviço e a mensagem. A primeira entrada corresponde a um registro normal, indicando que o processo (*daemon*) está executando em estado de escuta na porta 22 local. A segunda e terceira entradas correspondem a uma tentativa não autorizada de acesso, através de um ataque de força bruta por dicionário.

```
Oct  5 05:21:33 asgard sshd[2393]: Server listening on 0.0.0.0 port 22.
Oct  5 07:27:12 asgard sshd[3210]: Invalid user apple from X.Y.Z.132
Oct  5 07:27:12 asgard sshd[3210]: Failed password for invalid user apple
 from X.Y.Z.132 port 39427 ssh2
```

FIGURA 2.2 - Registro de eventos de aplicação.

2.1.1.3 *Logs* do Sistema Operacional

Estes *logs* contêm mensagens informativas sobre a inter-relação entre o sistema operacional, os componentes de *hardware* e os aplicativos em execução. Tentativas de acesso (local ou remoto), falhas de dispositivos e serviços ou mensagens do *kernel* do sistema são registradas neste tipo de *log*. Nos sistemas operacionais `unix-like` estes *logs* são armazenados no diretório `/var/log`.

Para exemplificar um *log* de sistema, na Figura 2.3 são mostradas mensagens do

kernel com a identificação do dispositivo de redes sem fio presente na máquina. As informações que este *log* provê são: data e hora, *hostname*, gerador da mensagem (neste caso o *kernel* do sistema), serviço ou dispositivo que disparou este evento (no exemplo, o *driver* da interface de rede sem fio "ipw2200") e a mensagem indicativa do evento.

```
1 Jan 15 19:30:11 asgard kernel: ipw2200: Intel(R) PRO/Wireless 2200/2915
2  Network Driver, 1.0.6
3 Jan 15 19:30:11 asgard kernel: ipw2200: Copyright(c) 2003-2004 Intel
4  Corporation
```

FIGURA 2.3 - *Log* do sistema representando a detecção de um dispositivo *wireless*.

2.1.1.4 *Logs* de Sistema de Detecção de Intrusão

Os *logs* dos sistemas de detecção de intrusão (IDS) em geral assemelham-se a *dumps* do `tcpdump`, uma vez que alguns IDSs são baseados na biblioteca de captura de pacotes `libpcap`[1]. Juntamente às informações do tráfego de rede, há um alerta associado que corresponde à identificação de uma possível tentativa de invasão. O identificador do alerta, seguido das informações contidas nos cabeçalhos dos protocolos da implementação da pilha TCP/IP podem ser verificados na Figura 2.4.

```
1 [**] [1:483:5] ICMP PING CyberKit 2.2 Windows [**]
2 [Classification: Misc activity] [Priority: 3]
3 08/01-03:37:36.117652 10.10.10.2 -> 192.168.20.29
4 ICMP TTL:121 TOS:0x0 ID:49936 IpLen:20 DgmLen:92
5 Type:8  Code:0  ID:16009  Seq:3967  ECHO
6 [Xref => http://www.whitehats.com/info/IDS154]
```

FIGURA 2.4 - Exemplo de *log* do sistema de detecção de intrusão "Snort".

2.1.2 Considerações sobre *Logs*

Haja vista a variedade de tipos de *logs*, não exaustivamente listada anteriormente, surgem os primeiros problemas. Dúvidas sobre quais *logs* deve-se priorizar, formas para sumarizá-los de modo a possibilitar a análise e experiência para discernir quais eventos são úteis e quais são descartáveis costumam levantar vários questionamentos, havendo maneiras diversificadas de se tratar cada caso.

[1]Maiores informações sobre o tcpdump e a libpcap podem ser obtidas em <http://www.tcpdump.org>, acessado em janeiro de 2007.

Entretanto, o principal problema relacionado à atividade de *logging* é a enorme massa de dados que tem de ser vasculhada por um analista em busca de situações anormais na rede ou nos sistemas sob sua responsabilidade. Além de o volume de *logs* ser, na maior parte dos casos, impossível de analisar, os analistas são constantemente sobrecarregados com outras atividades, incorrendo em decréscimo na efetividade da análise de *logs*. Isto faz com que esta tarefa não possa ser completada em um intervalo de tempo razoável e, por conseqüência, torna a implementação de contra-medidas ineficiente no que diz respeito à minimização ou mitigação de incidentes de segurança em redes de computadores e sistemas de informação.

Isto posto, embora os *logs* tenham funções peculiares e bastante úteis para auxiliar na descoberta de problemas de vários tipos e na administração de redes e sistemas, neste trabalho serão focados aqueles que possuem alguma ligação com aplicações no campo da segurança da informação. Os *logs* aqui abordados têm sua utilidade ao auxiliar o analista a identificar tentativas de ataque, ou eventos maliciosos que porventura tenham ocorrido através de uma rede de computadores e possam indicar incidentes de segurança.

2.2 Ataques a Sistemas Computacionais

Os três requisitos fundamentais para a segurança de um sistema de informação são: confidencialidade, integridade e disponibilidade (BISHOP, 2002). Um ataque corresponde a uma tentativa de violação de um destes requisitos, onde o acesso não autorizado a uma informação não pública implica na perda de confidencialidade; a modificação de informações corrompe os dados verdadeiros, incorrendo na perda de integridade; e a indisponibilidade é decorrente da destruição de informações ou do esgotamento de recursos do sistema.

O objetivo de um invasor é obter acesso a sistemas para realizar atividades indevidas, como roubo ou destruição de dados, ataques a outros sistemas e apropriação de identidades. Uma representação gráfica das etapas correspondentes a um ataque pode ser vista na Figura 2.5.

Como visto na Figura 1.1, o passar dos anos tornou os ataques mais elaborados enquanto cada vez menos conhecimento é requerido dos invasores. Os *hackers* descobrem vulnerabilidades em sistemas, protocolos e *software* e constróem ferramentas para explorá-las (*exploits*). Muitas destas ferramentas são disponibilizadas na Inter-

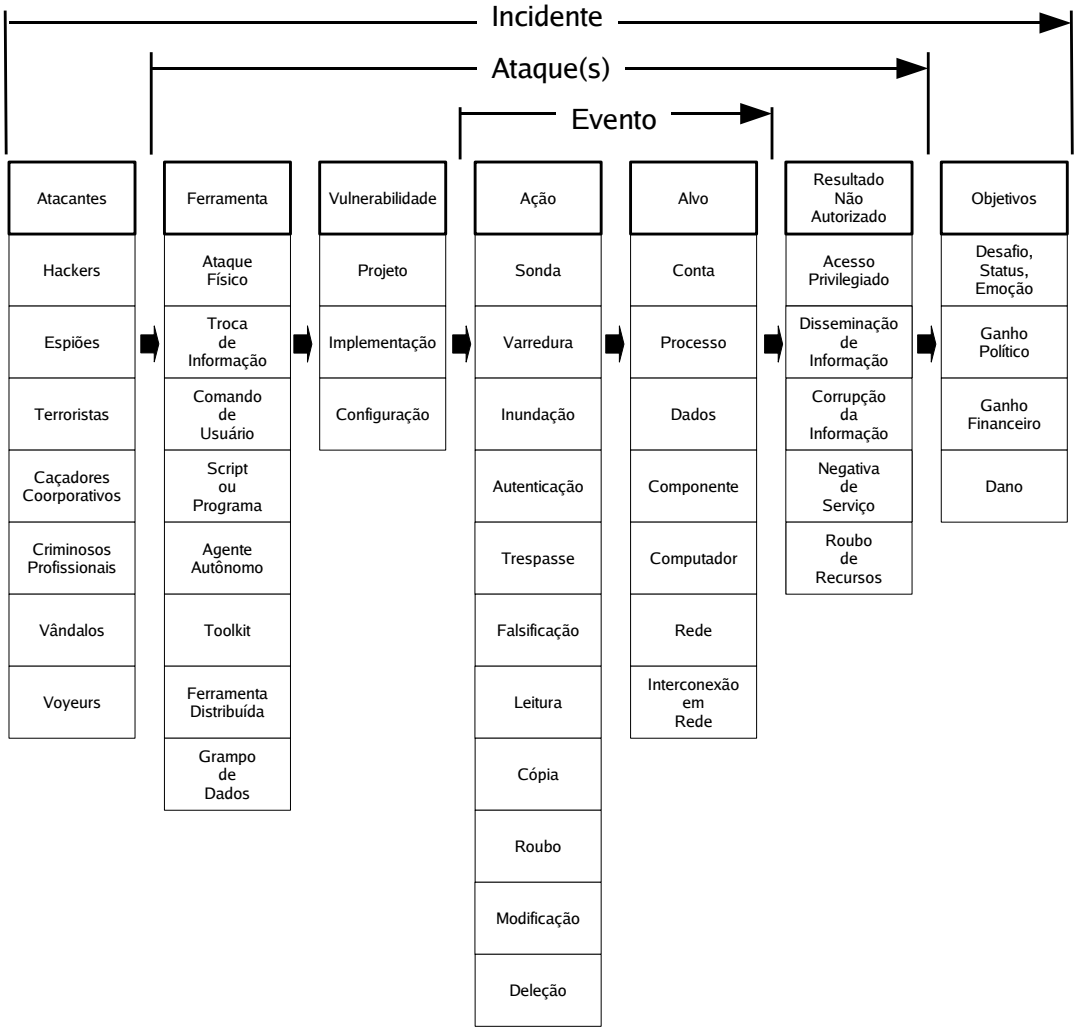

FIGURA 2.5 - Passos e objetivos de um ataque a computadores e redes.
Fonte: Adaptado de (HOWARD; LONGSTAFF, 1998).

net, ficando ao alcance de qualquer indivíduo mal intencionado.

Pessoas de má índole, por sua vez, tentam efetuar inúmeros ataques através do lançamento de *malware* na rede sem alvos específicos, tais como os vírus e *worms*, com o intuito de apropriação de outros sistemas para servir de trampolim (*stepping stone*) e formar redes de máquinas controladas remotamente, com o objetivo de varrer outras máquinas e realizar ataques de forma anônima (*botnets*).

Outra maneira é subverter mecanismos como o correio eletrônico (*spam e phishing*), visando o lucro através de propaganda não solicitada ou infecção das máquinas dos usuários para instalação de capturadores de tecla. Os ataques citados referentes à fraudes e *worms* compreendem uma porcentagem altíssima dos incidentes de segurança reportados para o CERT, como pode ser visto na Figura 2.6.

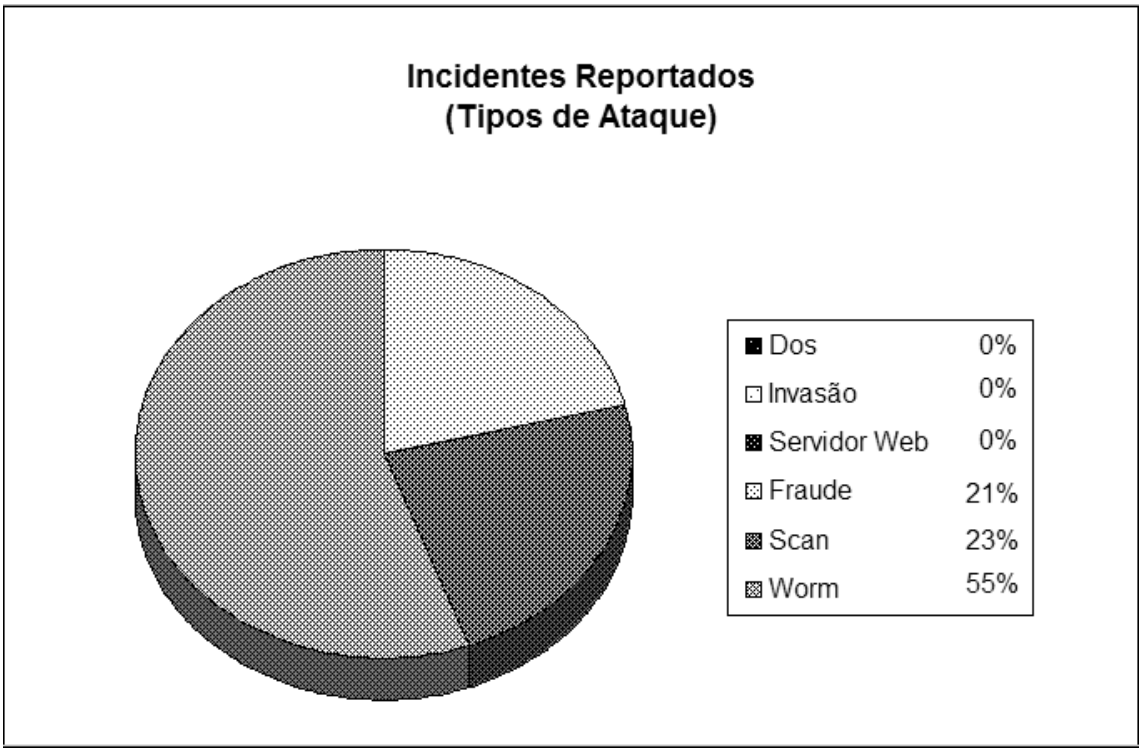

FIGURA 2.6 - Incidentes reportados ao CERT.br entre janeiro e dezembro de 2006.
Fonte: Adaptado de (CERT.BR, 2007a).

Apesar de todos os esforços voltados para a detecção de atividades maliciosas em redes de computadores, os ataques estão cada vez mais freqüentes e geralmente não possuem alvos específicos. Desta forma, tem-se uma gama maior de vítimas em

potencial, as quais servem para a aquisição de novos *stepping stones* para mascarar a origem de novos ataques, bem como para a formação de novas *botnets*, as quais podem ter um ou mais direcionamentos de alvo, após controladas.

Em paralelo, o número de incidentes reportados também tem aumentado a cada ano, mostrando que os administradores e analistas estão ficando mais conscientes com relação aos acessos indevidos a seus sistemas ou redes. Porém, uma quantidade ínfima destes relatos tem a ver com invasões, o que leva à inferência de que, ou estas não estão sendo reportadas por motivos de risco à imagem/displicência, ou não estão sendo percebidas (fato este que está diretamente ligado à análise adequada dos *logs*). Assim sendo, a detecção e conseqüente reporte de um incidente ocorre por meio da análise dos *logs*, o que torna clara a necessidade da rapidez e precisão na tarefa.

2.3 O cenário

Como visto na subseção 2.1.1, *logs* podem ser provenientes de diversas fontes e com diversas funcionalidades. Neste trabalho, o foco será dado aos *logs* de tráfego de rede. Ataques contra redes podem ocorrer sob a forma de pacotes de tráfego maliciosamente formados, interações ilegais com o sistema operacional na tentativa de subverter alguma funcionalidade ou explorar possíveis vulnerabilidades, e/ou acesso não autorizado à recursos e aplicações. Para os fins desta dissertação, apenas as informações derivadas de *logs* de tráfego de rede serão consideradas.

Os *logs* coletados para a análise são provenientes da rede do Instituto Nacional de Pesquisas Espaciais (INPE), situado na cidade de São José dos Campos, estado de São Paulo. O INPE é formado por departamentos, os quais podem ou não possuir uma zona desmilitarizada (DMZ) em sua rede interna local (LAN) com serviços próprios de *e-mail*, armazenamento de arquivos, *site* departamental, etc. Entretanto, há a DMZ institucional, que provê serviços como o *webmail*, cuja finalidade é a de unificar o acesso dos funcionários a seus respectivos servidores de *e-mail* por um único ponto de acesso externo; a resolução de nomes autoritativa para o domínio "inpe.br"; a página Web principal do Instituto, o servidor de `ssh` que permite acesso externo controlado à determinadas LANs, entre outros.

Os muitos serviços remotamente acessíveis da DMZ do INPE, são os mais visíveis e também os mais visados. Estes serviços estão intimamente ligados com a imagem do Instituto perante a comunidade em geral, pois permite que pesquisadores e

funcionários acessem os recursos do INPE de qualquer lugar, provêem informações relativas à previsão de tempo, servem aos alunos de pós-graduação, entre outras utilidades. Qualquer ataque que acarrete em negativa de serviço, descaracterização de páginas Web (*defacement*), exposição de informações sensíveis internas, ou uso da rede do Instituto para realização de ataques ou armazenamento de conteúdos ilegais pode causar situações desagradáveis (e até mesmo judiciais) envolvendo a imagem do INPE.

Com isso, é necessário que se viabilize a análise de *logs* de maneira efetiva para resguardar as informações sensíveis e proteger os recursos disponíveis do INPE. Todos os estudos feitos neste trabalho envolvem *logs* da DMZ do INPE, armazenados na forma de *dump* em um coletor para análise posterior. Há também um *honeypot* (Seção 2.4.3) na mesma rede que pertence ao Projeto Honeypots Distribuídos, parte do Consórcio Brasileiro de Honeypots (CBH) que monitora o espaço da Internet nacional e conta com mais de 30 participantes com *honeypots* espalhados pelo país[2]. Este *honeypot* também armazena *logs* de acesso em formato `pcap`[3]. A topologia simplificada da DMZ do INPE pode ser observada na Figura 2.7.

O volume de *logs* de tráfego armazenado no coletor diariamente costuma variar entre 300MB e 1 GB de dados para dias normais, que é uma quantidade inviável para análise manual. Esta quantidade refere-se apenas ao tráfego externo que envolve diretamente os servidores da DMZ do INPE. Ao se considerar todo o tráfego que passa pela DMZ, pode-se chegar a mais de 4GB de *dumps* diários. Além disso, uma grande parcela destes *logs* está associado a tráfego inofensivo, ou seja, acessos normais de usuários, comunicação de determinados protocolos entre máquinas da rede ou *scripts* automatizados para realizar tarefas de rotina. Uma outra parcela dos *logs* está relacionada a varreduras de sistemas (tentativa de localização e enumeração de serviços) e ferramentas maliciosas que tentam explorar vulnerabilidades, efetuar ataques de força bruta contra alguns serviços com autenticação, acessar serviços e recursos indiscriminadamente (mesmo que não existam), sendo ataques não direcionados ou não dirigidos a alvos do Instituto.

Assim, pode-se perceber que a maioria dos *logs* gerados consiste basicamente de

[2]Mais informações sobre o Projeto Honeypots Distribuídos e o Consórcio Brasileiro de Honeypots podem ser obtidas em <`http://www.honeypots-alliance.org.br`>, acessado em janeiro de 2007.

[3]O formato pcap está associado aos arquivos de dumps gerados por ferramentas que utilizam a libpcap.

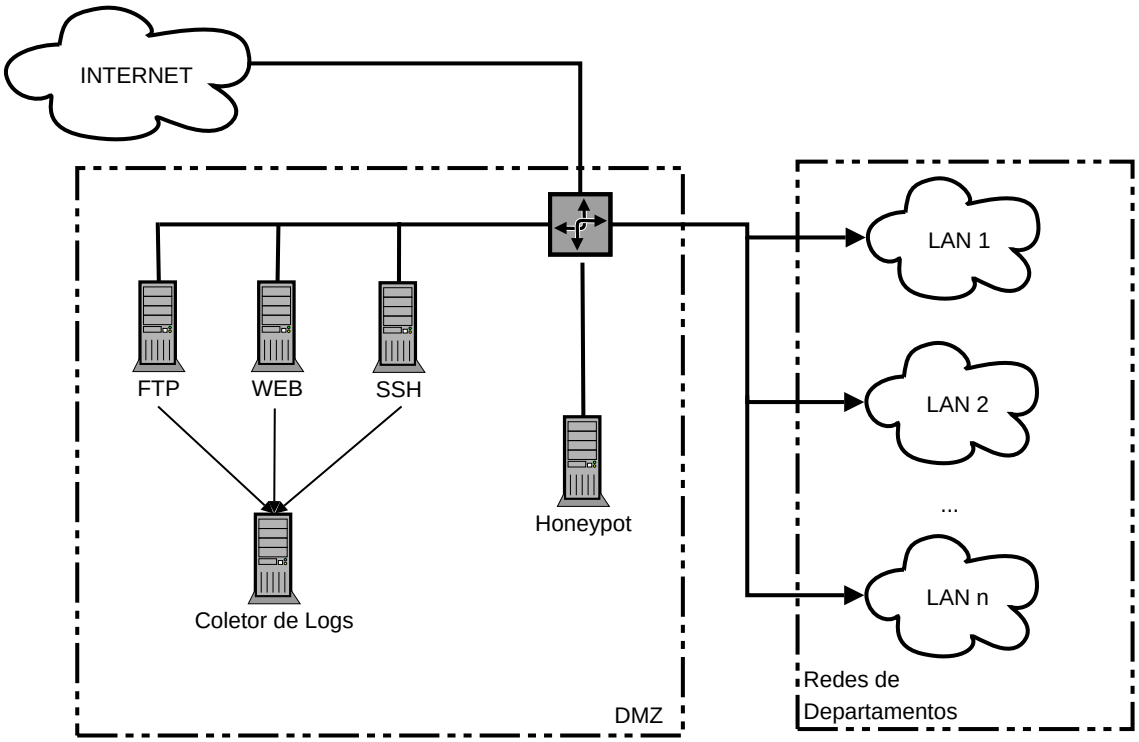

FIGURA 2.7 - Topologia simplificada da DMZ do INPE.

ruído – tráfego de rede conhecido, legítimo ou inofensivo que não provê informações úteis para a busca de situações envolvendo segurança de redes – interferindo no bom andamento da análise. O problema é que o analista precisa saber distinguir entre tráfego de interesse e ruído, ficando esta tarefa dependente de sua experiência. No entanto, esta pré-análise consome um certo tempo e, mesmo que se utilizem ferramentas para automatizar o processo, a confecção e configuração das mesmas pode não ser adequada para remover o ruído de maneira rápida e eficaz.

Desta forma, nota-se que há uma necessidade urgente de técnicas inteligentes e automatizadas para auxiliar o analista a realizar tarefas envolvendo filtragem de *logs* por eventos de segurança. Algumas técnicas que podem ser utilizadas para esta finalidade são descritas na próxima seção.

2.4 Abordagens tradicionais para análise de *logs*

Existem diversas ferramentas para processamento de *logs*, para geração de diferentes tipos de *log* e de relatórios baseados nestes. O objetivo destas ferramentas é a recuperação de informações úteis provenientes dos *logs* para auxiliar na administração

de sistemas ou redes.

Entretanto, obter informações que levem à resultados efetivos é um problema devido justamente à grande verborragia dos sistemas geradores de *logs*. Enquanto todas as atividades são registradas de várias maneiras e em diferentes níveis (rede, aplicação, sistema operacional), permitindo a reconstrução dos eventos, o grande problema é encontrar os eventos que realmente importam.

Serão apresentadas algumas técnicas e ferramentas que podem ser úteis na filtragem e apresentação dos *logs*, para que o analista possa visualizar melhor os eventos de suas redes e sistemas. A finalidade destas é permitir que o analista interprete de maneira mais fácil a grande massa de dados coletada.

2.4.1 Filtragem

A filtragem (*whitelisting*) – ou ignorância artificial (RANUM, 1997) – é uma técnica que consiste em se passar *streams* de eventos (por exemplo um *log*) por diversos filtros. Cada um destes filtros corresponde a um determinado padrão já conhecido pelo analista, o que faz com que tais entradas conhecidas sejam eliminadas. Os eventos restantes são considerados novos ou suspeitos, servindo de base para a criação de novos filtros.

Com isso, os filtros são constantemente refinados e a tendência é que a quantidade de falsos positivos seja reduzida com o passar do tempo. Outra vantagem da filtragem é que, como técnica de redução ela torna factível a análise de muitos eventos. Entretanto, por trabalhar com correspondência de expressões regulares, filtros muito específicos podem ser difíceis de gerar. Além disso, ataques que consistem de sucessivos eventos normais usualmente passam desapercebidos.

A técnica de filtragem descrita por Ranum e denominada ignorância artificial é comumente utilizada para redução e priorização de *logs* de sistema e aplicações, embora o conceito possa ser derivado para *logs* de tráfego de rede através da busca por expressões regulares com ferramentas como `ngrep`[4].

[4]Informações sobre ngrep podem ser encontradas em <http://ngrep.sourceforge.net>, acessado em janeiro de 2007.

2.4.2 Análise de fluxos

Segundo (CARVALHO, 2005), *um fluxo é identificado como uma seqüência unidirecional de pacotes entre um dado par origem-destino, ambos definidos pelo endereço IP na camada de rede e pelos números de porta na camada de transporte.* Para manipular dados de fluxos, mais especificamente de `NetFlow`, existe o conjunto `Flow-tools`[5], que dentre outras funcionalidades permite a geração de relatórios sobre estes dados.

Os fluxos de rede constituem um grande volume de informações e sua análise minuciosa pode trazer resultados rápidos e eficazes na administração de uma rede. Vale ressaltar que a análise de fluxos é bastante utilizada na detecção de intrusão por anomalia, por permitir comparações entre o perfil de tráfego de uma rede e as atividades de rede atuais, processo este que identifica imediatamente situações incomuns de utilização dos recursos da rede. Nestas situações incluem-se *worms*, programas que se conectam a redes de distribuição de conteúdo (*peer-to-peer*) e tentativas de se realizar negativas de serviço (*DoS*).

O uso combinado de algumas ferramentas da coleção `Flow-tools`, como `flow-cat`, `flow-nfilter` e `flow-stat` traz resultados interessantes na detecção de possível comportamento inadequado por parte dos usuários de uma determinada rede. Na Figura 2.8, é mostrado um exemplo de relatório com a contagem da quantidade de diferentes endereços IP de destino acessados a partir de *hosts* internos. Através deste tipo de relatório pode-se controlar melhor as atividades das máquinas da rede interna, uma vez que sob condições normais um usuário não consegue acessar milhares de endereços em um único dia de serviço.

2.4.3 *Honeypots*

Um *honeypot* (SPITZNER, 2003) é um recurso de segurança cujo valor reside em ser sondado, atacado e comprometido, enquanto coleta dados a respeito das varreduras e ataques. Isto é feito para que se possa analisar as atividades intrusivas que porventura sejam efetuadas contra os *honeypots*. A implementação de *honeypots* tem por finalidade a compreensão das técnicas utilizadas por um invasor na realização de ataques a redes de computadores. Os *honeypots* podem ser classificados basicamente da seguinte maneira (PROJECT, 2004):

[5]Maiores informações sobre as flow-tools podem ser obtidas em <`http://www.splintered.net/sw/flow-tools/docs/flow-tools.html`>, acessado em janeiro de 2007.

```
192.168.2.4         18857
192.168.3.1         16952
192.168.28.5         16580
192.168.34.50         15091
192.168.35.1         14593
192.168.27.8         14039
192.168.1.120         13755
192.168.12.201         12753
192.168.28.1         11773
192.168.1.210         11010
192.168.30.1         8723
192.168.13.5         7937
192.168.1.22         7360
192.168.2.5         7002
192.168.6.1         6990
192.168.16.1         6778
192.168.12.3         6035
192.168.40.1         5867
192.168.2.11         5695
192.168.13.245         5525
192.168.27.69         4932
192.168.25.4         4914
192.168.21.2         3487
192.168.13.1         3476
192.168.34.1         3442
192.168.13.242         3430
192.168.3.149         3147
192.168.22.1         3051
192.168.13.3         2562
192.168.2.34         2469
192.168.2.25         2427
192.168.5.11         2278
192.168.38.52         2189
192.168.16.88         2177
192.168.16.64         2127
192.168.43.1         2103
192.168.28.2         1678
192.168.28.6         1578
192.168.15.65         1336
192.168.12.8         1335
```

FIGURA 2.8 - Contagem de IPs de destino acessado por *hosts* internos.

- ***Honeypots* de baixa interatividade**, os quais têm a capacidade de emular uma rede completa de computadores em uma única máquina, com sistemas operacionais e serviços diversos;

- ***Honeypots* de alta interatividade**, que são máquinas com sistema operacionais funcionais, aplicativos instalados de fato e serviços reais oferecidos remotamente, permitindo uma maior interação entre o invasor e os computadores/serviços atacados, e coletando dados destas interações.

Os *honeypots* de baixa interatividade recebem grande quantidade de acessos provenientes tanto de máquinas externas quanto de máquinas da rede interna. Como o *honeypot* não é divulgado e nem provê serviços reais para a rede, qualquer acesso registrado é considerado ilegítimo. Desta forma, podem-se utilizar os *honeypots* para identificar máquinas da rede interna que estão mal configuradas ou infectadas por algum tipo de *malware*, ou mesmo usuários comentendo atos não condizentes com a política de segurança da instituição (GRÉGIO *et al.*, 2005).

Além disso, os acessos provenientes de máquinas da Internet refletem as tendências de ataque do momento, como por exemplo, um *worm* novo efetuando varreduras e/ou tentativas de exploração. A observação das estatísticas e sumários gerados por *logs* de um *honeypot* de baixa interatividade é muito útil para o analista se preparar melhor para os eventos ocorrendo em sua rede e descobrir atividades ilícitas.

2.4.4 Sistemas de Detecção de Intrusão

Os IDSs, como o próprio nome indica, têm por função avaliar *logs*, encontrar tráfego ou eventos suspeitos e emitir alertas. Isto porque a detecção de intrusão é baseada na hipótese de que existem diferenças suficientes entre o comportamento legítimo e intrusivo, para que o tráfego de rede referente a estes possa ser separado (STALLINGS, 2002). Um IDS pode ser classificado da seguinte maneira:

- *Host-based IDS*, ou HIDS, que são comumente encontrados em servidores ou estações de trabalho e monitoram *logs* de eventos e do sistema. São utilizados para detecção de abuso por parte de usuários maliciosos internos, os chamados *insiders* (ou invasores que conseguiram evadir os métodos de detecção da rede) e podem incorporar análise de assinaturas e heurísticas para maximizar a detecção (CUFF, 2003). Um exemplo de HIDS é o Snare[6];

- *Network-based IDS*, ou NIDS, correspondem a sensores situados em pontos estratégicos de uma rede, monitorando um determinado segmento para

[6]Snare: <http://www.intersectalliance.com/projects/Snare/>.

capturar o tráfego em fluxo em busca de atividades anômalas ou intrusivas. Exemplos de NIDS incluem o Snort, o Bro[7] e o Shadow[8];

- *Hybrid IDS*, que é a combinação de NIDS e HIDS para formar um conjunto com as características de um IDS baseado em redes no nível da máquina, reduzindo o problema da detecção de intrusão em redes comutadas. Um sistema de detecção de intrusão híbrido é o Prelude[9].

O modo de operação de um IDS baseado em rede pode ser classificado como por abuso ou por anomalia. No primeiro caso, tem-se uma base de dados de assinaturas de ataque (cadeias de caracteres, comandos ou informações dos cabeçalhos de pacotes de rede) que é utilizada para comparação com os *logs* do tráfego da rede, emitindo alertas se houver correspondência entre estes e uma ou mais assinaturas.

Na detecção por anomalia, métodos estatísticos são utilizados na construção de perfis de tráfego normal da rede, ligado ao comportamento dos usuários conforme um padrão temporal (horário de expediente, dia da semana). As atividades que diferirem deste perfil normal são então consideradas tráfego anômalo, o que pode ser um indicativo de intrusão.

2.5 Conclusão

Como pôde ser observado, a quantidade de *logs* é um fator crucial para a realização da tarefa de análise. O processo, se feito manualmente e sem o auxílio de ferramentas, é impossível de ser completado, pois a carga de trabalho é muito grande e o volume de dados contém uma quantidade enorme de *logs* sem utilidade na análise voltada para a segurança.

Uma estatística básica pode ser feita para verificação do problema da análise de *logs*: em um dia normal, o servidor Web do INPE contabiliza cerca de 100000 sessões. Como a imensa maioria destas sessões diz respeito a tráfego legítimo, um analista as analisa superficialmente e as descarta quase que imediatamente.

Algumas sessões podem conter pacotes estranhos que desacelerem o trabalho do analista. Um número mínimo de sessões que contenham tráfego completamente desconhecido fazem com que o analista pare por vários minutos para fazer uma análise

[7]Bro: <bro-ids.org>.
[8]Shadow: <http://www.nswc.navy.mil>.
[9]Prelude: <http://www.prelude-ids.org>.

minuciosa e identificar o que ocorreu. Supõe-se então que o tempo médio para analisar cada sessão seja de 30 segundos.

Assim, dado um *log* correspondente a um único dia de tráfego Web, um analista demoraria aproximadamente 833 horas (ou quase 35 dias) para analisá-las todas. Considerando um dia de serviço de 8 horas, seriam necessárias 104 pessoas para fazer a análise destes dados em um dia inteiro, o que ainda não é um intervalo de tempo adequado.

Dadas estas informações, nota-se que, devido ao fator tempo ser também primordial para que se possa identificar e responder adequadamente a incidentes de segurança, são necessárias ferramentas para tornar factível a análise de *logs*. Algumas técnicas e ferramentas utilizadas para o auxílio ao analista de segurança foram vistas neste capítulo, como filtragem, análise de fluxos, *honeypots* e sistemas de detecção de intrusão.

A técnica da filtragem consegue obter resultados muito bons no que diz respeito à priorização e sumarização de *logs*, mas isto demora um certo tempo até que se adeque os filtros para o padrão de comportamento dos eventos, além de ser completamente dependente do conhecimento do analista. Ainda que ajude na descoberta de novos eventos e, por conseqüência, novos filtros, diminuindo cada vez mais a quantidade de falsos alertas, é uma técnica que precisa de observação constante, não eliminando a probabilidade de que muitos falsos negativos ocorram.

A análise de fluxos também é bastante interessante para detectar anomalias no tráfego. Com a confecção de *scripts* simples pode-se facilitar a tarefa de análise diária, priorizando os eventos em que houve geração absurda de tráfego (quantidade de pacotes), trânsito excessivo de dados (quantidade de *bytes*), ou número inadequado de acessos por máquina (milhares de endereços IP acessados por uma única pessoa).

Com essa abordagem é possível identificar tentativas de causar indisponibilidade de serviço, disseminação de arquivos em redes de compartilhamento (P2P), máquinas infectadas fazendo varreduras ou tentando explorar outras máquinas, etc. Porém, o número de falsos alertas também é grande. Por exemplo, usuários com máquinas atrás de *Network Address Translation* (NAT) saindo por um mesmo endereço IP, podem gerar alertas que levem à tomada de medidas de bloqueio desnecessárias devido a algum tipo de controle existente na rede, ou ainda, usuários fazendo *download*

de imagens de sistemas operacionais (.iso) ou imagens grandes, como as de satélite, acabam por levantar suspeitas infundadas.

Quanto aos *logs* de *honeypots*, todos os acessos registrados são considerados maliciosos devido à natureza destes dispositivos. Desta forma, os *honeypots* podem ser utilizados como tecnologia de detecção de intrusão. Um aspecto bastante interessante dos *honeypots* é que eles funcionam como sensores de tendências do estado de criticalidade de uma rede específica.

Quando um *worm* é lançado e inicia o seu procedimento para propagação, ele certamente irá acessar algum *honeypot*, deixando seu tráfego registrado. Isto dá inúmeras possibilidades aos analistas, desde o desenvolvimento de *scripts* que emulam o protocolo com o qual o *worm* conversa (*listeners*) para entender seu mecanismo de funcionamento, até a implementação de regras de controle de acesso em roteadores ou *firewall*. Entretanto, um *honeypot* sem acessos não possui valor algum.

Os IDSs por anomalia são úteis para detecção de eventos similares àqueles citados na análise de fluxos e por isso, sofrem das mesmas desvantagens. Um outro fato a ser apontado é que este tipo de IDS funciona de modo adequado em redes cuja distribuição de tráfego é comportada. Em redes com comportamento de tráfego muito entrópico, a geração de falsos positivos é extrema, por vezes inviabilizando o uso da ferramenta. Ataques que não implicam em mudança no perfil de tráfego passam desapercebidos.

No caso dos IDSs por abuso, é necessário que se façam ajustes finos na base de dados de assinaturas de ataque, para que não haja quantidade extensa de alertas erroneamente gerados em situações normais. Estes sistemas possuem uma taxa não mensurável de falsos negativos, uma vez que se o ataque não estiver codificado na base de dados, nenhum alerta será emitido. Um outro ponto a se considerar no caso de IDS é o local na rede em que o sensor será colocado, pois dependendo do ponto da rede pode haver uma sobrecarga de processamento e parte do tráfego não ser analisada devido ao descarte de pacotes.

Como esperado, todas as técnicas e ferramentas apresentadas possuem pontos fortes e fracos. Apesar de não haver dúvidas quanto a ajuda que estas técnicas fornecem aos analistas, quando implementadas sozinhas, muitas vezes as desvantagens destas (principalmente se mal configuradas) acabam por ofuscar sua real utilidade. Entre-

tanto, um analista de segurança com conhecimentos adequados combinando algumas destas técnicas pode tirar vantagem das mesmas, facilitando sua tarefa.

A combinação mais eficiente é aquela que diminui a quantidade de *logs* para análise em tempo hábil, reduzindo também a quantidade de falsos positivos. Um dos requisitos para isto ocorrer é que o processo seja o mais automatizado possível, para que o analista não perca muito tempo na pré-análise e sim na análise propriamente dita.

No próximo capítulo, serão apresentados conceitos de mineração de dados e será feita uma revisão das técnicas e metodologias que vêm sendo utilizadas para o tratamento e classificação de *logs* (usando mineração de dados) nestes últimos tempos.

CAPÍTULO 3

REVISÃO DO ESTADO DA ARTE

3.1 Introdução

O registro de eventos tem por finalidade manter um histórico das atividades ocorridas em um sistema computacional. A análise destes registros pode fornecer informações que auxiliem na administração e manutenção das redes, tais como a geração de estatísticas de acesso a recursos ou utilização de serviços, identificação de usuários e atividades realizadas por estes, mau funcionamento de componentes do sistema ou problemas de comunicação em rede, bem como anormalidades. Neste trabalho, um dos objetivos é o de descobrir comportamentos suspeitos de sistemas ligados em redes de computadores através da análise de *logs*, enfocando aspectos de segurança.

A análise de *logs* em busca de padrões que caracterizem violações em computadores ou redes é o ponto chave na construção de sistemas de deteção de intrusão. Algumas técnicas e abordagens gerais que podem ser utilizadas para se realizar detecção de intrusão por abuso ou por anomalia podem ser vistas em (VERWOERD; HUNT, 2002). A eficácia de um IDS depende de atualizações e ajustes constantes, seja da base de dados de assinaturas ou dos perfis de tráfego criados, o que tem causado um aumento na utilização de técnicas de mineração de dados para a obtenção de melhores resultados, principalmente nos IDSs por anomalia.

Como relatado anteriormente, é importante classificar *logs* e reduzir o número de falsos positivos, para que a análise possa ser feita de forma mais eficiente. Abordagens utilizando mineração de dados estão sendo pesquisadas e aplicadas à segurança de computadores (CHAN *et al.*, 2003) nos últimos tempos, fornecendo resultados satisfatórios. Algoritmos de mineração de dados aplicados em *logs* de segurança automatizam a tarefa de filtragem, reduzindo a carga de trabalho do analista.

Embora a mineração de dados não seja a panacéia que muitos apregoam, seu uso combinado com alguma abordagem tradicional de análise de *logs* pode, através da exploração dos pontos fortes de cada uma, trazer resultados bem melhores do que a utilização das abordagens em separado. Nas próximas seções serão apresentados conceitos, algumas técnicas de mineração de dados que podem ser empregadas na análise de *logs* e pesquisas que têm sido desenvolvidas para a busca de padrões suspeitos com mecanismos baseados em técnicas de mineração de dados.

3.2 Técnicas de Mineração de Dados

Mineração de Dados ou *Data Mining* é uma das etapas em processos de descoberta de conhecimentos em bancos de dados (*Knowledge Discovery in Databases ou KDD*). Uma descrição simplificada deste processo é "processo geral de descoberta de conhecimentos úteis (previamente desconhecidos) a partir de grandes bancos de dados" (adaptado de (FAYYAD *et al.*, 1996)).

Alguns pontos desta definição são questionáveis: os conhecimentos podem não ser estritamente previamente desconhecidos e os dados a ser analisados não precisam necessariamente estar organizados em bancos de dados. O conceito de "grandes" bancos de dados também é questionável; podemos ter bancos de dados com poucos registros mas com estruturas extensas ou complexas, que não poderiam ser facilmente analisáveis de outras formas.

O processo de descoberta de conhecimento é composto de várias etapas independentes, e que nem sempre têm distinções claras entre si. O sub-processo de mineração de dados, em particular, pode ser composto pelo uso e análise de resultados de várias técnicas estatísticas, de inteligência artificial e reconhecimento de padrões, de modelagem de bancos de dados, heurísticas, computação gráfica e visualização, computação de alto desempenho, etc. Outro subprocesso interessante é o de visualização, que pode tanto permitir a inferência sobre uma estrutura ou organização nos dados a ser analisados quanto apresentar resultados de análise de forma clara.

O processo de descoberta de conhecimento envolve o levantamento de requisitos e necessidades, e estudos iniciais sobre o problema a ser atacado, sendo composto por algumas fases:

a) Compreender o domínio da aplicação e entender as expectativas do usuário final do processo.

b) Criar/selecionar uma coleção de dados para aplicação.

c) Pré-processar e limpar os dados (eliminar ruídos e dados irrelevantes).

d) Transformar (reduzir e reprojetar) os dados (encontrar atributos úteis e interessantes).

e) Escolher a tarefa, métodos, modelos, parâmetros etc. do processo de mineração de dados e executar este processo.

f) Interpretar os resultados, iterar se necessário.

g) Consolidar o conhecimento adquirido, resolver conflitos, iterar se necessário.

É muito importante ressaltar o aspecto exploratório de mineração de dados: é necessário o conhecimento de um especialista nos dados sendo analisados e algum conhecimento das técnicas de mineração de dados para sugerir a melhor abordagem para a análise dos dados. Mesmo assim pode ser que a "melhor abordagem" só seja descoberta depois de muitas tentativas com os diversos algoritmos e parâmetros existentes. Em outras palavras, dependendo do problema a ser tratado e da qualidade dos dados, nem sempre os resultados serão confiáveis e consistentes.

Técnicas de mineração de dados têm sido usadas com sucesso em várias áreas científicas e comerciais. As aplicações destas técnicas têm sido motivadas parcialmente pelo aumento no poder de processamento de computadores, que permite a aplicação de algoritmos mais complexos para análise de dados; e parcialmente pelo aumento na capacidade de coleta e armazenamento de dados.

Alguns exemplos de aplicações de mineração de dados envolvem conjuntos de dados da ordem de terabytes (FAYYAD *et al.*, 1996; PIATETSKY-SHAPIRO, 2003) ou mesmo petabytes, e são apresentados resumidamente a seguir.

- **Amazon.com** (SAS, 2006): melhoria da customização da interface com o usuário (melhoria de vendas por indicação), eliminação de fraudes.
- **1-800-FLOWERS.com** (SAS, 2006): compreensão e antecipação de comportamento de clientes, descoberta de tendências e explicação de observações.
- **U.S. Census Bureau** (SAS, 2006): análise de dados espaciais (com SAS e software da ESRI) de ensino público para determinar políticas para melhoria na educação.
- **Highmark** (SAS, 2006): detecção de fraudes em planos de saúde.
- **Japan Credit Bureau** (SAS, 2006): melhoria da resposta a campanhas de marketing, retenção de clientes, identificação de novos segmentos de mercado.

- **Segmentação de clientes de serviços bancários** (ADRIAANS; ZANTINGE, 1996): uso de agrupamento e árvores de decisão para verificar que clientes são parecidos entre si, e porquê. A segmentação do banco de dados de clientes permite análise do comportamento dos clientes, das políticas e promoções do banco e da eficácia do *marketing*.

- **Columbia Interactive/Columbia University** (SAS, 2006): Análise de visitas a sites, coletando "trilhas" de usuários (como usam o site, que páginas são mais atraentes para usuários, quando usuários deixam o site) para melhorar interatividade e planejar conteúdo.

- **Casino** (SAS, 2006): cadeia com 115 hipermercados, 400 supermercados, mais de 4000 lojas e 260 lanchonetes. Criou programa de cartões de fidelidade e tem coletado dados dos cartões e hábitos de consumo.

- **TIM (Telecom Italia Mobile)** (SAS, 2006): redução de *churn*[1], análise de comportamento do usuário e segmentação do banco de dados de usuários.

- **Verizon Wireless** (DAS, 2003): redução de *churn* de 2 para menos de 1.5 por cento: de 34.6 milhões de usuários, aproximadamente 170.000 foram retidos.

- **argonauten360°** (STATSOFT, 2006): Consultoria alemã para empresa de telecomunicações que provê serviços "call-by-call" de telefonia móvel: custo baixo ou zero para serviço básico, tarifa por uso. Estudou volume de tráfego minuto a minuto e identificou possíveis faixas de uso onde melhor competitividade podia ser alcançada.

- **IMS America** (WASSERMAN, 2000): Uma das maiores empresas de pesquisa de mercado farmacêutico do mundo, mantém um banco de dados de 1.5 bilhões de prescrições de 600.000 médicos, usadas em 33.000 farmácias. Usa o banco para verificar que médicos mudaram seu padrão de prescrições para informar à companhias farmacêuticas, que podem decidir por campanhas de marketing dirigido aos médicos.

- **Harrah´s Entertainment Inc.** (WASSERMAN, 2000): Um cassino de Las Vegas dobrou lucros usando informações de cartões de "jogadores freqüentes", identificando que um grupo de jogadores que gastavam entre 100 e

[1] *churn* neste caso é uma medida percentual que indica o número de assinantes de um serviço ou consumidores que abandonam ou trocam de provedor.

499 dólares (30% dos jogadores) geravam a maior parte do lucro do cassino. O cassino testou diferentes promoções para este grupo, obtendo melhor fidelidade com menor custo e aumentando a resposta a campanhas de marketing.

Bases de dados deste porte em operação já estão se tornando cada vez mais freqüentes (Winter Corporation, 2006; Information Week, 2002), e como algumas contém informações cujo conhecimento pode representar vantagem competitiva, é de se esperar que técnicas de mineração de dados sejam cada vez mais aplicadas.

Nesta seção serão apresentados alguns algoritmos clássicos, técnicas de mineração de dados e visualização, e serão comentadas suas principais características e aplicabilidade.

3.2.1 Classificação

O processo de classificação dos dados envolve basicamente a criação de uma função que seja capaz de, usando dados de entrada e um algoritmo, prever a classe ou categoria discreta correspondente a estes dados. Como entrada para algoritmos deste tipo, temos uma quantidade razoável de dados para os quais as classes são conhecidas. Com isso cria-se um classificador ou modelo, o que consiste da fase de treinamento.

Em uma segunda fase, utilizam-se dados para os quais as classes não são conhecidas para indicar classes para os mesmos. Assume-se que dados desconhecidos "próximos" de dados conhecidos terão a mesma classe dos dados conhecidos.

O processo pode ser avaliado utilizando-se dados com classes conhecidas, fazendo a sua classificação e comparando os resultados previstos com os obtidos.

Algumas das técnicas de classificação são:

- **Sistemas especialistas**, onde um especialista define *regras* que serão aplicadas aos dados dos *logs* para decidir a que categoria eles devem pertencer. Sistemas especialistas têm a vantagem de contar com o conhecimento real de um especialista, mas podem ser limitados por não poder lidar com inconsistências ou ocorrências não previstas, requerendo assim constante realimentação. Sistemas especialistas podem ser usados como filtros para

pré-processamento, como feito com sistemas de *whitelisting* (ignorância artificial).

- **Árvores de decisão**, onde um algoritmo cria uma estrutura parecida com um sistema especialista mas usando análise dos dados para definir regras que classificam os dados.

  A vantagem de uma árvore de decisão é que a estrutura criada pode ser facilmente interpretada por humanos, e recriada automaticamente no evento de novos dados ou fenômenos.

- **Algoritmos estatísticos**, que podem criar automaticamente (através da análise do espaço de atributos dos dados) funções de discriminação por regiões ou hiperplanos. Diferentes algoritmos podem usar diferentes medidas estatísticas para calcular as funções que definem as regiões de classificação e rejeição.

  Como estas funções são descritas numericamente, não se pode sempre esperar que sejam facilmente interpretáveis por humanos.

- **Algoritmos baseados em redes neurais**, que são usados como algoritmos estatísticos, freqüentemente podendo criar funções de separação mais precisas mas bem mais complexas.

Técnicas de classificação fazem partições do espaço de atributos e, dependendo do algoritmo e seus parâmetros, é possível fazer partições mais imprecisas (em menor tempo ou com algoritmos mais simples) ou precisas (com melhor treinamento e provavelmente maior custo computacional).

É possível também incorrer em erros de *underfitting* com funções de separação inadequadas, que causam erros de classificação; ou *overfitting* com funções de separação extremamente especializadas, incapazes de generalizar classificações.

3.2.2 Regressão

Regressão de dados é uma técnica estatística similar à classificação, exceto que ao invés de obter uma classe discreta para os dados, tem-se um valor numérico real (valor objetivo) e como resultado do treinamento do algoritmo obtém-se uma função numérica para o cálculo deste valor.

Como entrada para algoritmos de regressão deve-se ter uma quantidade razoável de dados para os quais os valores da função são conhecidos, e a função será então estimada usando um de vários modelos existentes. Os valores da função podem então ser calculados para as entradas.

O processo também pode ser avaliado usando-se a função criada com dados para os quais já temos o valor objetivo, e verificando as diferenças entre os valores objetivo dados e calculados.

3.2.3 Agrupamento

Agrupamento ou clusterização (*clustering*) é o nome dado a técnicas que identificam grupos semelhantes entre si no espaço amostral, o que provavelmente corresponde a eventos semelhantes em um *log*. Estas técnicas também assumem que dados em um grupo devem ser diferentes de dados em outro grupo.

Nesta técnica os dados de entrada não precisam ter identificação de classes, já que os grupos serão identificados sem esta informação. Os algoritmos identificam determinado número de grupos de dados e calculam a associação dos dados de entrada aos grupos de saída. Adicionalmente estatísticas e outras informações sobre os grupos podem ser criadas.

A qualidade do agrupamento pode ser medida com algumas métricas, mas comparações entre algoritmos são complicadas. É necessário uma fase de pós-processamento para fazer uma correspondência entre os grupos e classes de eventos.

Existem duas técnicas principais de agrupamento: a *hierárquica* que começa juntando dados e formando grupos pequenos até que todos os dados pertençam a um único grupo, e que permite a exploração de diversas opções de agrupamento; e a *particional*, que agrupa dados iterativamente até que o número de grupos solicitado esteja estável (não se modificando com mais iterações).

Algoritmos de agrupamento são então quase sempre iterativos, necessitando a comparação de dados entre si várias vezes até que seja possível chegar a grupos estáveis, o que pode causar custos de processamento e tornar a aplicação inviável para grandes massas de dados.

3.2.4 Associação

Técnicas de descoberta de associação trabalham de forma diferente das apresentadas até agora. Basicamente estas técnicas tentam descobrir elementos que ocorrem (ou não) em comum em coleções de dados. Os dados de entrada são estruturas com associações (por exemplo, lista de artigos comprados, pequenas séries temporais multivariadas, existência ou não de um determinado fato em uma entrada de *log*, etc.) e o algoritmo identifica a existência de elementos em comum e suporte para esta existência.

O algoritmo tenta descobrir regras do tipo "Se X ocorre na base de dados, então Y também ocorre (com alguma relação à X)" ou de co-ocorrência, como "se X, Y e Z ocorrem na base de dados então A também ocorre (com alguma relação à X, Y e Z)". Consideramos X, Y e Z como os *antecedentes* da associação; A é o *conseqüente*.

Implementações de algoritmos de associação podem usar várias métricas de qualidade que indicam qual é a credibilidade da regra de associação descoberta. Uma destas métricas é *suporte* em uma associação, que indica o quão rara a associação é em bases de dados. Outra métrica é a *confiança* em uma associação: o antecedente pode ocorrer várias vezes na base de dados mas nem sempre com o mesmo conseqüente associado.

3.3 Análise de *Logs* por Mineração de Dados

Uma vez que as técnicas tradicionais (Seção 2.4) para análise de *logs* são computacionalmente custosas, dado que envolvem configurações por vezes complicadas de ferramentas, não reduzem de maneira inteligente o volume de dados, despendem uma quantidade de tempo absurda e são inteiramente dependentes do conhecimento do analista, novas abordagens devem ser estudadas para fornecer soluções melhores para estes problemas.

Tem havido um esforço muito grande voltado para a pesquisa da aplicação de técnicas de mineração de dados em problemas envolvendo segurança da informação (DOKAS *et al.*, 2002; MENA, 2003; LEE; STOLFO, 1998). Alguns trabalhos e técnicas associadas a aplicações de mineração de dados que podem ser empregadas à análise de *logs* são citados a seguir:

a) Modelos estatísticos (VALDES; SKINNER, 2001);

- Vizinhos mais próximos (STEARLEY, 2006);

b) Algoritmos genéticos (MÉ, 1998);

c) Redes neurais artificiais (CANSIAN, 1997);

- Mapas auto-organizáveis de Kohonen (RAMADAS *et al.*, 2003; RHODES *et al.*, 2000).

d) Máquinas de vetores de suporte (MUKKAMALA *et al.*, 2002);

Nas subseções adiante, serão discutidas abordagens e ferramentas em desenvolvimento ou disponíveis publicamente que utilizam técnicas de mineração de dados para analisar *logs* com enfoque em segurança.

3.3.1 Ferramentas para Visualização

A utilização de técnicas de visualização em *logs* pode ser um primeiro passo para o analista interpretar acontecimentos na grande massa de *logs* coletada. Existem diversas ferramentas que implementam formas diferentes de visualização, algumas delas específicas para *logs* ou tráfego de rede. Algumas ferramentas que realizam visualização de *logs*, na tentativa de auxiliar o analista a encontrar padrões em *logs* úteis no aspecto de segurança são descritas a seguir.

3.3.1.1 Sisyphus

O *Sisyphus* (STEARLEY, 2006) é um conjunto de ferramentas de mineração de dados para tratamento dos *logs* de eventos de sistemas *unix-like*. A proposta do *Sisyphus* é retirar o máximo de carga do analista, promovendo uma interação maior com os *logs* através de métodos de *ranking* e colorização de *logs* baseados em teoria da informação, bem como de uma interface *web*.

Trata-se de um *software* em desenvolvimente e de código aberto, que se utiliza de mineração de conjuntos de itens freqüentes sobre *logs* gerados pelo *syslog-ng*[2]. Além disso a ferramenta foi desenvolvida para utilização em *logs* dos eventos do sistema operacional, mais especificamente de supercomputadores.

[2]Mais informações sobre o syslog-ng podem ser obtidas em <http://freshmeat.net/projects/syslog-ng/>.

3.3.1.2 TNV

The Network Visualizer (TNV) é uma ferramenta de visualização de tráfego de redes de computadores[3], que serve para observar o tráfego de rede pela visualização de pacotes e *links* entre máquinas locais e remotas.

Sua função é facilitar o aprendizado do que vem a ser atividades normais em uma rede, investigar eventos de segurança ou auxiliar na administração de redes através da análise de tráfego. Pode-se ver detalhes dos pacotes, atividades de portas relacionadas a um determinado *host* e ter uma visão geral do conjunto de dados.

TNV pode ler pacotes no formato `pcap`, ou capturar pacotes diretamente da interface de rede, possibilitando, com isso, que um analista a utilize para estudar o comportamento de tráfego normal e anômalo.

3.3.1.3 NVisionIP

NVisionIP é uma ferramenta para visualização de fluxos de rede (LAKKARAJU *et al.*, 2004). Trata-se de uma ferramenta interativa que permite a manipulação das informações geradas por sensores de rede, deixando a tomada de decisões de segurança para o analista. Os fluxos de rede utilizados como entrada para NVisionIP são provenientes do `NetFlow`.

A ferramenta provê uma visão geral da rede, tem capacidade de filtragem por máquinas e portas, auxiliando na detecção de máquinas comprometidas, varreduras, infecções por *worms* e outros tipos de incidentes de segurança, por meio de vários níveis de visualização e baseada no conhecimento do analista.

3.3.2 Detecção de Intrusão por Mineração de Dados

Devido às limitações dos sistemas de detecção de intrusão tradicionais, representados pelas técnicas baseadas em abuso ou anomalia, diversos trabalhos vêm tentando superar os problemas conhecidos e desenvolver sistemas de detecção de intrusão com técnicas de mineração de dados. Espera-se com isso aumentar a taxa de identificação de comportamento intrusivo e reduzir a taxa de falsos positivos.

Como as técnicas de mineração de dados são muitas e variadas, aspectos como a

[3]Mais informações sobre o TNV podem ser obtidas em <`http://tnv.sourceforge.net`>, acessado em janeiro de 2007.

capacidade explicativa, complexidade, custo computacional e capacidade de aprendizagem incremental dos modelos são determinantes na hora de selecionar qual o melhor a ser utilizado em cada situação (ZURUTUZA; URIBEETXEBERRIA, 2005). A seguir serão apresentados algumas das abordagens existentes no campo de IDS utilizando mineração de dados.

3.3.2.1 MINDS

O *Minnesota Intrusion Detection System* (MINDS) é um sistema de detecção de intrusão que usa um conjunto de técnicas de mineração de dados para detectar ataques automaticamente contra redes de computadores e sistemas (ERTOZ *et al.*, 2004). Seu funcionamento é baseado na técnica de detecção de intrusão por anomalia feita de maneira não-supervisionada, na qual é dada uma pontuação para cada conexão de rede que irá refletir no nível de anomalia da conexão. As conexões consideradas anômalas são sumarizadas por um módulo baseado em análise de padrões de associação. O objetivo do MINDS é o de detectar tipos inéditos de tentativas de invasão que não possuem assinaturas conhecidas e, portanto, não podem ser identificadas por sistemas como o Snort. O procedimento realizado pelo MINDS consiste das seguintes etapas:

a) Dados do tráfego de rede são coletados com o conjunto `flow-tools`, provenientes de roteadores;

b) Estes dados são filtrados de modo a remover as conexões de rede que não venham a ser interessantes para análise;

c) É realizado o pré-processamento para extrair características que serão utilizadas na análise através de mineração de dados;

d) A informação resultante serve de entrada para o módulo de detecção de ataques conhecidos, o qual os remove do conjunto;

e) Os dados alimentam então o módulo de detecção por anomalia que se utiliza de um algoritmo de detecção de *outliers* para pontuar as conexões;

f) Finalmente, o módulo de análise de padrões de associação resume as informações, fornecendo para o analista um relatório contendo as conexões que obtiveram as pontuações mais altas.

O módulo de detecção por anomalia executa um esquema de detecção de *outliers* baseado na computação da densidade da vizinhança dos pontos, o que resulta na atribuição de um "fator de *outlier* local", utilizado em conjunto com o algoritmo dos vizinhos mais próximos (NN). O módulo de análise de padrões de associação se utiliza da técnica das regras de associação para executar diversas tarefas no sistema como um todo, tais quais sumarizar as conexões anômalas em relação a sua pontuação, auxiliar na criação de novas assinaturas ou modelos de ataques e ajudar na definição do perfil normal de tráfego.

3.3.2.2 GASSATA

A ferramenta *Genetic Algorithm for Simplified Security Audit Trails Analysis* (GASSATA) propõe aumentar a eficiência da análise de *logs* de segurança de maneira automatizada através da detecção por abuso (MÉ, 1998). O GASSATA utiliza-se de algoritmos genéticos para concluir sua tarefa e, como estes são algoritmos de buscas otimizadas a função de *fitness* deve representar o maior risco ao sistema dentre todos os subconjuntos possíveis de ataques.

São criadas matrizes de ataques com as assinaturas dos ataques sendo representadas por cadeias de comandos em ambientes *Unix*, os quais servem de base para a comparação feita com os comandos dados neste ambiente por usuários com diversos níveis de habilidade.

O objetivo da implementação do algoritmo genético no GASSATA é o de criar correspondências entre um determinado *log* de sistema e um subconjunto das combinações de ataques conhecidos disponíveis. Em um determinado conjunto de *logs* é feita a geração de estatísticas de se há ou não a probabilidade de haver um ataque. Se a probabilidade for alta, o administrador deve vasculhar o *log* para verificar se há ou não atividades maliciosas e onde elas se encontram.

3.3.2.3 ADAM

O sistema *Audit Data Analysis and Mining* (ADAM) funciona como um ambiente de testes para o uso de técnicas de mineração de dados na detecção de intrusão. (BARBARÁ *et al.*, 2001). O projeto ADAM utiliza regras de associação na detecção de anomalias, pois permite a derivação de correlações de características diversas retiradas do tráfego de rede, se estas existirem. O objetivo é encontrar eventos anômalos que permitam a classificação do tráfego em normal ou anômalo.

Primeiramente, é construído e armazenado um perfil de tráfego livre de ataques, representado por um conjunto de itens freqüentes denominados "normais" – Os itens, ou episódios freqüentes são empregados na descoberta de eventos que ocorrem em uma determinada janela de tempo. É então aplicada uma combinação de classificação e mineração de regras de associação sobre *dumps* do `tcpdump`, nos quais são realizadas comparações para determinar e descartar os dados que são claramente não maliciosos. O restante dos dados é comparado aos conjuntos de itens armazenados no perfil normal, onde, se não houver correspondência, as conexões suspeitas são classificadas como **tipo de ataque conhecido**, **tipo desconhecido** ou **alarme falso**, baseadas em um limiar.

Após este período de treinamento, o ADAM está pronto para ser colocado para operar na rede. O algoritmo de mineração de regras de associação atua sobre uma janela de conexões atuais e determina se estas são suspeitas ou não. Sendo consideradas suspeitas, as conexões são marcadas e enviadas juntamente com seus respectivos vetores de características para o classificador já treinado, de maneira a serem etiquetadas apropriadamente.

Para tentar resolver o problema do treinamento quanto às conexões suspeitas, o ADAM se utiliza de um método baseado em estimadores pseudo-Bayesianos, que são usados na detecção de ataques inéditos para os quais não existem dados de treinamento. Este método de estimativa é aplicado através da construção de classificadores conhecidos como "*Naïve Bayes*" (RISH, 2001), os quais são baseados nas probabilidades condicionais.

3.3.2.4 MADAM ID

O *framework "Mining Audit Data for Automated Models for Intrusion Detection"* (MADAM ID) é composto por programas de classificação, de episódios freqüentes e regras de associação, um sistema de construção de características e um sistema de conversão que traduz regras aprendidas *off-line* para módulos de tempo real (LEE; STOLFO, 1999).

Trata-se de um sistema que busca automatizar o processo de aprendizado para prover maior generalização nas regras de classficação, superando o problema de codificação destas regras inerente ao elemento humano. Desta forma, os resultados não ficam limitados à detecção de intrusões conhecidas, mas de novas intrusões que sejam

variações destas instâncias ou que fujam de quaisquer padrões.

Há um pré-processamento inicial que retira características básicas dos *logs*, como duração da sessão, endereço IP e porta de origem e destino, número de *bytes* transmitidos, etc. Os algoritmos de mineração de dados são então aplicados para descrever correlações entre características do sistema e gerar sumários estatísticos das atividades do sistema.

Posteriormente, padrões de *logs* correspondendo a intrusões são utilizados para gerar características adicionais a aumentar a acurácia na detecção de intrusão. Uma outra característica agregada ao MADAM ID é a medida de custo, que considera, por exemplo, vazão e recursos de memória (STOLFO *et al.*, 2000).

3.3.2.5 PHAD

O *Packet Header Anomaly Detector* (PHAD) é um projeto experimental para realizar detecção de anomalias, utilizando dados dos campos do cabeçalho de pacotes de rede relativos aos protocolos TCP, IP, UDP, ICMP e Ethernet (MAHONEY; CHAN, 2001). O treinamento consiste em um algoritmo de detecção de anomalias, o próprio PHAD, aprender as faixas de valores normais para cada campo do cabeçalho dos pacotes nas camadas de enlace, rede e transporte. A taxa de anomalias obtida durante o treinamento é utilizada pelo PHAD para estimar a probabilidade de uma anomalia ocorrer enquanto a ferramenta está executando em modo de detecção.

3.3.2.6 Abordagens por Redes Neurais Artificiais

Muitos trabalhos são realizados utilizando redes neurais artificiais, devido a vasta gama de tipos e arquiteturas possíveis e das suas características de flexibilidade e adaptatividade. As abordagens variam dos perceptrons de múltiplas camadas (HAYKIN, 1998) até os mapas auto-organizáveis de Kohonen (KOHONEN, 1995).

No SADI (Sistema Adaptativo de Detecção de Intrusão) (CANSIAN *et al.*, 1997), as redes neurais são utilizadas para realizar detecção de intrusão por abuso, com a criação de assinaturas em um formato que possa servir de entrada para ser processada pelos neurônios. A proposta de adaptatividade visa facilitar a inserção de novos perfis à base de dados de perfis de intrusões bem conhecidas e realizar o retreinamento da rede neural para levá-los em conta.

Uma outra abordagem proposta em (CHAVES; MONTES, 2005) utiliza mapas de Kohonen para detecção de *backdoors* e canais dissimulados. O processo de detecção envolve a reconstrução das sessões TCP/IP, a análise de comportamento do protocolo utilizado na sessão e classificação como *backdoor* ou canal dissimulado, a geração de relatório contendo o resultado da análise, a classificação feita e informações das sessões TCP/IP.

Também é possível se utilizar múltiplos mapas auto-organizáveis para detecção de intrusão (RHODES *et al.*, 2000). Nesta abordagem, uma pilha de monitoração é utilizada para a reconstrução das atividades de rede, com analisadores de protocolo que reduzem e segregam o tráfego antes deste ser submetido ao mapa. Cada rede neural pode ser vista como um especialista treinado para reconhecer atividades normais de um determinado protocolo e emitir alertas quando um desvio significante é detectado.

3.3.2.7 Abordagens por Sistemas Imunológicos Artificiais

Os sistemas imunológicos artificiais baseiam-se no comportamento do sistema imunológico natural para defesa do organismo, controlando o que pode entrar e identificando o que é próprio ou não.

Um trabalho nesta linha é um *framework* geral para um sistema adaptativo distribuído chamado *"Artificial Immune System"* (ARTIS), o qual foi aplicado na área de segurança de computadores sob a forma de um IDS denominado *"Lightweight Intrusion Detection System"* (LISYS) (HOFMEYR; FORREST, 2000). O sistema realiza detecção de intrusão por anomalia e por abuso, fazendo inclusive a extração das assinaturas de dados impróprios, e é considerado eficiente na detecção de intrusão enquanto mantém um baixo nível de falsos positivos.

Outro trabalho envolvendo sistemas imunológicos artificiais é o ADENOIDS, baseado em um *framework* cujos objetivos são o da detecção e resposta precisas a ataques conhecidos e ataques que o sistema aprendeu a reconhecer, e detecção de ataques desconhecidos através da análise de evidências de ataques bem sucedidos contra o sistema (PAULA *et al.*, 2004). O ADENOIDS foi projetado para detectar ataques na camada de aplicação e automatizar a extração de assinaturas para ataques remotos de *buffer overflow*.

3.4 Conclusão

Como visto neste capítulo, existem várias tentativas de se utilizar mineração de dados para auxiliar a análise de *logs*, que não é uma tarefa possível de ser completada plenamente por um analista humano sem lançar mão de diversas ferramentas e recursos.

Qualquer uma destas abordagens não resolve sozinha o problema de classificação de *logs*, embora seu uso combinado entre si ou com outras técnicas de análise e filtragem podem ter resultados bastantes satisfatórios em determinados ambientes.

Muitas das ferramentas citadas tratam-se de protótipos, servem de apoio para sistemas de detecção de intrusão por abuso, tal qual o Snort, e não estão publicamente disponíveis para testes. Baseados nos artigos que as descrevem os resultados obtidos são promissores.

Um outro ponto a ser notado é que existem muitas abordagens de mineração de dados aplicadas à detecção de intrusão, impossibilitando a análise e descrição de todas elas. Além disso, a grande maioria das abordagens é experimental, não se encontrando ainda em condições de disponibilização ao público. Por este fato, só os sistemas julgados mais relevantes foram comentados no presente capítulo.

Dentre as ferramentas citadas, pode-se perceber que uma boa parte é aplicada a um tipo específico de ataque, não abordando os problemas de segurança relacionados ao tráfego de rede de modo genérico. Uma análise superficial indica que muitas delas não podem ser adotadas em um ambiente de produção, devido principalmente à falta de testes mais extensos ou à complexidade da implementação.

Além disso, estas ferramentas estão sujeitas aos mesmos problemas dos sistemas de detecção de intrusão por anomalia ou assinatura, seja na dificuldade de se gerar perfis de tráfego normal sem contaminação, seja na incapacidade de detectar ataques que fogem de determinados padrões estabelecidos.

No próximo capítulo será feita uma avaliação de alguns algoritmos de mineração de dados aplicados a análise de *logs*, visando verificar a possibilidade de uso com o tráfego de rede do INPE. Será apresentada também uma proposta de um protótipo para análise de *logs* cujo objetivo é o de automatizar a tarefa de filtragem, reduzindo o volume de dados para análise humana e a quantidade de falsos positivos, enquanto

identifica tendências de comportamento intrusivo para a rede em questão.

CAPÍTULO 4

CONSTRUÇÃO DO PROTÓTIPO

4.1 Introdução

Como visto, há inúmeras possibilidades para aplicação de mineração de dados em segurança de computadores. Algumas se propõem a detectar comportamentos intrusivos específicos, outras a trabalhar em conjunto com sistemas de detecção de intrusão, sendo que os autores de todos os trabalhos referenciados reportaram bons resultados em diferentes tipos de detecção de intrusão. Como o passo inicial da detecção de intrusão engloba a análise de dados, sendo que na maior parte dos casos estes dados são *logs* de tráfego de rede, tais trabalhos mostraram que é viável o emprego de mineração de dados na análise de *logs* de tráfego de rede.

Se uma técnica de mineração de dados aplicada ao problema da análise de *logs* obtiver um bom índice de detecção de eventos suspeitos, ao mesmo tempo em que conseguir manter baixas taxas de falsos positivos, então é possível utilizar tal técnica para classficar *logs* e prover um conjunto reduzido para o analista.

Neste trabalho, pretende-se utilizar mineração de dados não para substituir as abordagens tradicionais, mas para complementá-las, procurando encontrar informação útil na enorme massa de *logs* que não geram alertas em IDSs. Esta informação pode ajudar na identificação de incidentes de segurança que não possuem assinaturas de ataque codificadas ou não causam desvios significativos em relação aos perfis normais de tráfego de uma rede.

Este capítulo tem por objetivo avaliar alguns algoritmos de mineração de dados aplicados à análise de *logs* de tráfego de rede do INPE. Esta avaliação visa verificar qual dos algoritmos avaliados é o mais adequado nesta situação específica, levando em conta algumas características desejadas descritas adiante, tais como tempo de processamento, compreensão dos resultados e taxa de redução de *logs*. Serão comentados os resultados preliminares e, após isto, descreve-se a metodologia que será utilizada e o modelo do protótipo que será implementado para automatizar a classificação.

4.2 Avaliação de Algoritmos

O processo de testes de algoritmos de mineração de dados com os dados disponíveis consiste de vários passos. Estes passos envolvem a escolha dos algoritmos, os atributos que irão representar os dados, a fase de pré-processamento para criar um conjunto de dados de entrada, a aplicação do algoritmo propriamente dito e, por fim, a avaliação dos resultados.

4.2.1 Requisitos Desejáveis

Uma abordagem de mineração de dados será considerada adequada para o prosseguimento do trabalho se obedecer de maneira equilibrada aos seguintes requisitos:

- Conseguir tratar uma grande quantidade de dados de forma a observar o *timing*, ou seja, não provocar atrasos que inviabilizem a análise de *logs* e possível tomada de contra-medidas em tempo hábil;
- Não possuir tempo de treinamento extenso para garantir a flexibilidade do modelo, ou seja, possibilitar que novos dados sejam inseridos e reconhecidos sem um custo de treinamento elevado;
- Permitir a compreensão dos resultados de modo que seja-se capaz de entender quais são os atributos que fazem a diferença, ou seja, aqueles que realmente propiciam a separação dos dados em classes distintas;
- Obter resultados com taxas aceitáveis de falsos positivos e falsos negativos, de forma a reduzir drasticamente a quantidade de *logs* e eventualmente detectar comportamento suspeito no tráfego de rede.

Após a validação dos requisitos desejáveis para alguma das técnicas de mineração de dados avaliadas, será feita a implementação de um protótipo funcional utilizando esta técnica, o qual possa ser melhor testado e avaliado, com refinamentos no método.

4.2.2 Escolha do Algoritmo

O escopo deste trabalho consiste basicamente na classificação de *logs* de tráfego de rede em normais ou suspeitos. Com isto em mente, foram escolhidas três técnicas utilizadas em tarefas de classificação supervisionada para serem aplicadas no problema: *k-vizinhos mais próximos* (KNN), redes neurais artificiais do tipo *perceptron* de múltiplas camadas (MLP) e árvores de decisão.

O KNN trabalha com a hipótese de que as instâncias de uma mesma classe estão próximas entre si no espaço de atributos (ADRIAANS; ZANTINGE, 1996). O algoritmo de classificação KNN funciona da seguinte forma: para cada amostra com classe desconhecida, compara-se a distância dela para cada amostra com classe conhecida. Então, é atribuída a classe das "K" amostras mais próximas da instância desconhecida (SANTOS, 2006).

Redes neurais artificiais são estruturas de computação paralela ou distribuída compostas por unidades simples de processamento denominadas neurônios (HAYKIN, 1998). As redes neurais artificiais são geralmente usadas para solucionar tarefas de aprendizagem complexas, sendo capazes de lidar com problemas generalizados ou especializados. Há muitos algoritmos diferentes de redes neurais, mas neste trabalho será utilizado o modelo do perceptron de múltiplas camadas (MLP).

Uma árvore de decisão (QUINLAN, 1993) é um tipo de algoritmo de aprendizado de máquina conhecido também como algoritmo de particionamento sucessivo. Este tipo de algoritmo opera através do particionamento do conjunto de dados original em mais subgrupos homogêneos sucessivamente, os quais são particionados em nós até que o nível de detalhamento desejado seja alcançado. Uma decisão de partição deve ser tomada em cada nó, de modo a classificar os dados de acordo com a classe de um nó-folha em um subgrupo.

Estas técnicas são implementadas no pacote de *software* WEKA (WITTEN; FRANK, 2000), que é o acrônimo para "Waikato Environment for Knowledge Analysis", um *software* de código aberto sob licença GNU. WEKA é uma coleção de algoritmos para aprendizado de máquina para tarefas de mineração de dados, que contém ferramentas para pré-processamento de dados, classificação, regressão, agrupamento, descoberta de regras de associação e visualização.

4.2.2.1 Escolha de Atributos

Foi feita uma análise preliminar para determinar quais atributos são importantes para classificar o tráfego como normal ou suspeito. Dado que o trabalho baseia-se em traços de rede e o objetivo não é buscar por assinaturas no conteúdo dos pacotes, os atributos devem ser derivados dos cabeçalhos TCP/IP destes pacotes. Os cabeçalhos TCP e IP (STEVENS, 1994) contêm informações de tráfego importantes, tais como endereço IP de origem e destino, porta de origem e destino, protocolo da camada de

transporte e de rede, quantidade de dados enviados no pacote, número de seqüência e ACK, *flags* de informação e o conteúdo do pacote. A ilustração dos campos destes cabeçalhos pode ser vista nas Figura 4.1 e Figura 4.2, adiante neste capítulo.

A proposta é a de separar os *logs* de acordo com a classe atribuída às sessões. Uma sessão (TANENBAUM, 2003) trata-se de uma comunicação envolvendo dois pares únicos – endereço IP e porta de origem, endereço IP e porta de destino – os quais possuem marcações de tempo representando o seu começo e final, podendo ser consideradas completas.

A análise das sessões, em detrimento da análise de pacotes individuais, provê outras informações sobre uma comunicação específica, como o número de pacotes e o número de *bytes* trafegados. No caso do número de atributos, não há um padrão fixo e sim escolhas empíricas. O MINDS utiliza 16 atributos para detecção de intrusão, o INBOUNDS com o módulo de mapas auto-organizáveis (ERTÖZ *et al.*, 2003) utiliza 6, a abordagem de (CHAVES; MONTES, 2005) utiliza 9 atributos, a KDD Cup de 1999 (Association for Computing Machinery (ACM), 1999) propõe mais de 30 atributos que podem ser utilizados na detecção de intrusão. Nesta dissertação trabalha-se com a hipótese de que há um conjunto limitado de atributos associados com sessões TCP/IP que podem ser usados na diferenciação entre uma sessão normal e uma sessão com suspeita de ataque.

Os atributos foram escolhidos de forma que não haja combinação linear entre eles, isto é, um atributo hipotético "média de *bytes* da sessão" é a combinação linear de dois outros atributos possíveis – "número de pacotes da sessão" e "quantidade de *bytes* da sessão" – e por isso não será utilizado inicialmente. Assim, escolheu-se ter um número mínimo de atributos que podem ser diretamente retirados do cabeçalho TCP/IP de uma sessão de rede. O cabeçalho TCP (POSTEL, 1981b) pode ser visto na Figura 4.1 e o cabeçalho IP (POSTEL, 1981a) pode ser visto na Figura 4.2. Os atributos que serão utilizados são:

- **Duração da sessão**: corresponde ao intervalo de tempo compreendido entre o primeiro e último pacote de uma sessão;
- **Porta do servidor**: é o serviço oferecido por um servidor da rede onde o sensor de análise foi colocado;
- **Pacotes do servidor**: o número de pacotes enviados pela máquina servi-

dora durante a sessão;

- ***Bytes* do servidor**: quantidade de *bytes* enviados pelo servidor durante a sessão;

- **Pacotes do cliente**: o número de pacotes enviados pelo cliente durante a sessão;

- ***Bytes* do cliente**: quantidade de *bytes* enviados pelo cliente durante a sessão;

- **Classe**: valor nominal que pode ser normal ou suspeito, e é relacionado à fonte dos dados (DMZ ou *honeypot*).

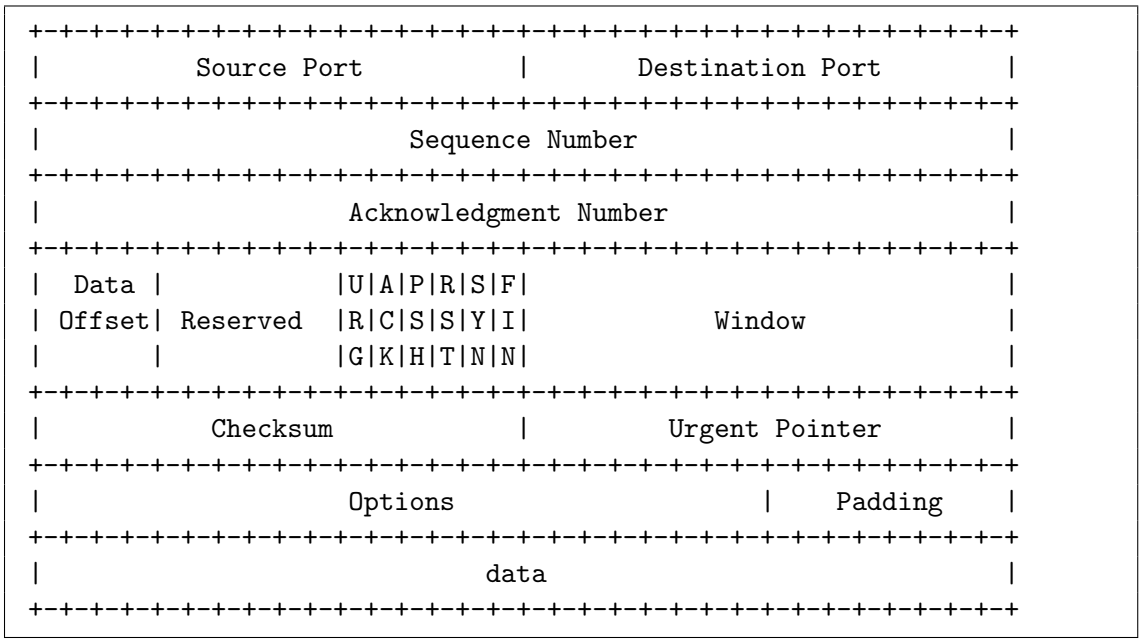

FIGURA 4.1 - Cabeçalho TCP.

Ressalta-se que o atributo **Classe** não influencia a análise dos dados, mas serve para que o modelo seja treinado e que estatísticas dos resultados sejam calculadas. O valor atribuído à classe pode ser normal, se os dados forem advindos da DMZ, ou suspeito, se provenientes de um *honeypot*. Um exemplo dos vetores de atributos utilizados neste trabalho pode ser observado na Figura 4.3, na qual os valores entre vírgulas representam os atributos escolhidos na ordem em que foram descritos anteriormente.

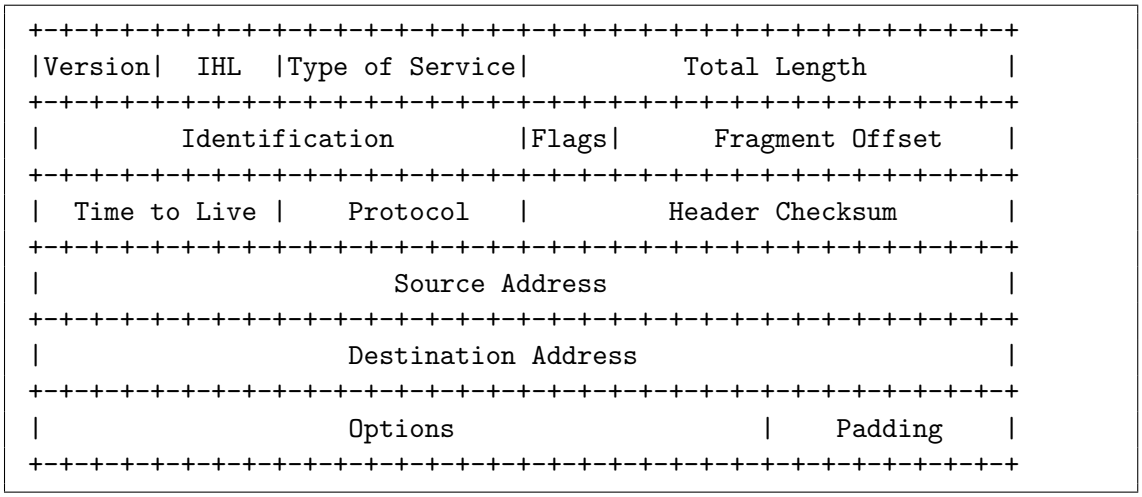

FIGURA 4.2 - Cabeçalho IP.

```
99,22,14,144,32,0,normal
99,22,9,176,8,36,normal
99,80,2,0,1034,0,normal
99,80,7,6,43,0,normal
3,80,0,0,2,0,suspicious
4,80,0,0,2,0,suspicious
0,80,1,0,1,0,suspicious
0,80,1,0,1,0,suspicious
```

FIGURA 4.3 - Conjunto de vetores de atributos conforme utilizados neste trabalho.

4.2.2.2 Pré-processamento de Dados

De modo a avaliar os algoritmos de mineração de dados citados, dois tipos de conjuntos de dados foram separados: um conjunto contendo tráfego proveniente da DMZ do INPE que foi classificado como normal e um conjunto com tráfego direcionado ao *honeypot* de baixa interatividade no bloco IP do INPE que foi classificado como suspeito. Dados obtidos de *honeypots* são inerentemente associados a ataques, já que estes não são divulgados nem oferecem serviços reais. Nesta avaliação inicial, é necessário enfatizar que o tráfego classificado como normal não passou por nenhum tipo de filtragem, isto é, há possibilidade de que haja contaminação por eventuais ataques que possam ter ocorrido contra servidores da DMZ.

Para testar os algoritmos, foram construídos diversos conjuntos com dados reais da DMZ e dados do *honeypot* separados por dia. Algumas ferramentas foram desenvolvidas para auxiliar o processo de obter um dado de *log* em formato `pcap` e transformá-lo em vetores de atributos que pudessem ser lidos pelas implementações contidas no WEKA. Estas ferramentas extraem informações básicas do cabeçalho

TCP/IP de cada sessão e as classificam de acordo com sua origem (DMZ ou *honeypot*), convertendo os *dumps* para arquivos em formato ARFF (*Attribute-Relation File Format*) utilizados pelo WEKA.

Com a etapa de pré-processamento concluída, os algoritmos selecionados são testados tendo como entradas os conjuntos de dados gerados. As próximas subseções contêm os procedimentos dos testes realizados para cada algoritmo, assim como são apresentados e comentados os resultados.

4.2.2.3 Dados para Teste

Os dados para teste foram retirados do coletor de *logs* (Figura 2.7) e correspondem a alguns dias dos meses de fevereiro, abril e junho de 2005. Para cada dia, separou-se o *dump* da DMZ e do *honeypot*, e os dados foram processados como descrito anteriormente. Neste ponto do processo deparou-se com a primeira dificuldade de se lidar com *logs*: *backups* corrompidos ou inexistentes. *Logs* corrompidos ou sessões incompletas foram descartados, de modo que apenas dados confiáveis fossem utilizados, e dados confiáveis puderam ser obtidos somente de alguns dias do período referido. Estes dias e a quantidade correspondente de entradas em *logs* normais e suspeitas são listadas na Tabela 4.1.

Um fato importante sobre os dados das sessões suspeitas deve ser mencionado. Pela experiência do autor em administração de redes, o esperado é que, em uma rede de computadores, a razão entre os ataques e o tráfego legítimo seja bem menor do que 1. No caso da comparação entre tráfego recebido por máquinas da DMZ e por um *honeypot*, o número de sessões contendo ataques corresponde a uma fração pequena da atividade normal de rede. Além disso, os serviços abordados no perfil normal têm que ser os mesmos abordados no perfil suspeito, ou seja, se há um servidor Web na DMZ, filtra-se os *logs* do *honeypot* por ataques direcionados à porta 80 (deve-se, de preferência, emular um servidor similar).

Como o *honeypot* acaba por receber tráfego para diversos serviços emulados que não existem na DMZ – ou rede de produção – como por exemplo, um servidor de banco de dados aberto para a Internet operando na porta 1433, o tráfego para estes serviços é retirado dos *logs*. Para haver correspondência entre os *logs* da DMZ e os *logs* normais, foi separado o tráfego direcionado às portas 21, 22 e 80 em ambos os conjuntos de dados. Por motivos de separabilidade das classes e dos atributos escolhidos, tráfego

TABELA 4.1 - Número de sessões normais e suspeitas por dia do mês.

Dia-Mês	Número de Sessões Normais	Número de Sessões Suspeitas
01-fevereiro (01/02)	376136	26026
02-fevereiro (02/02)	293058	24008
03-fevereiro (03/02)	299394	24012
04-fevereiro (04/02)	290343	16016
05-fevereiro (05/02)	8193	264
06-fevereiro (06/02)	113535	12024
07-fevereiro (07/02)	119898	10836
08-fevereiro (08/02)	120366	10010
09-fevereiro (09/02)	52074	2412
28-abril (28/04)	189806	19038
29-abril (29/04)	220838	20020
30-abril (30/04)	119774	12462
01-junho (01/06)	571550	26026
04-junho (04/06)	122064	6513
06-junho (06/06)	204112	13013
07-junho (07/06)	175848	15015
28-junho (28/06)	10914	714

para as portas 25/TCP (SMTP) e 53/TCP (DNS) foram desconsiderados, já que a redução destes *logs* e possível detecção de comportamento suspeito envolve também a análise do conteúdo dos pacotes.

Os algoritmos que estão sendo avaliados tendem a fazer esquemas simples de classificação para os dados, portanto, se os dados do *honeypot* forem utilizados do mesmo jeito que eles são coletados (dezenas de ataques em relação a centenas de milhares de sessões de tráfego legítimo), eles seriam completamente ignorados, isto é, considerados normais, de modo a obter um classificador mais simples e com maior taxa de correção. Isto não é desejado neste caso, pois é preferível utilizar um classificador com uma taxa de correção menor, mas que seja mais preciso na classificação. Para evitar que isto ocorra, os dados de ataque foram replicados, sem adição de dados derivados ou de perturbação aleatória, para corresponder a pelo menos 5% dos dados normais.

Os testes e resultados individuais de cada um dos três algoritmos são expostos a seguir. As tabelas utilizadas para descrever os resultados obtidos consistem dos campos abaixo relacionados:

- **Tempo de construção do modelo em segundos (TCM (s))**, isto é,

o tempo em segundos gasto pelo WEKA para construir um modelo para avaliar os dados de entrada;

- **Falsos positivos (FP)**, ou a porcentagem de dados normais classificados erroneamente como suspeitos;
- **Falsos negativos (FN)**, ou a porcentagem de dados suspeitos classificados erroneamente como normais;
- **Correção**, que corresponde à porcentagem de dados corretamente classificados com o modelo atual, relacionados à todas as instâncias avaliadas;
- **Efetividade**, que mede a usabilidade do modelo para outros conjuntos de dados. Esta medida é similar à "Correção", com o diferencial de que um modelo gerado é aplicado a dados do dia anterior disponível (ou do próximo dia disponível, no caso de um conjunto de dados pertencer a um mês diferente).

Estes testes são feitos para que se possa principalmente estimar a taxa de falsos positivos e negativos de cada algoritmo, bem como o tempo que eles demoram para avaliar os dados. Após analisar os resultados, um dos algoritmos será escolhido para ser implementado no protótipo.

4.2.2.4 Resultados do KNN

O primeiro lote de testes foi feito utilizando o método mais simples possível com o algoritmo KNN: o que compara e classifica de acordo com apenas um vizinho mais próximo. Este classificador trabalha calculando distâncias e comparando-as no espaço de atributos, atribuindo à uma instância a classe da instância cuja distância é a menor. Com os dados de *logs* apresentados, a separação dos dados foi feita através da criação de um conjunto de hiperplanos para dividir as instâncias em duas classes: normal e suspeita. Dependendo da quantidade de dados e do número de vizinhos escolhidos como parâmetro de comparação, o algoritmo dos vizinhos mais próximos pode ter um tempo de processamento muito longo. Muito tempo de processamento não é desejável neste trabalho, independente do nível de classificação que o modelo possa prover.

Na primeira tentativa, muitas horas se passaram sem que o processo tivesse sido completado. Decidiu-se portanto verificar o tempo de classificação do KNN com

um procedimento de incremento passo a passo. Na Tabela 4.2 é possível observar os resultados obtidos em alguns conjuntos utilizando o algoritmo do vizinho mais próximo, onde a coluna "QS" representa a quantidade de sessões por conjunto. Assim, tentou-se classificar o menor conjunto de dados, "(05/02)", com o método do 1-vizinho mais próximo. Este teste levou 70 segundos e resultou em 100% de correção. Então, aumentou-se o K de 1 para 3 e aplicou-se o algoritmo no mesmo conjunto de dados, o que resultou também em 100% de correção, levando 91 segundos para finalizar o processo.

A seguir, passou-se para o segundo menor conjunto de dados (28/06), o qual demorou 197 segundos para terminar, com 97.76% de correção. Nota-se que um aumento de aproximadamente 30% na quantidade de sessões do conjunto de dados quase triplicou o tempo para classificação. O próximo passo foi avaliar um conjunto de dados um pouco maior (09/02), o qual teve uma taxa melhor de correção, correpondendo a 99.25%, mas seu tempo de classificação aumentou consideravelmente, chegando a 5163 segundos. O último teste deste lote foi feito no conjunto "(28/04)", que possui 189806 instâncias consideradas normais e 19038 instâncias suspeitas. Após mais de três horas em execução (exatamente 11676 segundos), o classificador havia avaliado apenas 36900 instâncias, ou 17.67% do total.

TABELA 4.2 - Resultados obtidos utilizando o método do vizinho mais próximo.

Data	QS	TCM	Correção
(05/02)	8457	70s	100%
(28/06)	11628	3m17s	97.76%
(09/02)	54486	1h43m	99.25%
(28/04)	208844	3h24m	apenas 17.67% das sessões foram avaliadas

Com a utilização de mais vizinhos para comparação, o tempo gasto na classificação deve aumentar em uma proporção muito grande, devido a quantidade de cálculos de distâncias requeridos. Os resultados aqui obtidos mostraram a impossibilidade de utilizar o KNN nos conjuntos de dados propostos sem modificações, uma vez que a maior parte destes conjuntos possuem mais de 100000 instâncias e o tempo é um fator chave neste trabalho. Uma variação a se considerar é armazenar menos exemplos classificados para otimizar a comparação, em um esquema de sumarização ou agrupamento de instâncias similares. Deste modo, diminui-se a complexidade de

processamento, conseqüentemente reduzindo o tempo de execução.

4.2.2.5 Resultados das Redes Neurais

A arquitetura da rede neural MLP utilizada foi "6:2:2", isto é, 6 entradas, 2 neurônios na camada escondida e 2 neurônios de saída que indicam as classes. Esta arquitetura pode ser visualizada na Figura 4.4.

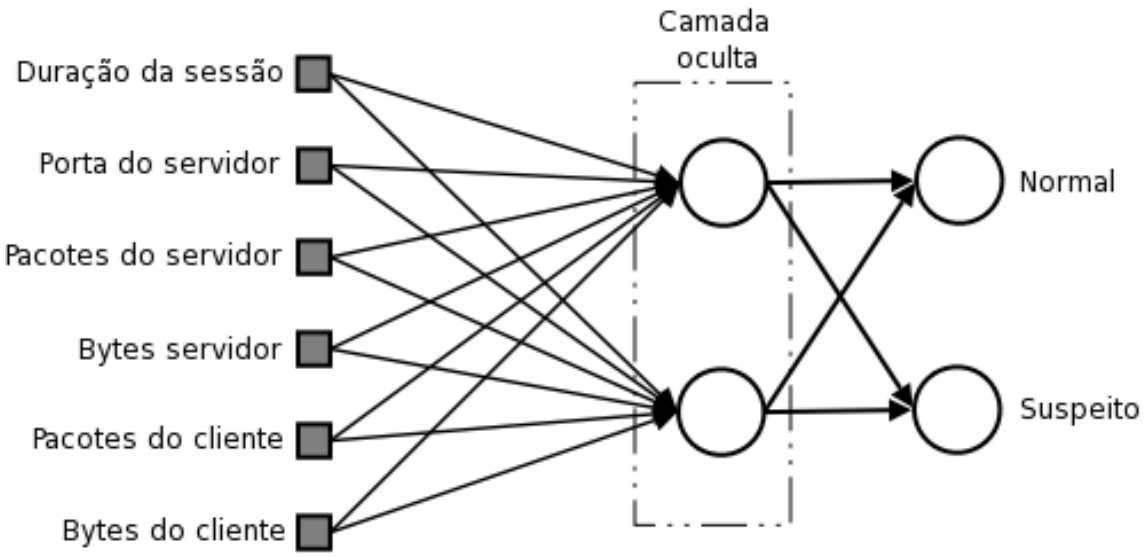

FIGURA 4.4 - Arquiterura de uma MLP "6:2:2"

Como não é possível observar em um gráfico tradicional a separabilidade dos dados, já que o espaço de atributos possui seis dimensões, é preciso inferir o número de neurônios na camada escondida. Não há um padrão quanto ao número de neurônios ou camadas intermediárias que devem ser utilizados, sendo este valor definido empiricamente para cada caso. Os neurônios da camada escondida são utilizados para a criação de hiperplanos e combinações, almejando classificar distribuições não linearmente separáveis. Testes iniciais com os dados mostraram que mais do que uma camada escondida retarda o processo de classificação e não apresenta resultados melhores (GRÉGIO *et al.*, 2006).

A normalização é feita automaticamente para cada conjunto, e o tempo gasto em cada teste variou entre 37 e 2216 segundos, o que foi considerado aceitável. As taxas de falsos positivos foram menores do que 1% em todos os conjuntos de dados

avaliados, mas a taxa de falsos negativos foi demasiado alta na maioria dos casos. Este tipo de resultado indica que os dados estão sobrepostos em algumas partes, não sendo possível em alguns casos identificar o que são sessões normais e o que são sessões de ataque no espaço de atributos e com a quantidade de hiperplanos gerados. No conjunto "(02/02)" não houve nenhuma classificação.

Apesar de este trabalho não ter a pretensão de ser um sistema de detecção de intrusões, e sim ter o objetivo de reduzir *logs* para encaminhá-los ao analista de segurança, casos como os descritos acima não podem ocorrer. Mesmo que a taxa de falsos positivos seja bem baixa, não é desejado que uma quantidade tão grande de sessões suspeitas sejam consideradas normais, pois com estes modelos o analista perderá uma boa parte dos ataques ou comportamentos suspeitos ocorrendo em sua rede. Os resultados para todos os conjuntos de dados disponíveis são apresentados na Tabela 4.3.

TABELA 4.3 - Resultados obtidos utilizando uma rede neural MLP 6:2:2.

Data	TCM (s)	FP	FN	Correção	Efetividade
(01/02)	1281.65	0.11%	30.76%	97.89%	99.76%
(02/02)	1098.86	0	100%	92.42%	96.37%
(03/02)	950.39	0	0.04%	99.99%	91.71%
(04/02)	974.38	0.49%	62.25%	96.27%	96.62%
(05/02)	49.28	0	7.57%	99.76%	99.80%
(06/02)	450.95	0.13%	36%	96.42%	86.20%
(07/02)	467.82	0.02%	27.77%	97.67%	99.99%
(08/02)	481.89	0.004%	80%	93.85%	99.70%
(09/02)	165.68	0.01%	33.33%	98.51%	99.26%
(28/04)	1047.1	0.16%	52.63%	95.05%	99.85%
(29/04)	741.69	0.47%	50%	95.40%	94.71%
(30/04)	467.07	0.35%	22.52%	97.55%	94.73%
(01/06)	2216.69	0.87%	12.28%	98.62%	99.88%
(04/06)	470.2	0.06%	50.97%	95.02%	98.36%
(06/06)	598.81	0.09%	15.38%	98.99%	96.18%
(07/06)	580.87	0.01%	13.33%	98.94%	95.84%
(28/06)	37.91	0.43%	42.85%	96.95%	98.17%

Considerando o objetivo da redução de dados para análise humana, as redes neurais obtiveram bons resultados, embora tenham causado o efeito colateral de deixar passar indetectadas várias sessões suspeitas. Quanto aos modelos criados, há a facilidade

de reutilização da arquitetura e dos pesos, mas a interpretação dos resultados é difícil, tornando-se uma espécie de "caixa-preta" devido a complexidade do mecanismo de classificação.

Testes adicionais utilizando 10 neurônios na camada escondida foram realizados, visando verificar se a separação seria melhor com mais hiperplanos. Estes novos testes apenas aumentaram o tempo de avaliação sem que melhorias fossem observadas nos resultados. Por exemplo, a utilização do conjunto de dados "(06/02)" na nova arquitetura resultou em um tempo de construção do modelo e avaliação das instâncias de 7354 segundos, enquanto que aumentou a taxa de correção em 0.23%. É possível que algumas dezenas ou centenas de neurônios escondidos obtenham resultados melhores, mas esta abordagem foi desconsiderada devido às restrições temporais impostas à análise de *logs*.

4.2.2.6 Resultados das Árvores de Decisão

Por último, foram realizados testes utilizando árvores de decisão para classificação supervisionada. Devido ao importante requisito da compreensão do processo de classificação, não é conveniente gerar uma árvore com nível de profundidade alto. Uma árvore muito profunda pode demorar para ser gerada e se especializar demais, o que não é bom na análise de *logs* de rede por causa da mudança gradual do comportamento das redes ao longo do tempo. Além disso, uma árvore de menor profundidade facilita a interpretação dos resultados, como pode ser visto no exemplo da Figura 4.5. Neste trabalho foram utilizadas árvores com podas e fator de confiança de 0.25.

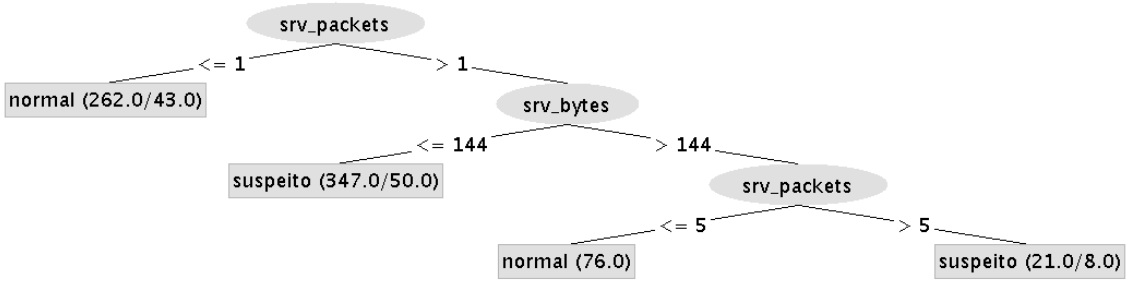

FIGURA 4.5 - Um exemplo simples de árvore de decisão.

Como todos os atributos foram testados de modo a criar nós e esta criação é baseada no ganho de informação provido pelo atributo em relação à classe, as árvores de de-

cisão não possuem a carga de processamento envolvida no treinamento/aprendizado ou no cálculo de distâncias entre milhares de instâncias. Isto faz com que os resultados apareçam rapidamente. A discretização dos atributos numéricos, a qual é feita pelo próprio algoritmo da árvore, contribui para a compreensão dos resultados finais, que foram os melhores obtidos nos testes. As taxas de falsos positivos obtidos para cada conjunto de dados foram bem baixas, enquanto que as taxas de falsos negativos são muito melhores do que as obtidas com as redes neurais utilizadas. Os resultados destes testes são mostrados na Tabela 4.4.

TABELA 4.4 - Resultados obtidos utilizando uma árvore de decisão C4.5.

Data	TCM (s)	FP	FN	Correção	Efetividade
(01/02)	48.05	0.05%	23.07%	98.45%	99.91%
(02/02)	19.38	1.78%	25%	96.45%	97.96%
(03/02)	51.54	1.62%	0	98.49%	98.21%
(04/02)	29.35	1.47%	0	98.59%	98.33%
(05/02)	0.62	0	0	100%	98.85%
(06/02)	11.01	0.97%	0	99.11%	99.54%
(07/02)	11.82	1.08%	0	99%	98.86%
(08/02)	6.06	0.02%	0	99.97%	100%
(09/02)	3.97	0.01%	16.66%	99.24%	99.97%
(28/04)	24.06	2.19%	0	98%	98.21%
(29/04)	41.45	1.37%	10%	97.57%	97.84%
(30/04)	19.98	0.14%	9.67%	98.95%	99.86%
(01/06)	78.11	0.1%	3.84%	99.73%	96.23%
(04/06)	10.92	0.84%	0	99.19%	97.34%
(06/06)	20.71	0.07%	7.69%	99.46%	99.90%
(07/06)	8.27	0.11%	0	99.89%	99.95%
(28/06)	0.98	0.35%	33.19%	97.62%	98.04%

4.2.2.7 Considerações Finais

Os algoritmos foram avaliados com o propósito de verificar se podem ser usados para simplificar uma tarefa que deveria ser feita por um analista humano. Para evitar a análise do conteúdo das conexões de rede, foi enfocado um conjunto relevante de características que podem ser extraídas diretamente do cabeçalho dos pacotes. Espera-se que somente estes atributos sejam suficientes para prover alguma classificação de atividades de rede em normais ou suspeitas.

Dos algoritmos avaliados, concluiu-se que as árvores de decisão conseguiram os me-

lhores resultados, tanto numericamente como considerando sua representação do processo de decisão (GRÉGIO *et al.*, 2007) – árvores de decisão são estruturalmente muito similares a sistemas especialistas e, portanto, podem ser facilmente compreendidas por analistas humanos e modificada para incluir mais conhecimento. Considera-se esta característica como sendo das mais importantes, já que permite ao analista identificar não apenas o que pode ou não ser um ataque, mas o porquê de uma sessão ter sido classificada como suspeita, isto é, quais foram as características e valores usados na classificação.

Dentre os requisitos desejados para uma abordagem ser adequada ao uso neste trabalho, as árvores de decisão cumpriram todos com bastante satisfatoriedade: taxas suficientemente baixas de falsos positivos, com bons níveis de detecção e intervalo de tempo de construção do modelo e avaliação dos dados adequado.

Quanto às redes neurais artificiais, existem vários estudos sobre a capacidade que MLPs tem para reconhecer padrões, sendo consideradas um dos mecanismos mais eficientes de classificação. No entanto, a obscuridade do mecanismo usado para diferenciar classes e a imperfeição na separação dos dados sobressaiu-se, fazendo com que a abordagem fosse descartada para o trabalho.

O algoritmo tradicional de aprendizado de máquina KNN tem o ponto interessante de não necessitar de fase de treinamento e é bastante preciso no caso de as classes serem completamente separáveis. Os resultados obtidos com este algoritmo foram bons, mas a implementação utilizada é computacionalmente muito cara, não sendo aplicável no ambiente de produção do INPE, por não permitir um tempo de resposta aceitável para o analista.

4.3 Metodologia para implementação do protótipo

O cenário a ser trabalhado nesta proposta consiste da DMZ do INPE (Figura 2.7) e de um sensor Snort (BEALE *et al.*, 2004) adicional, como mostrado em Figura 4.6. A máquina coletora de eventos recebe os *dumps* relativos ao tráfego total e os alertas gerados pelo Snort que são exportados. O *honeypot* de baixa interatividade armazena seus próprios *logs* e um coletor situado no CERT.br, em São Paulo, busca estes *logs*, exportando-os por sua vez para um coletor no CenPRA (Centro de Pesquisas Renato Archer), em Campinas.

A construção deste protótipo visa automatizar as tarefas de pré-processamento,

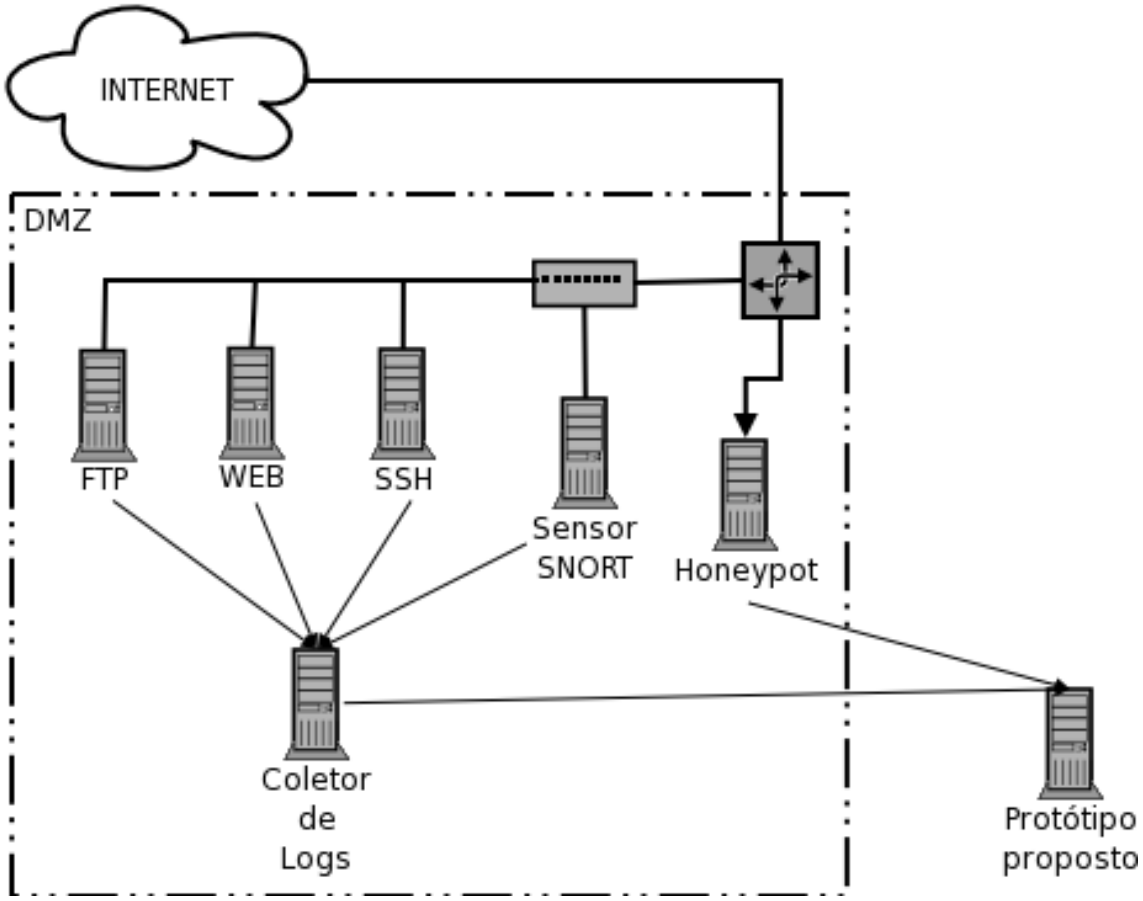

FIGURA 4.6 - Cenário do ambiente com o qual o protótipo irá interagir

classificação e análise, provendo para o analista um conjunto reduzido de *dumps*. A arquitetura do protótipo pode ser observada na Figura 4.7, cujos módulos foram implementados em Java e Perl.

O procedimento a ser executado envolve a criação de um classificador em um determinado dia e sua aplicação nos dados do próximo dia, podendo ser resumido nos seguintes itens:

- Os *logs* da DMZ e do *honeypot* de um certo dia X são coletados pela máquina na qual o protótipo se encontra, sendo tratados pelo módulo de pré-processamento;

- O pré-processamento dos *logs* consiste da extração de sessões contendo tráfego de serviços específicos pré-determinados, e da filtragem do conjunto de dados da DMZ por alertas do Snort;

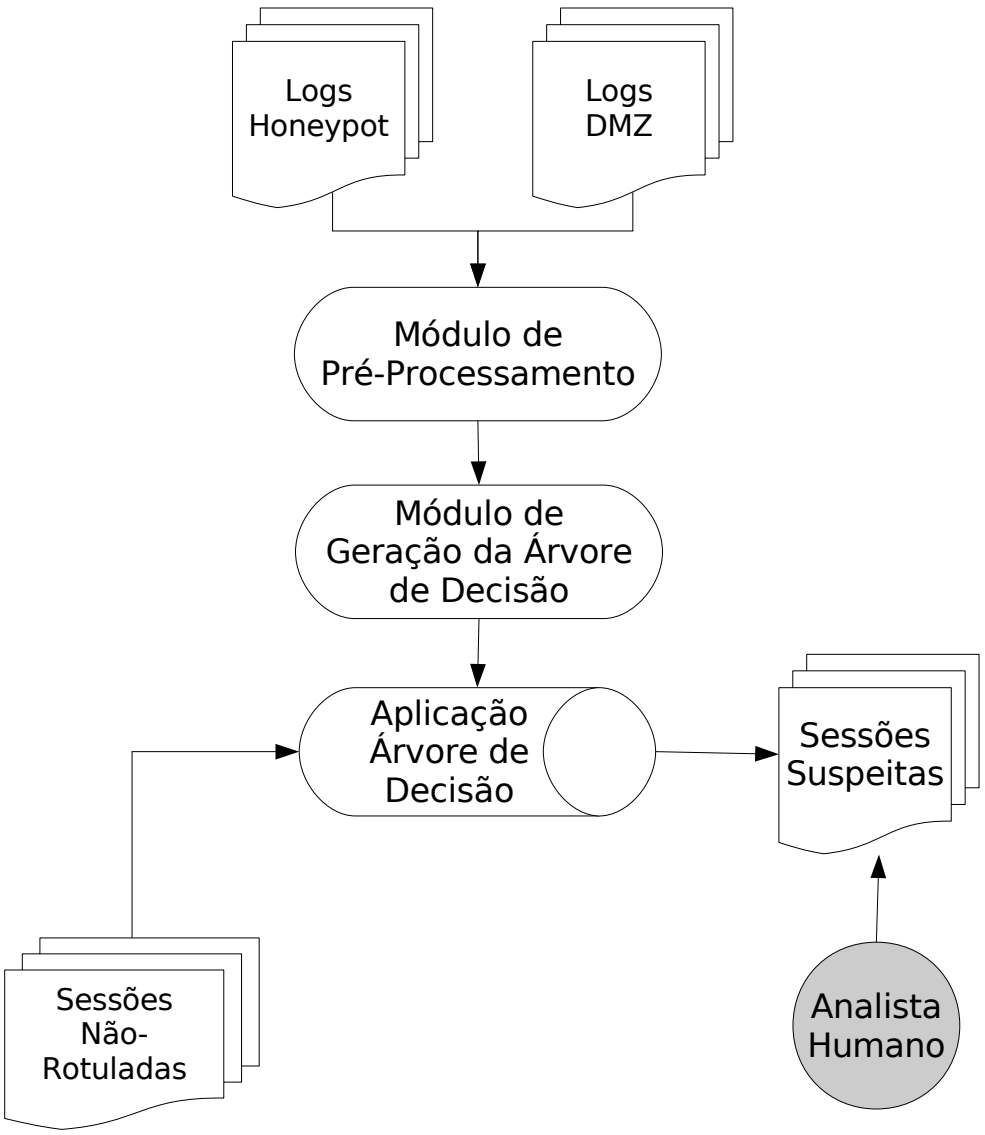

FIGURA 4.7 - Arquitetura do protótipo para classificação e redução de *logs*

- Após transformados em vetores de atributos que constituem os perfis normal e suspeito, estes *logs* seguem para o módulo de geração da árvore de decisão, no qual as regras de classificação são criadas e resultam em uma aplicação Java;

- Conforme forem sendo coletados, os *logs* do dia $X + 1$ são processados e passam pela aplicação da árvore criada, gerando arquivos que apontam as sessões suspeitas;

- Estas sessões suspeitas ficam então disponíveis em formato `pcap`, para que um analista humano as avalie e verifique possíveis atividades maliciosas.

4.3.1 Módulo de Pré-Processamento

O objetivo do módulo de pré-processamento é basicamente o de receber *logs* em formato `pcap` e transformá-los em vetores de atributos que possam ser inseridos no algoritmo da árvore de decisão. Durante este processo, é feita a filtragem dos *logs* da DMZ em busca de ataques e a extração das sessões dos conjuntos de *logs* para a criação dos perfis.

O primeiro passo a ser feito é coletar os *dumps* do *honeypot* e da DMZ, os quais são disponibilizados em arquivos separados. Visando a redução de contaminação do perfil normal, executa-se o Snort sobre os *logs* da DMZ. As regras do Snort têm de ser adaptadas para cada DMZ, visando reduzir o número de falsos positivos.

O Snort irá então gerar os alertas relacionados aos *logs* da DMZ, os quais são extraídos do conjunto e concatenados aos *logs* do *honeypot*. Esta parte do pré-processamento enriquece o perfil suspeito, ao mesmo tempo em que elimina uma parte das impurezas contidas no perfil normal.

O próximo passo consiste na retirada de informações sobre as sessões contidas nos *logs*. Isto é feito utilizando-se a *engine* do Snort, que processa os *dumps* e gera um arquivo de sessões como o da Figura 4.8

```
[*] Session => Start: 01/15/07-13:01:13 End Time: 01/15/07-13:01:28
[Server IP: 192.168.20.50  port: 80  pkts: 6  bytes: 3558] [Client
 IP: 10.0.0.100  port: 2974  pkts: 7  bytes: 316]
```

FIGURA 4.8 - Informações sobre as sessões retiradas de um *logs* de rede.

Deste arquivo de sessões serão retirados os atributos que formarão os vetores de entrada para a árvore de decisão. O valor normal ou suspeito do atributo classe é anexado a cada vetor, de acordo com sua proveniência. O arquivo de vetores é criado em formato ARFF, pois o protótipo realiza uma chamada a uma classe em Java, correspondente à implementação de árvore de decisão do WEKA. O arquivo gerado ao final da etapa de pré-processamento pode ser visto na Figura 4.9.

```
@RELATION log

@ATTRIBUTE session_time REAL
@ATTRIBUTE srv_port REAL
@ATTRIBUTE srv_pkt REAL
@ATTRIBUTE srv_bytes REAL
@ATTRIBUTE cli_pkt REAL
@ATTRIBUTE cli_bytes REAL
@ATTRIBUTE class {normal, suspicious}

@DATA
1,80,6,1044,5,248,normal
0,80,8,4550,6,252,normal
21,21,22,765,33,232,normal
1,22,13,2811,14,387,normal

(...)

15,80,11,11857,10,318,suspicious
25,80,5,279,5,254,suspicious
2,22,59,720,42,580,suspicious
```

FIGURA 4.9 - Arquivo de entrada para a árvore de decisão contendo vetores de atributos.

4.3.2 Módulo de Geração da Árvore de Decisão

O arquivo de entrada é submetido ao módulo de geração da árvore de decisão, que, como o próprio nome indica, cria regras de classificação baseadas nos perfis de tráfego apresentados. Após a classificação ser feita, a árvore resultante é colocada em um arquivo com uma estrutura semelhante a da Figura 4.10.

A partir desta árvore, o módulo faz a análise das regras criando uma aplicação em Java que implemente as estruturas condicionais obtidas. Um fragmento da aplicação contendo as condições para classificação pode ser observado na Figura 4.11.

```
srv_port <= 22
|   srv_pkt <= 1
|   |   srv_bytes <= 18: ataque (8037.0/21.0)
|   |   srv_bytes > 18: normal (162.0)
|   srv_pkt > 1: normal (2664.0)
srv_port > 22
|   cli_pkt <= 1
|   |   srv_pkt <= 0: normal (12837.0)
|   |   srv_pkt > 0
|   |   |   session_time <= 0
|   |   |   |   cli_pkt <= 0: normal (216.0)
|   |   |   |   cli_pkt > 0
|   |   |   |   |   srv_bytes <= 0: ataque (5082.0/1074.0)
|   |   |   |   |   srv_bytes > 0: normal (30.0)
|   |   |   session_time > 0: normal (651.0)
|   cli_pkt > 1: normal (95880.0)
```

FIGURA 4.10 - Árvore de decisão gerada pelo módulo.

4.3.3 Aplicação Árvore de Decisão

Com a aplicação pronta, pode-se determinar um período de tempo (de hora em hora, diariamente, etc.) no qual os *dumps* de tráfego serão avaliados pela árvore gerada. Como a análise não é feita em tempo real, uma opção é utilizar os *logs* assim que eles são rotacionados, pois é necessário extrair as informações sobre as sessões antes de aplicar o algoritmo.

No caso do INPE, os *logs* são separados de hora em hora, tempo em que lhes é feita a compactação em um arquivo nomeado com a data e hora corrente. Para os fins deste trabalho, foram utilizados os *logs* de um dia inteiro. Estes foram coletados e tranformados em sessões, as quais servem de entrada para serem avaliadas pela árvore criada com os perfis do dia anterior. É então gerado um arquivo com as sessões consideradas suspeitas e vários arquivos de *logs* em formato `pcap` para análise posterior, referentes a cada uma destas sessões. O relatório com as sessões suspeitas pode ser visto na Figura 4.12.

4.4 Considerações sobre o Protótipo

Na etapa de pré-processamento, em relação a filtragem do conjunto de dados insuspeitos, espera-se que a intersecção entre as classes seja mínima, embora a separação de classes não dependa só disto. No caso de ser possível dividir os *logs* em classes distintas com os atributos esolhidos, esta filtragem ajuda o processo a ser mais

```
if (srv_port <= 22 ) {
 if (srv_pkt <= 1 ) {
  if (srv_bytes <= 18 ) {
isAttack = true;
  }
  else if (srv_bytes > 18 ) {
isAttack = false;
  }
 }
 else if (srv_pkt > 1 ) {
isAttack = false;
 }
}
else if (srv_port > 22 ) {
 if (cli_pkt <= 1 ) {
  if (srv_pkt <= 0 ) {
isAttack = false;
  }
  else if (srv_pkt > 0 ) {
   if (session_time <= 0 ) {
    if (cli_pkt <= 0 ) {
isAttack = false;
    }
    else if (cli_pkt > 0 ) {
     if (srv_bytes <= 0 ) {
isAttack = true;
     }
     else if (srv_bytes > 0 ) {
isAttack = false;
     }
    }
   }
   else if (session_time > 0 ) {
isAttack = false;
   }
  }
 }
 else if (cli_pkt > 1 ) {
isAttack = false;
 }
}
```

FIGURA 4.11 - Parte do código da Aplicação Árvore de Decisão

efetivo.

A adição dos dados de ataques recebidos pelo *honeypot* contribui para a formação

```
####################
# Date: 2005-Aug04 #
####################

#########################################################

514 unique IP addresses generated 3136 suspicious sessions.

#########################################################

Client IP: AAA.AAA.101.130     Client port: 48811
Server IP: XXX.XXX.XXX.20      Server port: 22

#########################################################

Client IP: BBB.BBB.62.120      Client port: 3045
Server IP: XXX.XXX.XXX.20      Server port: 22

#########################################################

Client IP: CCC.CCC.248.155     Client port: 15132
Server IP: XXX.XXX.XXX.20      Server port: 22

#########################################################
```

FIGURA 4.12 - Relatório contendo as sessões consideradas suspeitas pela árvore.

de um conjunto de treinamento de qualidade e personalizado para as tendências de eventos recebidos pela rede em que o sistema for instalado. O resultado esperado é um treinamento acurado que identifique eventos suspeitos de forma bastante confiável.

Dado que o protótipo é modular, pode-se facilmente modificar a técnica de mineração de dados utilizada para classificar os *logs*, desde que se insira esta nova técnica como uma classe em Java que seja compatível com o formato de entrada de dados disponível.

A aplicação que implementa as regras da árvore de decisão criada é bastante rápida, fornecendo ao analista os *logs* de rede com o conteúdo dos pacotes, para que haja uma avaliação mais detalhada. Este conteúdo não é capturado por completo no INPE, limitando-se a 96 *bytes*. Espera-se com este trabalho que o protótipo possa operar filtrando o tráfego da DMZ, reduzindo seu conteúdo para um conjunto de sessões suspeitas com uma taxa razoável de falsos positivos e consolidando um conjunto que possua tráfego suspeito que não seria alertado pelo Snort.

Cabe ressaltar que a confiabilidade adquirida nos conjuntos de dados é bastante relativa às configurações dos dispositivos componentes do cenário. No caso do Snort, para se obter uma taxa baixa de falsos-positivos são necessários ajustes constantes nas regras de geração de alertas, com o intuito de refletir os serviços realmente instalados na subrede em que o sensor atua. Apesar de, em teoria, todo o tráfego destinado aos *honeypots* ser malicioso, na prática isto não é completamente verdadeiro. Alguns casos podem se tratar de pacotes de rede perdidos ou máquinas mal configuradas, não constituindo tentativas de ataque.

No próximo capítulo, será feito um estudo de caso utilizando o protótipo, de forma a avaliar seus resultados para um ambiente real.

CAPÍTULO 5

RESULTADOS

5.1 Estudo de caso

Após a construção dos módulos do protótipo, foi separado um conjunto de *logs* para a realização do estudo de caso. Recuperou-se os *logs* em formato `pcap` da DMZ relativos ao mês de agosto de 2005. Estes *dumps* correspondem a todo o tráfego com sentido às máquinas da DMZ. Também foram recuperados os *logs* do *honeypot* do INPE deste mesmo período. Neste capítulo serão mostrados os testes realizados, enfatizando os problemas e soluções encontrados para cada caso.

5.1.1 Primeiro Teste

Primeiramente, ambos conjuntos de *logs* foram filtrados para conter somente sessões envolvendo as portas 21, 22 e 80, visando tornar o processo de classificação não tendencioso. Dada a topologia apresentada, o tráfego direcionado à DMZ passa por um sensor Snort, que busca por assinaturas suspeitas. As sessões que causaram a geração de alertas pelo Snort são utilizadas para compor uma parcela do perfil suspeito e retiradas do conjunto de sessões que representa o tráfego normal.

A partir desta filtragem por portas e da separação das sessões indicadas pelo Snort, inicia-se o processo de criação do classificador. Na Tabela 5.1 podem ser vistos números relativos às sessões consideradas legítimas, sessões que causaram a emissão de alertas pelo Snort e sessões extraídas do *honeypot*, separadas por dia.

Supostamente, a utilização de informações sobre sessões que geraram alertas do Snort serviria para o enriquecimento do perfil suspeito. Primeiro por aumentar numericamente o conjunto de dados disponíveis, depois por acrescentar novos tipos de comportamentos suspeitos que podem ser úteis para detecção de atividades intrusivas.

Gerou-se então um perfil normal sem a contaminação de tráfego que o Snort detecta como suspeito, e um perfil suspeito composto pela concatenação das sessões de alerta com as sessões do *honeypot*. A cada sessão dos dois arquivos de perfil foi atribuída uma classe, normal ou suspeita, unindo então os dois perfis com a adição de um cabeçalho para formar um arquivo "ARFF". Este processo foi repetido para cada

TABELA 5.1 - Número de sessões normais e suspeitas por dia do mês de agosto de 2005.

Dia-Mês	Sessões Normais	Sessões do Snort	Sessões do *Honeypot*
01-agosto (01/08)	67473	59	25
02-agosto (02/08)	64209	82	53
03-agosto (03/08)	28252	21	17
04-agosto (04/08)	28271	12	7
05-agosto (05/08)	72811	73	7
06-agosto (06/08)	47605	0	24
07-agosto (07/08)	56555	89	20
08-agosto (08/08)	103321	285	12
09-agosto (09/08)	109852	47	11
10-agosto (10/08)	N/D	N/D	12
11-agosto (11/08)	95124	20	3
12-agosto (12/08)	88052	66	24
13-agosto (13/08)	45039	1	12
14-agosto (14/08)	43171	0	4
15-agosto (15/08)	98399	32	15
16-agosto (16/08)	105387	145	13
17-agosto (17/08)	112731	34	5
18-agosto (18/08)	119608	11	12
19-agosto (19/08)	103401	35	10
20-agosto (20/08)	56130	0	14
21-agosto (21/08)	59117	0	12
22-agosto (22/08)	117173	50	23
23-agosto (23/08)	123396	14	4
24-agosto (24/08)	126925	54	10
25-agosto (25/08)	109135	13	3
26-agosto (26/08)	98838	110	11
27-agosto (27/08)	63190	216	8
28-agosto (28/08)	72127	0	11
29-agosto (29/08)	77339	6	202
30-agosto (30/08)	74071	68	11
31-agosto (31/08)	71559	0	14

dia do conjunto de dados, resultando em 30 arquivos de entrada para a geração de árvores de decisão.

Nota-se que o conjunto "(10/08)" não possui dados da DMZ, devido a algum problema no processo de *logging* daquele dia. Portanto, a partir deste ponto tal conjunto não mais será referenciado.

Para cada um dos arquivos "ARFF" gerados treinou-se uma árvore de decisão, a

qual foi implementada em uma aplicação Java, conforme visto no Capítulo 4, subseção 4.3.2. Como a quantidade de sessões normais é da ordem de dezenas ou centenas de milhares na maioria dos casos as sessões suspeitas não chegam a centenas de instâncias, a classificação falhou miseravelmente. Todas as árvores geradas desta maneira ignoraram as sessões suspeitas, resultando em um classificador que considera qualquer sessão como sendo normal.

Isto ocorre porque, em alguns casos, a acurácia (ou taxa de correção) do modelo criado é maior quando não se prediz uma determinada classe que possui poucas instâncias para treinamento. Este efeito é chamado de "Paradoxo da Acurácia", o qual define que modelos preditivos com um certo nível de acurácia podem ter um poder de predição melhor do que modelos com acurácia maior (BRUCKHAUS, 2006). Uma discussão sobre o uso de dados sintéticos na classificação, isto é, amplificar os dados de treinamento (como foi feito para os testes do Capítulo 4), pode ser vista em (NONNEMAKER, 2006). Na tentativa de tornar os conjuntos de dados um pouco menos desbalanceados, pensou-se em uma proposta contrária à citada, que será discutida a seguir.

5.1.2 Segundo Teste

Dados os fatos subseqüentes, foi necessário lançar mão de uma nova abordagem: uma vez que existem muitas sessões normais que possuem valores idênticos dos atributos escolhidos, é possível reduzir a representatividade deste grupo de sessões considerando-as como uma instância apenas. Pode-se prover uma explicação para este fato, se, simplesmente, várias pessoas têm a página principal do INPE como padrão para seus navegadores e, logo que executam o *browser* acessam outro *site*. Em algumas vezes, além da transação *bytes*/pacotes ser a mesma, a duração das sessões também o é. Outros casos semelhantes envolvendo outros protocolos são passíveis de gerar resultados similares ao exemplo dado.

Assim sendo, havia muitas sessões repetidas atrapalhando o processo de classificação e houve a necessidade de redução. Uma amostra destas instâncias repetidas pode ser observada na Figura 5.1, que mostra os vetores de atributos utilizados como entrada para o módulo de geração de árvores de decisão. Nota-se que as linhas 1, 9, 10, 11, 12 e 13 da figura são vetores idênticos e serão representados como um vetor apenas.

A nova abordagem só foi possível graças a não se ter nenhum atributo muito espe-

```
0,80,3,0,4,0
1,80,2,0,11,0
0,80,3,0,3,0
0,21,8,112,14,70
25,80,3,0,7,0
22,80,3,0,7,0
49,22,1,0,3,0
1,80,3,0,4,0
0,80,3,0,4,0
0,80,3,0,4,0
0,80,3,0,4,0
0,80,3,0,4,0
0,80,3,0,4,0
3,22,2,0,2,0
1,80,2,0,12,0
1,80,2,0,13,0
```

FIGURA 5.1 - Amostra de vetores de atributos repetidos no conjunto das sessões normais.

cífico, tais como aqueles que identificam os pares em uma comunicação. A Tabela 5.2 mostra a quantidade de vetores de atributos únicos restantes após o processo de redução, na qual **VAU** é o número de vetores de atributos únicos e **RzIO** é a porcentagem que estes vetores representam do conjunto de instâncias original.

TABELA 5.2 - Valores representando a quantidade de instâncias nos conjuntos de sessões normais após redução para vetores de atributos únicos (VAU) e razão entre os novos conjuntos e os conjuntos de dados originais (RzIO).

	(01/08)	**(02/08)**	**(03/08)**	**(04/08)**	**(05/08)**	**(06/08)**
VAU	7143	7013	3802	3746	7886	5040
RzIO	10.58%	10.92%	13.45%	13.25%	10.83%	10.58%
	(07/08)	**(08/08)**	**(09/08)**	**(11/08)**	**(12/08)**	**(13/08)**
VAU	5609	10245	10965	9751	9058	4911
RzIO	9.91%	9.91%	9.98%	10.25%	10.28%	10.90%
	(14/08)	**(15/08)**	**(16/08)**	**(17/08)**	**(18/08)**	**(19/08)**
VAU	4504	9622	10313	11076	11770	10419
RzIO	10.43%	9.78%	9.78%	9.82%	9.84%	10.07%
	(20/08)	**(21/08)**	**(22/08)**	**(23/08)**	**(24/08)**	**(25/08)**
VAU	5708	6102	11678	12300	12666	11458
RzIO	10.16%	10.32%	9.96%	9.96%	9.97%	10.49%
	(26/08)	**(27/08)**	**(28/08)**	**(29/08)**	**(30/08)**	**(31/08)**
VAU	9825	6795	6770	8218	8657	8522
RzIO	9.94%	10.75%	9.38%	10.62%	11.68%	11.90%

Feitas as modificações nos conjuntos de dados que representam as sessões normais, o procedimento foi refeito até a geração das novas árvores. Prosseguiu-se então da mesma forma como proposto para um ambiente real: o classificador criado no primeiro dia é utilizado com os dados do segundo dia, e assim sucessivamente. A Tabela 5.3 contém os resultados sumarizados desta abordagem, onde pode-se observar o número de sessões suspeitas encontradas por dia (**SS/D**) e sua razão em relação ao total de sessões que serviram de entrada para o classificador (**RzT**).

TABELA 5.3 - Resultados utilizando Snort para enriquecer o perfil suspeito, onde (SS/D) é o número de sessões que foram consideradas suspeitas no dia e (RzT) é a razão entre tais sessões e todas as sessões que entraram no classificador neste dia.

	(02/08)	**(03/08)**	**(04/08)**	**(05/08)**	**(06/08)**	**(07/08)**
SS/D	22702	12871	3136	8243	14477	0
RzT	35.35%	45.55%	11.09%	11.32%	30.41%	0
	(08/08)	**(09/08)**	**(11/08)**	**(12/08)**	**(13/08)**	**(14/08)**
SS/D	35812	45790	11422	36218	5218	4601
RzT	34.66%	41.68%	12%	41.13%	11.58%	10.65%
	(15/08)	**(16/08)**	**(17/08)**	**(18/08)**	**(19/08)**	**(20/08)**
SS/D	10579	3495	0	2479	10654	7015
RzT	10.75%	3.31%	0	2.07%	10.30%	12.49%
	(21/08)	**(22/08)**	**(23/08)**	**(24/08)**	**(25/08)**	**(26/08)**
SS/D	0	47967	13379	40034	40282	47613
RzT	0	40.93%	10.84%	31.54%	36.91%	48.17%
	(27/08)	**(28/08)**	**(29/08)**	**(30/08)**	**(31/08)**	
SS/D	32937	815	1631	24962	7940	
RzT	52.12%	1.12%	2.10%	33.70%	11.09%	

Analisando os resultados, pode-se verificar que houve uma boa taxa de redução em relação ao conjunto diário de sessões, de mais de 50% na quase totalidade dos casos. Ainda assim, tais resultados não foram satisfatórios, pois um dia com 40 mil sessões suspeitas continua trabalhoso para análise humana. Este mau resultado fez com que os alertas do Snort fossem revistos, para verificar se em vez de enriquecer o conjunto de treinamento para o perfil suspeito, o contrário não estava ocorrendo.

Dentre os alertas gerados, não se observou nenhum referente ao serviço de FTP (porta 21), nem ao serviço de `ssh` (porta 22). Os alertas emitidos para sessões envolvendo o servidor Web (porta 80) eram todos como os da Figura 5.2.

```
1 [**] [116:54:1] (snort_decoder): Tcp Options found with bad lengths [**]
2 08/01-14:29:56.627730 AAA.AAA.AAA.130:58438 -> XXX.XXX.XXX.7:80
3 TCP TTL:114 TOS:0x0 ID:25279 IpLen:20 DgmLen:48 DF
4 ******S* Seq: 0x5C2A63C2  Ack: 0x0  Win: 0x4000  TcpLen: 28
5
6 [**] [116:55:1] (snort_decoder): Truncated Tcp Options [**]
7 08/01-14:29:56.845648 AAA.AAA.AAA.130:58439 -> XXX.XXX.XXX.7:80
8 TCP TTL:114 TOS:0x0 ID:25286 IpLen:20 DgmLen:48 DF
9 ******S* Seq: 0xEEDE976C  Ack: 0x0  Win: 0x4000  TcpLen: 28
```

FIGURA 5.2 - Amostra dos alertas emitidos para o serviço Web.

Segundo Martin Roesch, autor e principal desenvolvedor do Snort, o alerta da linha 1 representa algum tipo de anomalia do protocolo, como um pacote malformado (vide Anexo 1 – Capítulo A). Este alerta pode também aparecer no caso específico de haver uma tentativa de negação de serviço ao Snort, em versões anteriores à 2.4.0, como pode ser visto no Anexo B.

Na linha 6, o alerta partiu do mesmo endereço IP de origem, para o mesmo servidor da DMZ e com o *timestamp* equivalente até o campo dos segundos. No Anexo C há uma explicação para este alerta, que tem a ver com uma opção do cabeçalho TCP cujo comprimento é diferente do configurado.

A análise manual das sessões que geraram estas assinaturas mostrou que as mesmas simplesmente se tratavam de algum erro na comunicação. Quando este tipo de alerta ocorreu, a máquina cliente não conseguiu efetuar o último passo do *three-way handshake*, enviando um `RESET` para o servidor. Poucos minutos mais tarde, a mesma máquina cliente consegue estabelecer a conexão e navega normalmente pela página do INPE.

Uma vez que todos os alertas de porta 80 seguiam o mesmo padrão, decidiu-se retirar estas sessões dos conjuntos suspeitos e realizar novos testes, com o objetivo de verificar o porquê de os resultados não terem sido adequados o suficiente.

5.1.3 Terceiro Teste

Visando obter resultados melhores na redução da quantidade de sessões que é encaminhada ao analista, todos os conjuntos de dados contendo vetores de atributos classificados como suspeitos foram refeitos, excluindo os vetores provenientes de alertas do Snort. Assim, para o treinamento, foi utilizado um conjunto de sessões normais contendo apenas vetores de atributos únicos e um conjunto de sessões suspeitas

somente com vetores de atributos extraídos dos dados do *honeypot*.

Novamente, todas as árvores foram criadas seguindo o procedimento já descrito anteriormente, e os resultados da aplicação destas foram animadores. Na Tabela 5.4 estão os resultados deste último teste, mostrando a quantidade de sessões que foram consideradas suspeitas e a redução alcançada com as novas árvores. Os campos da referida tabela são **Data**, que corresponde ao dia e mês dos dados utilizados; **SDMZ**, que é a quantidade de sessões advindas da DMZ; **SA**, ou a quantidade de sessões consideradas suspeitas para serem verificadas pelo analista e, por fim, **Redução**, que é a porcentagem de sessões restantes em relação às sessões de entrada originais.

Pode-se perceber que os resultados foram bem melhores neste último do que as abordagens tentadas anteriormente. Analisando os resultados da tabela, percebe-se que os dias 5, 6, 11, 12, 15, 17, 18, 24, 26, 28 e 29 não geraram sessão suspeita alguma. Este resultado peculiar diz respeito às árvores classificadoras do dia anterior e será explicado a seguir:

- No caso do dia 11, isto ocorreu devido a não ter sido gerado um classificador para o dia 10, dado que não havia *logs*;
- No caso dos dias 5, 6, 12, 15, 18, 24 e 26, os classificadores gerados no dia anterior atribuíram o valor normal a qualquer sessão, por causa do Paradoxo da Acurácia;
- No caso dos dias 17, 28 e 29, as árvores foram criadas e simplesmente não classificaram nenhuma sessão como suspeita.

Já que o dia 11 foi um caso isolado de falta dos *logs* do dia 10, o processo foi refeito para este dia, aplicando-se a árvore criada no dia 9. O resultado foram 3095 sessões suspeitas, com uma taxa de redução para 3.25% do total.

5.2 Resultados alcançados

Levando-se em conta as animadoras taxas de redução obtidas, que na imensa maioria dos casos foram de bem mais de 90%, será feita uma análise de alguns dos relatórios gerados.

TABELA 5.4 - Resultados alcançados com árvores treinadas com sessões da DMZ sem tráfego de alerta e sessões suspeitas do *honeypot*.

Data	SDMZ	SA	Redução
(02/08)	64209	1786	2.78%
(03/08)	28252	2772	9.81%
(04/08)	28271	3133	11.08%
(05/08)	72811	0	0
(06/08)	47605	0	0
(07/08)	56555	610	1.07%
(08/08)	103321	2411	2.33%
(09/08)	109852	3209	2.92%
(11/08)	95124	0	0
(12/08)	88052	0	0
(13/08)	45039	3	0.006%
(14/08)	43171	4674	10.82%
(15/08)	98399	0	0
(16/08)	105387	2222	2.10%
(17/08)	112731	0	0
(18/08)	119608	0	0
(19/08)	103401	1832	1.77%
(20/08)	56130	473	0.84%
(21/08)	59117	661	1.11%
(22/08)	117173	2675	2.28%
(23/08)	123396	10508	8.51%
(24/08)	126925	0	0
(25/08)	109135	2	0.001%
(26/08)	98838	0	0
(27/08)	63190	6	0.009%
(28/08)	72127	0	0
(29/08)	77339	0	0
(30/08)	74071	8710	11.75%
(31/08)	71559	3037	4.24%

5.2.1 *Report* de 13 de agosto de 2005

No dia 13 de agosto, foram registradas três sessões suspeitas envolvendo o servidor `ssh`. Duas eram referentes à Instituições conveniadas que utilizam esta máquina em um projeto. A outra referia-se ao servidor sendo utilizado como um *proxy* para a rede interna a partir de uma máquina externa, o que só é possível ser feito a partir de contas previamente cadastradas.

5.2.2 *Report* de 20 de agosto de 2005

Em 20 de agosto, o relatório continha 473 sessões suspeitas. As sessões com o servidor `ssh` eram legítimas, e obedeciam o padrão das sessões citadas na subseção anterior. Prosseguindo a análise, encontrou-se sessões Web normais, até o ponto em que haviam, em seguida, 10 sessões suspeitas provenientes do mesmo endereço IP e com dois intervalos de porta de origem incrementando de um em relação a sessão anterior. A parte do relatório que mostra este acontecimento pode ser vista no Apêndice A.

No mesmo intervalo de tempo, foram encontradas inúmeras sessões de um endereço IP real, estabelecendo conexões com o servidor Web e com números de porta complementares ou sobrepostos aos números de porta das sessões identificadas como suspeitas. Neste período, o servidor começa a fazer múltiplas retransmissões, e a receber vários `ACK`s duplicados, enviando `RESET`s em paralelo ao recebimento dos `ACK` do IP de origem reservado. Isto pode ser um indício de um ataque de negação de serviço.

Continuando a análise, havia no relatório 5 sessões suspeitas envolvendo outra classe de endereços reservados. Os *logs* mostram que se trata de uma varredura por meio de pacotes TCP com a *flag* `FIN` habilitada (*FIN scan*). Este tráfego pode ser melhor visto na amostra da Figura 5.3. A análise das sessões adjacentes fez com que fosse encontrado um endereço IP de origem válido, utilizando as mesmas portas que o endereço reservado no mesmo intervalo de tempo. As dezenas de retransmissões ocorridas podem se tratar de um problema momentâneo na comunicação de rede, mas os pacotes com endereço de origem forjado associados a uma máquina com IP real são, com certeza, vestígios de uma tentativa de violação da segurança da rede do Instituto.

Neste mesmo relatório foram identificadas todas as sessões relativas a Web *spiders* do Google tentando efetuar o *download* do arquivo `robots.txt` para indexar as páginas do *site* institucional.

5.2.3 *Report* de 21 de agosto de 2005

Na data de 21 de agosto, 661 sessões suspeitas foram registradas. Três delas eram referentes ao servidor de `ssh`, das quais uma foi associada a uma Instituição parceira e as outras duas eram atividades normais envolvendo máquinas internas. Estas

```
16:29:08.721019 IP 192.168.192.30.3585 > yyy.yyy.yyy.7.80: F 3783388124:
3783388124(0) ack 405027488 win 16210
16:29:08.721114 IP yyy.yyy.yyy.7.80 > 192.168.192.30.3585: R 405027488:
405027488(0) win 0

(...)

16:35:00.550399 IP 192.168.192.30.3641 > yyy.yyy.yyy.7.80: F 0:0(0) ack 1
 win 17446
16:35:00.550456 IP yyy.yyy.yyy.7.80 > 192.168.192.30.3641: R 4217235317:
4217235317(0) win 0

16:35:25.073213 IP 192.168.192.30.3592 > yyy.yyy.yyy.7.80: F 0:0(0) ack 1
win 16371
16:35:25.073274 IP yyy.yyy.yyy.7.80 > 192.168.192.30.3592: R 2260244903:
2260244903(0) win 0

16:36:34.872247 IP 192.168.192.30.3711 > yyy.yyy.yyy.7.80: F 3844902297:
3844902297(0) ack 2148286573 win 16369
16:36:34.872305 IP yyy.yyy.yyy.7.80 > 192.168.192.30.3711: R 2148286573:
2148286573(0) win 0

16:39:03.230033 IP 192.168.192.30.3582 > yyy.yyy.yyy.7.80: F 0:0(0) ack 1
win 16914
16:39:03.230126 IP yyy.yyy.yyy.7.80 > 192.168.192.30.3582: R 1425042054:
1425042054(0) win 0
```

FIGURA 5.3 - Endereço IP não roteável associado a FIN scan.

sessões foram descartadas como suspeitas.

Muitas das sessões eram referentes à tráfego `http` legítimo, fazendo *download* de arquivos de tamanho grande. Entretanto, houveram sessões suspeitas confirmadas. Na Figura 5.4, pode-se observar um tráfego estranho, de um endereço IP reservado enviando um `ACK` para o servidor Web.

Procurou-se no tráfego adjacente algo que pudesse explicar este fragmento de sessão, sendo encontrados vários pacotes de `RESET` com *timestamp* próximo de um endereço IP válido (Apêndice B). Isto pode indicar um tipo de varredura chamada de *Decoy scan*, que tenta camuflar o endereço de origem utilizando tráfego forjado.

Cinco sessões suspeitas estavam associadas a Web *spiders*, as quais tentaram fazer o *download* do arquivo `robots.txt`. Destas, duas sessões eram oriundas do *Google* e as outras três de uma universidade americana. Verificando os *logs* deste dia, estas

```
12:38:01.932017 IP xxx.xxx.40.11.1026 > yyy.yyy.yyy.20.22: S 2004042161:
2004042161(0) win 16384 <mss 1460,nop,nop,sackOK>

12:38:01.937386 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1026: S 1061229196:
1061229196(0) ack 2004042162 win 16384 <mss 1460,nop,nop,sackOK>

12:38:04.926234 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1026: S 1061229196:
1061229196(0) ack 2004042162 win 16384 <mss 1460,nop,nop,sackOK>

12:38:05.014499 IP xxx.xxx.40.11.1026 > yyy.yyy.yyy.20.22: R 2004042162:
2004042162(0) win 0

```

FIGURA 5.4 - Tráfego suspeito com endereço IP de origem não roteável.

cinco sessões que foram classificadas como suspeitas correspondiam a todas as sessões envolvendo *spiders* registradas.

Algumas sessões suspeitas deste relatório foram iniciadas da rede do *Google* ou da *Microsoft*, indexando páginas, enquanto que outras dizem respeito a sessões que terminaram abruptamente. Muitas sessões são provenientes da China, país com o qual o INPE mantém um convênio, e sua análise revelou que tratam-se de apenas *downloads* de arquivos grandes.

5.2.4 *Report* de 25 de agosto de 2005

No dia 25 de agosto, foi gerado um relatório com apenas duas sessões consideradas suspeitas. As sessões suspeitas eram referentes a acessos `ssh` no servidor Web. Ambos os endereços IP que iniciaram a conexão são de máquinas internas que estão permitidas a efetuar conexões deste tipo, com o objetivo de realizar tarefas de administração do sistema operacional ou do serviço Web. Apesar de não maliciosa, este tipo de sessão é suspeita por não acontecer freqüentemente.

5.2.5 *Report* de 27 de agosto de 2005

O relatório gerado para o dia 27 de agosto acusou 6 sessões suspeitas envolvendo o servidor de `ssh`. Destas, três eram referentes a tráfego interno, saindo do servidor de `ssh` para acessar clientes `ssh` na rede do INPE. Embora seja suspeito um servidor iniciar conexões, este serve como um *proxy* que possibilita a pessoas cadastradas acessarem computadores específicos das subredes do INPE, provenientes de uma rede externa.

Outras duas entradas eram de redes de Instituições parceiras e cadastradas que utilizam o servidor diariamente para depositar informações de projetos em conjunto com o Instituto.

A entrada restante pertencia ao portal de uma cidade adjacente. A sessão considerada suspeita pode ser vista na Figura 5.5. Foi feita uma busca nos *logs* do dia por outras entradas envolvendo este endereço IP, e encontrou-se várias situações semelhantes, como pode ser observado no Apêndice C. Este tipo de tráfego é semelhante ao tráfego ocasionado por varreduras, e as sucessivas tentativas de estabelecimento de conexão em um período de tempo curto, trocando as portas de origem, pode ser associado à coleta de informações sobre o sistema operacional.

```
12:38:01.932017 IP xxx.xxx.40.11.1026 > yyy.yyy.yyy.20.22: S 2004042161:
2004042161(0) win 16384 <mss 1460,nop,nop,sackOK>

12:38:01.937386 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1026: S 1061229196:
1061229196(0) ack 2004042162 win 16384 <mss 1460,nop,nop,sackOK>

12:38:04.926234 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1026: S 1061229196:
1061229196(0) ack 2004042162 win 16384 <mss 1460,nop,nop,sackOK>

12:38:05.014499 IP xxx.xxx.40.11.1026 > yyy.yyy.yyy.20.22: R 2004042162:
2004042162(0) win 0

```

FIGURA 5.5 - Sessão suspeita confirmada: varredura/obtenção de versão do sistema operacional.

5.2.5.1 Considerações Finais

Os testes feitos com a utilização do protótipo foram realizados pensando na redução de *logs* de forma que permitisse ao analista realizar seu trabalho. Algumas abordagens diferentes tiveram que ser tentadas, visando adequar o treinamento do protótipo com o resultado esperado. Contornou-se os problemas encontrados de forma simples, conseguindo-se gerar classificadores que reduzem a quantidade de *logs* ao mesmo tempo em que encontram atividades inicialmente suspeitas.

Todas as árvores geradas são de fácil interpretação e de tamanho adequando, sendo que a maior delas tem profundidade igual a 22 e número de folhas igual a 12. Os resultados aqui obtidos são animadores e mostram a viabilidade de implementação de um protótipo do tipo. Cabe ressaltar que os eventos encontrados associados a

varreduras não foram identificados pelo Snort, uma vez que o tráfego de alerta foi filtrado antes de os dados passarem pela árvore de decisão.

No próximo capítulo, as conclusões gerais a respeito deste trabalho, sugestões de melhorias e trabalhos futuros serão discutidas.

CAPÍTULO 6

CONCLUSÕES

6.1 Considerações Finais

No início desta dissertação, explanou-se sobre a situação a que as redes de computadores estão sujeitas atualmente, geração de registros de auditoria (*logs*), problema atribuídos à coleta, armazenamento e análise de *logs*. No decorrer do trabalho, discorreu-se sobre a importância da análise de *logs* para a segurança de sistemas de informação, bem como técnicas e ferramentas que auxiliam no cumprimento desta tarefa.

Abordagens convencionais e abordagens utilizando mineração de dados para tratamento e classificação de *logs* foram expostas, discutindo-se vantagens e desvantagens de cada método. Tendo em vista os bons resultados obtidos na classificação de *logs* por técnicas de mineração de dados, foi realizada a avaliação de alguns algoritmos clássicos para verificar sua atuação quanto aos *logs* de tráfego da rede do INPE.

Haja vista a necessidade de conjuntos de dados distintos para representar as classes nas quais se quer separar os *logs*, definiu-se uma nova abordagem para a criação de perfis ilegítimos (ou suspeitos): o uso de *logs* de *honeypots*. Devido a obtenção de resultados interessantes no processo de avaliação dos algoritmos propostos, partiu-se para a construção de um protótipo com o objetivo de verificar a adequabilidade da proposta de redução de *logs*, em um ambiente de produção.

A finalidade do protótipo inclui a redução de *logs* para quantidades que viabilizem a análise humana, ao mesmo tempo em que seja possível a identificação de tráfego suspeito baseada nas tendências de ataque contra uma rede. Estas tendências são passadas para o classificador através da observação dos *logs* de um *honeypot* situado na rede em questão. Ajustes no método de geração de perfis foram necessários para que o protótipo atingisse resultados adequados. No que diz respeito à escalabilidade, o algoritmo das árvores de decisão mostrou-se altamente eficiente e eficaz, obtendo resultados rápidos com massas de dados consideráveis e incentivando seu uso nos casos de grandes volumes de informação para análise.

6.2 Conclusões e Trabalhos Futuros

Como foi possível observar, a versão final do protótipo conseguiu prover índices de mais de 97% de redução do conjunto de *logs* originais, na maior parte dos casos. Isto significa que uma parcela muito pequena da grande massa dos *logs* de tráfego que iriam para o "arquivo morto" ainda pode ser analisada por um elemento humano. Esta análise, dado o treinamento do classificador com as tendências de tráfego normal e suspeito, pode resultar na identificação de comportamento intrusivo ou problemas no funcionamento da rede.

Com o aprimoramento da técnica e conseqüente armazenamento do conhecimento do analista para gerar árvores mais acuradas, o processo de filtragem torna-se bem mais eficaz, resultando em taxas ainda mais reduzidas de falsos positivos. Isto torna mais rápida a análise das sessões suspeitas restantes, promovendo a tomada ágil de conta-medidas e uma redução ainda maior do conjunto de *logs* originais que foram passados ao classificador.

Além dos resultados obtidos com a redução do volume de *logs* para análise, conseguiu-se ter sucesso no treinamento das árvores de decisão para classificação de tráfego, mesmo com um conjunto tão escasso de sessões oriundas do *honeypot* de baixa interatividade. Ressalta-se que não houve manipulação nos dados de treinamento utilizados no protótipo de forma a expandir sinteticamente algum dos conjunto, como foi feito com as sessões suspeitas na etapa de avaliação dos algoritmos. Para aumentar um pouco a representatividade das instâncias suspeitas, utilizou-se do artifício da diminuição da quantidade de instâncias consideradas normais, sem no entanto alterar sua variedade.

Considerando que o protótipo foi implementado para tornar possível a, por vezes, impraticável tarefa de análise de *logs*, os resultados foram promissores. A idéia por trás do protótipo é a de complementar, e de forma alguma substituir, os sistemas de detecção de intrusão existentes em uma rede. Tais sistemas utilizam seus métodos para identificar atividades intrusivas e geram alertas para o analista, sendo que o restante do tráfego é considerado normal e geralmente nunca mais é olhado. A utilização deste protótipo tem por finalidade permitir que estes *logs* restantes possam ser analisados em busca de comportamento suspeito ou novidades.

No estudo de caso foi possível observar que, além da drástica redução do conjunto

de dados para análise, algumas sessões identificadas como suspeitas realmente representavam atividades indevidas ou anormais. Isto levando-se em conta que as classificações foram feitas a partir de árvores de decisão pequenas, com um número reduzido de atributos simples, que podem ser facilmente obtidos ao se separar *dumps* de tráfego em sessões.

As sessões suspeitas confirmadas não foram detectadas pelo Snort, uma vez que o tráfego de entrada para o classificador foi previamente processado pela referida ferramenta. Além disso, o processo de classificação é bastante rápido, uma vez que a árvore já foi gerada. Há também a facilidade de modificação desta árvore, permitindo ao analista adicionar ou remover nós de acordo com seus próprios critérios para classificação de sessões.

À observação de tráfego suspeito que suscite dúvidas, como no caso da possível negativa de serviço, podem ser somadas outras abordagens presentes na rede, como a verificação dos fluxos relativos a estas sessões. Com isto, as suspeitas podem ser dissipadas ou confirmadas, mas o protótipo exerceu seu papel adequadamente. Em caso de sessões que indiquem tentativas de violação de *login*, como ataques por força bruta, pode-se checar os *logs* do sistema supostamente atacado. Desta forma, faz-se o correlacionamento de várias fontes para se obter repostas acuradas sobre os eventos ocorrendo na rede.

O uso do protótipo em conjunto com abordagens tradicionais de detecção de comportamento intrusivo pode auxiliar o analista a criar regras de controle de acesso em roteadores, regras de *firewall* e a notificar os responsáveis pelas redes que possuam máquinas causando incidentes de segurança. Sem o protótipo para reduzir os *logs* e viabilizar a análise, alguns destes incidentes passariam desapercebidos. Por exemplo: no estudo de caso foram identificadas ações de *spiders*, que são programas que buscam páginas Web para análise. Em geral, estes programas são inofensivos, como as *spiders* do Google, mas elas podem ser utilizadas para automatizar o envio de *spam*, caso encontrem formulários em páginas Web.

No caso de instituições que não armazenam *logs*, ou não dispõem de espaço suficiente para armazenar muito tempo de *logs*, o protótipo pode ser usado para gerar um conjunto reduzido de *dumps* que pode ser armazenado para análise posterior. Isto ocasiona em economia de espaço com tráfego que tem a possibilidade de ser analisado, em detrimento do armazenamento de centenas de *Megabytes* que seriam

ignorados *ad eternum*.

Assim sendo, além da redução do conjunto de *logs* e classificação de sessões de tráfego de rede, as principais contribuições deste trabalho dizem respeito à geração dos perfis de treinamento do classificador. A técnica de retirar o tráfego que causa alertas do Snort faz com que haja menos contaminação no perfil de tráfego normal, enquanto que a utilização de *logs* de *honeypots* personaliza o perfil de tráfego suspeito para uma certa rede, o que acaba refletindo o comportamento não padronizado de redes de computadores distintas.

É muito importante o fato da geração do perfil suspeito como proposto neste trabalho, uma vez que não é nem fácil obter conjuntos reais de dados para treinamento que sejam adequados para redes em geral, nem medir o nível de correção que um conjunto sintético pode prover quando um classificador for utilizado em ambientes de produção. Estes problemas estão descritos em (NONNEMAKER, 2006), que discute sobre o uso seguro de dados sintéticos para classificação e menciona o Paradoxo da Acurácia.

A despeito de todos os resultados animadores obtidos nos testes com o protótipo, este tem diversas limitações. Dadas as condições de treinamento, com pouquíssimas instâncias de sessões suspeitas e um conjunto de sessões normais com muitas instâncias, as árvores de decisão foram geradas não com a acurácia em mente, mas com a finalidade de conseguir separar os dados em duas classes. A disparidade entre os conjuntos de dados foi sentida nos dias em que os classificadores não puderam ser gerados, ou seja, todas as instâncias de treinamento foram consideradas normais. Uma solução para isso seria aumentar o conjunto de sessões suspeitas durante o treinamento, o que pode ser feito acumulando-se vários dias de sessões suspeitas provenientes de *honeypots*.

É importante notar também que, apesar de ter havido uma redução substancial da quantidade de *logs* ao final do processo, em alguns casos essa redução não foi suficiente. Um conjunto reduzido a 10% da quantidade original contendo 10 mil sessões consideradas suspeitas continua a tornar a análise impraticável em tempo hábil.

Outro ponto importante diz respeito à detecção de comportamento suspeito. Se este aspecto for considerado, as características observadas nas sessões que foram

classificadas como suspeitas no relatório final são constituídas, em sua maior porção, por falsos positivos. Quanto aos falsos negativos, não dá para se ter uma medida exata de sua quantidade no ambiente de produção, a menos que todos os *logs* sejam analisados, o que incorre no problema original. A diminuição dos falsos positivos pode ser obtida ao se manter um histórico – levantado pelo analista – das características de sessões que não devem ser consideradas suspeitas, sendo que tal histórico seria em forma de regras de decisão e entraria como um filtro pré-classificatório, retirando os falsos positivos já sabidos do conjunto de *logs* a ser passado para o classificador.

É possível estimar a taxa esperada de falsos positivos e falsos negativos com base nas matrizes de confusão de cada árvore de decisão criada. Entretanto, essas taxas podem não corresponder à realidade quando se lida com um ambiente de produção sujeito a problemas no enlace, quedas de energia, defeito em dispositivos de rede, etc.

O protótipo, da maneira que foi implementado, possui muitos processos intermediários entre a coleta de dados brutos (*dumps*) e transformação destes em um conjunto de vetores de entrada para o treinamento do classificador. Isto dificulta sua utilização para geração de modelos em tempo real, sendo necessárias mudanças profundas na abordagem de extração das sessões do tráfego de rede. Para classificação *off-line*, como apresentado neste trabalho, o protótipo é adequado.

Cabe ressaltar que, embora a metodologia implementada no protótipo possa ser utilizada para gerar árvores diárias, semanais, por hora, por período do dia, etc., os resultados dependem da representatividade dos conjuntos de dados providos. Além disso, o comportamento de uma rede pode não ser estável o suficiente para a geração de classificadores equilibrados em períodos de tempo muito curtos.

O modo de implementação também afeta o processo de ajuste do classificador. Embora seja possível modificar uma árvore existente para refletir um padrão desejado, o modelo atual cria árvores diárias, não aproveitando o conhecimento adquirido ou inserido no classificador anterior. Isto vem a ser uma vantagem no caso em que as mudanças de comportamento serão acompanhadas pelo classificador, mas pode ser uma desvantagem se o tráfego que representa falsos positivos continuar a aparecer sempre.

Concluindo, o protótipo implementado e os testes realizados mostraram que a utili-

zação de mineração de dados pode ser efetiva na resolução de problemas de classificação e análise de *logs* de tráfego, identificando eventos suspeitos enquanto provê um conjunto reduzido para o analista. Mesmo com o fato de o modelo não contemplar horários específicos na geração de perfis, os resultados obtidos foram interessantes, o que leva a conclusão de que um número pequeno de atributos consegue identificar certas atividades intrusivas.

Modificações visando a melhoria do protótipo e do projeto como um todo e direções para trabalhos futuros incluem:

- Automatização do processo de escolha do classificador para um determinado dia em que não houver *logs* ou os disponíveis estiverem corrompidos. Este processo pode ser baseado nas taxas esperadas de falsos positivos e falsos negativos obtidas durante o treinamento das árvores;
- Confecção automática de dados sintéticos confiáveis, quando houver problemas nos conjuntos de dados de treinamento que inviabilizem a construção do classificador;
- Melhorias no algoritmo de classificação e inserção de novos módulos de geração da aplicação classificadora, possibilitando que o analista escolha entre várias abordagens de mineração de dados disponíveis;
- Desenvolvimento de uma interface que torne mais agradável a apresentação dos relatórios para o analista, além de permitir a organização e manipulação dos *logs*;
- Adequação do protótipo para funcionar em ambientes distribuídos, podendo criar modelos de classificação personalizados e tratar de *logs* de redes distintas;
- Estabelecimento de métricas para avaliação de árvores de decisão, analisando o impacto de modificações do comportamento dos conjuntos de dados na estrutura das árvores;
- Análise do comportamento do algoritmo de árvores de decisão com a variação dos parâmetros utilizados e com a utilização de atributos adicionais.

REFERÊNCIAS BIBLIOGRÁFICAS

ADRIAANS, P.; ZANTINGE, D. **Data mining**. Boston, MA, USA: Addison-Wesley, 1996. 56, 73

Association for Computing Machinery (ACM). **Knowledge discovery in databases cup 1999**. 1999. Disponível em: <http://www.acm.org/sigs/sigkdd/kddcup/>. Acesso em jan. de 2007. 74

BABBIN, J.; KLEIMAN, D.; CARTER, E. F.; FAIRCLOTH, J.; BURNETT, M.; GUTIERREZ, E. **Security log management**: identifying patterns in the chaos. New York, NY, USA: Syngress, 2006. ISBN 1597490423. 28

BARBARÁ, D.; COUTO, J.; JAJODIA, S.; WU, N. Adam: a testbed for exploring the use of data mining in intrusion detection. **ACM SIGMOD record**, v. 30, n. 4, p. 15–24, Dec 2001. 64

BEALE, J.; BAKER, A. R.; CASWELL, B.; POOR, M. **Snort 2.1 intrusion detection**. New York, NY, USA: Syngress Publishing, 2004. ISBN 1931836043. 85

BISHOP, M. **Computer security**: art and science. Boston, MA, USA: Addison-Wesley Professional, 2002. ISBN 0201440997. 38

BRUCKHAUS, T. **Accuracy paradox**. Jan 2006. Disponível em: <http://en.wikipedia.org/wiki/Accuracy_paradox>. Acesso em jan. de 2007. 97

CANSIAN, A. M. **Desenvolvimento de um sistema adaptativo de detecção de intrusos e redes de computadores**. Tese (Doutor em Física Computacional) — Universidade de São Paulo (USP), São José do Rio Preto, 1997. 61

CANSIAN, A. M.; MOREIRA, E. S.; CARVALHO, A.; JR, J. M. B. Network intrusion detection using neural networks. In: INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE AND MULTIMEDIA APPLICATIONS (ICCIMA'97), 1997, Sidney, Australia. **Proceedings...** Australia: ICCIMA, 1997. p. 276–289. 66

CARVALHO, B. P. **Detecção de intrusão em redes de alta velocidade**. 107 p. Dissertação (Mestrado em Computação Aplicada) — Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2005. A ser publicada. 45

CERT. **Vulnerability discovery: bridging the gap between analysis and engineering**. 2006. Disponível em: <http://www.cert.org/archive/pdf/CERTCC_Vulnerability_Discovery.pdf>. Acesso em jan. de 2007. 27

CERT.BR. **Incidentes reportados ao CERT.br – janeiro a dezembro de 2006**. 2007. Disponível em: <http://www.cert.br/stats/incidentes/2006-jan-dec/tipos-ataque.html>. Acesso em jan. de 2007. 40

_____. **Total de incidentes reportados ao CERT.br por ano**. 2007. Disponível em: <http://www.cert.br/stats/incidentes/>. Acesso em jan. de 2007. 29

CHAN, P.; KUMAR, V.; LEE, W.; (ORGANIZERS), S. P. **ICDM workshop on data mining for computer security (workshop notes)**. 2003. Disponível em: <http://www.cs.fit.edu/~pkc/dmsec03/dmsec03notes.pdf>. Acesso em jan. de 2007. 53

CHAVES, C. H.; MONTES, A. Sistema de detecção de backdoors e canais dissimulados. In: SIMPóSIO BRASILEIRO EM SEGURANçA DA INFORMAçãO E DE SISTEMAS COMPUTACIONAIS (SBSEG 2005), 5th., 2005, Florianopolis, SC. **Anais...** Brasil: SBSEG, 2005. 67, 74

CUFF, A. **Intrusion detection terminology (part two)**. 2003. <http://www.securityfocus.com/infocus/1733>. Acesso em jan. 2007. 47

DAS, M. **How verizon cut customer churn by 0.5 per cent**. Oct 2003. Disponível em: <http://www.financialexpress.com/fe_full_story.php?content_id=43556>. Acesso em jan. de 2007. 56

DOKAS, P.; ERTOZ, L.; KUMAR, V.; LAZAREVIC, A.; SRIVASTAVA, J.; TAN, P. N. Data mining for network intrusion detection. In: NFS WORKSHOP ON NEXT GENERATION DATA MINING, 2002, Baltimore. **Proceedings...** Baltimore, MD, 2002. 60

ERTÖZ, L.; EILERTSON, E.; LAZAREVIC, A.; TAN, P.; DOKAS, P.; KUMAR, V.; SRIVASTAVA, J. Detection and summarization of novel network attacks using data mining. In: **Technical Report**. University of Minnesota, Computer Science

Department, 2003. Disponível em: <http://www.cs.umn.edu/research/MINDS/papers/raid03.pdf>. 74

ERTOZ, L.; EILERTSON, E.; LAZAREVIC, A.; TAN, P.; SRIVASTAVA, J.; KUMAR, V.; DOKAS, P. The minds - minnesota intrusion detection system. **Next Generation Data Mining**, 2004. 63

FAYYAD, U. M.; PIATETSKY-SHAPIRO, G.; SMYTH, P.; UTHURUSAMY, R. **Advances in knowledge discovery and data mining**. Cambridge, MA, USA: MIT Press, 1996. ISBN 0262560976. 54, 55

GRÉGIO, A.; DUARTE, L. O.; MONTES, A.; CANSIAN, A. M. Eficácia de honeypots no combate a worms em instituições. **Reunião do Grupo de Trabalho em Segurança de Redes (GTS)**, v. 01/05, jul 2005. 47

GRÉGIO, A.; SANTOS, R.; MONTES, A. Análise de logs: abordagens tradicionais e por data mining (mini-curso). In: SIMPóSIO SEGURANçA EM INFORMáTICA, 8., 2006, São José dos Campos. **Anais...** São José dos Campos: ITA, 2006. 81

_____. Evaluation of data mining techniques for suspicious network activity classification using honeypots data. In: SPIE SECURITY AND DEFENSE SYMPOSIUM, 2007, Orlando, FL. **Proceedings...** USA: SPIE, 2007. p. 657006. 85

HAYKIN, S. **Neural networks**: a comprehensive foundation. Boston, MA, USA: Prentice Hall, 1998. 66, 73

HOFMEYR, S. A.; FORREST, S. Architecture for an artificial immune system. **Evolutionary Computation**, v. 8, n. 4, p. 443–473, 2000. 67

HOWARD, J. D.; LONGSTAFF, T. A. A common language for computer security incidents. **Sandia Technical Report**, Oct 1998. 39

Information Week. **Tower of power**. Feb 2002. Disponível em: <http://www.informationweek.com/story/IWK20020208S0009>. Acesso em jan. de 2007. 57

KOHONEN, T. **Self-organizing maps**. New York, NY, USA: Springer, 1995. 66

KUROSE, J.; ROSS, K. **Computer networking**: a top-down approach featuring the internet. Boston, MA, USA: Addison-Wesley Professional, 2002. ISBN 0201976994. 25

LAKKARAJU, K.; YURCIK, W.; BEARAVOLU, R.; LEE, A. J. Nvisionip: an interactive network flow visualization tool for security. In: IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN, AND CYBERNETICS (SMC), 2004, The Hague, Netherlands. **Proceedings...** Netherlands: IEEE, 2004. p. 2675–2680. 62

LEE, W.; STOLFO, S. J. Data mining approaches for intrusion detection. In: USENIX SECURITY SYMPOSIUM, 7th., 1998, San Antonio, TX. **Proceedings...** USA: USENIX, 1998. 60

_____. Combining knowledge discovery and knowledge engineering to build idss. In: RECENT ADVANCES IN INTRUSION DETECTION (RAID 1999), 2., 1999, West Lafayette, Indiana. **Proceedings...** USA, 1999. 65

MAHONEY, M.; CHAN, P. Phad: packet header anomaly detection for identifying hostile network traffic. **Florida Tech Technical Report**, Apr 2001. 66

MÉ, L. Gassata, a genetic algorithm as an alternative tool for security audit trail analysis. In: RECENT ADVANCES IN INTRUSION DETECTION (RAID), 1st., 1998, Belgium. **Proceedings...** USA: Springer-Verlag, 1998. 61, 64

MENA, J. **Investigative data mining for security and criminal detection**. Burlington, MA, USA: Butterworth Heinemann, 2003. ISBN 0750676132. 60

MONTES, A.; STEDING-JESSEN, K.; HOEPERS, C. Honeypots distribuídos. **Reunião do Grupo de Trabalho em Segurança de Redes (GTS)**, v. 02/03, dez 2003. 30

MUKKAMALA, S.; JANOSKI, G.; SUNG, A. Intrusion detection using neural networks and support vector machines. In: INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS (IJCNN), 2002, Hawaii. **Proceedings...** USA: IEEE, 2002. p. 1702–1707. 61

NONNEMAKER, J. E. **The safe use of synthetic data in classification**. Aug 2006. Disponível em: <http://www.cse.lehigh.edu/~baird/Pubs/nonnemaker_proposal_06aug1.pdf>. Acesso em jan. de 2007. 97, 112

PAULA, F. S.; CASTRO, L. N.; GEUS, P. L. An intrusion detection system using ideas from the immune system. In: INTERNATIONAL CONFERENCE ON

EVOLUTIONARY COMPUTATION (IEEE ICEC), 2004, Portland. **Proceedings...** USA: IEEE, 2004. p. 1059–1056. 67

PIATETSKY-SHAPIRO, G. **From data mining to knowledge discovery: an introduction**. Oct 2003. Disponível em: <http://www.kdnuggets.com/dmcourse/other_lectures/data-mining-to-knowledge-discovery.ppt>. Acesso em jan. de 2007. 55

POSTEL, J. **RFC 791: Internet protocol**. Sep 1981. Disponível em: <http://www.faqs.org/rfcs/rfc791.html>. Acesso em jan. de 2007. 74

_____. **RFC 793: transmission control protocol**. Sep 1981. Disponível em: <http://www.faqs.org/rfcs/rfc793.html>. Acesso em jan. de 2007. 74

PROJECT, T. H. **Know your enemy**: learning about security threats. Boston, MA, USA: Addison-Wesley, 2004. ISBN 0321166469. 45

QUINLAN, J. R. **C4.5**: programs for machine learning. New York, NY, USA: Morgan Kaufmann, 1993. 73

RAMADAS, M.; OSTERMANN, S.; TJADEN, B. Detecting anomalous network traffic with self-organizing maps. In: RECENT ADVANCES IN INTRUSION DETECTION (RAID), 6th., 2003, Pittsburgh, PA. **Proceedings...** USA: Springer-Verlag, 2003. p. 36–54. 61

RANUM, M. **Artificial ignorance: how-to guide**. 1997. <http://www.ranum.com/security/computer_security/papers/ai/index.html>. Acesso em out. 2006. 44

RHODES, B.; MAHAFFEY, J.; CANNADY, J. Multiple self-organizing maps for intrusion detection. In: NATIONAL INFORMATION SYSTEMS SECURITY CONFERENCE (NISSC), 2000, Baltimore, MD. **Proceedings...** USA: NIST, 2000. 61, 67

RISH, I. An empirical study of the naive bayes classifier. In: IJCAI 2001 WORKSHOP ON EMPIRICAL METHODS IN ARTIFICIAL INTELLIGENCE, 2001, Seattle, Washington. **Proceedings...** USA, 2001. 65

SANTOS, R. **Material de referência do curso CAP-359 (Princípios e Aplicações de Mineração de Dados)**. 2006. Disponível em:

<http://www.lac.inpe.br/~rafael_santos/CAP/cap359-2006.html>. Acesso em jan. de 2007. 73

SAS. **Success stories**. 2006. Disponível em: <http://www.sas.com/success/technology.html>. Acesso em jan. de 2007. 55, 56

SCHMIDT, K. J. Threat analysis using log data. **INSECURE Magazine**, n. 5, p. 17–26, Jan 2006. Disponível em: <http://www.insecuremag.com>. 28, 35

SPITZNER, L. **Honeypots**: tracking hackers. Boston, MA, USA: Addison-Wesley, 2003. ISBN 0321108951. 45

STALLINGS, W. **Network security essentials**: applications and standards. Boston, MA, USA: Pearson Education, 2002. ISBN 0130351288. 47

STATSOFT. **Statistica recent success stories**. 2006. Disponível em: <http://www.statsoft.com/company/success_stories/success_stories.html>. Acesso em jan. de 2007. 56

STEARLEY, J. **Sisyphus: an event log data-mining toolkit**. 2006. <http://www.cs.sandia.gov/sisyphus/>. Acesso em nov. 2006. 61

STEVENS, R. **TCP/IP illustrated vol. 1**: the protocols. Boston, MA, USA: Addison-Wesley, 1994. 73

STOLFO, S. J.; FAN, W.; LEE, W.; PRODROMIDIS, A.; CHAN, P. K. Cost-based modeling for fraud and intrusion detection: results from the jam project. In: 2000 DARPA INFORMATION SURVIVABILITY CONFERENCE AND EXPOSITION (DISCEX '00), 2000, Los Alamitos, CA. **Proceedings...** USA: IEEE, 2000. p. 1130. 66

STOLFO, S. J.; LEE, W.; CHAN, P. K.; FAN, W.; ESKIN, E. Data mining-based intrusion detectors: an overview of the columbia ids project. **SIGMOD Record**, v. 30, n. 4, p. 5–14, Dec 2001. 28

STOLL, C. **The cuckoo's egg**: tracking a spy through the maze of computer espionage. Westminster, MD, USA: Doubleday, 1989. ISBN 0385249462. 26

TANENBAUM, A. **Computer networks. 4 ed.** Boston, MA, USA: Prentice Hall, 2003. 74

VALDES, A.; SKINNER, K. Probabilistic alert correlation. In: RECENT ADVANCES IN INTRUSION DETECTION (RAID), 4th., 2001, Davis, CA. **Proceedings...** USA: Springer-Verlag, 2001. p. 54–68. 60

VERWOERD, T.; HUNT, R. Intrusion detection techniques and approaches. **Computer Communications**, v. 25, n. 15, p. 1356–1365, Sep 2002. 53

WASSERMAN, M. **Mining data**. 2000. Disponível em: <http://www.bos.frb.org/economic/nerr/rr2000/q3/mining.htm>. Acesso em jan. de 2007. 56

Winter Corporation. **2005 topten award winners**. Feb 2006. Disponível em: <http://www.wintercorp.com/VLDB/2005_TopTen_Survey/TopTenWinners_2005.asp>. Acesso em jan. de 2007. 57

WITTEN, I. H.; FRANK, E. **Data mining**: practical machine learning tools and techniques with java implementations. New York, NY, USA: Morgan Kaufmann Publishers, 2000. 73

ZURUTUZA, U.; URIBEETXEBERRIA, R. Revisión del estado actual de la investigación en el uso de data mining para la detección de intrusiones. In: SIMPOSIO ESPAñOL SOBRE SEGURIDAD INFORMATICA (CEDI'05), 1., 2005, Granada. **Anais...** España: CEDI, 2005. 63

APÊNDICE A

PARTE DO RELATÓRIO DE 20/08/2005

Na Figura A.1, é possível ver a parte do relatório que indica sessões suspeitas envolvendo máquinas com endereços IP de origem reservados.

```
####################
# Date: 2005-Aug20 #
####################

#########################################################

253 unique IP addresses generated 473 suspicious sessions.

#########################################################
Client IP: 172.16.81.70      Client port: 1074
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1075
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1076
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1077
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1108
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1109
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1110
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1111
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.16.81.70      Client port: 1112
Server IP: yyy.yyy.yyy.7      Server port: 80
#########################################################
Client IP: 172.171.32.9      Client port: 1392
Server IP: yyy.yyy.yyy.7      Server port: 80
```

FIGURA A.1 - Sessões suspeitas contíguas com endereço IP de origem não roteável.

APÊNDICE B

TRÁFEGO SUSPEITO EM 21/08/2005

Na Figura B.1 tem-se o tráfego do IP real possivelmente associado a uma das sessões que foi considerada suspeita no dia 21 de Agosto, o que serviria para confirmar a suspeita sobre a sessão em questão.

```
19:57:08.480231 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 80385
win 8760
19:57:08.480238 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123147708:
2123147708(0) win 0

19:57:08.516212 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 81409
win 7736
19:57:08.516246 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123148732:
2123148732(0) win 0

19:57:08.589872 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 82433
win 8760
19:57:08.589920 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123149756:
2123149756(0) win 0

19:57:08.609491 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 82433
win 8760
19:57:08.609540 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123149756:
2123149756(0) win 0

19:57:08.674369 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 83457
win 7736
19:57:08.674415 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123150780:
2123150780(0) win 0

(...)

19:57:08.995410 IP xxx.xxx.56.135.1034 > yyy.yyy.yyy.7.80: . ack 87553
win 8760
19:57:08.995459 IP yyy.yyy.yyy.7.80 > xxx.xxx.56.135.1034: R 2123154876:
2123154876(0) win 0
```

FIGURA B.1 - Tráfego associado a pacotes suspeitos do dia 21 de agosto de 2005.

APÊNDICE C

TRÁFEGO SUSPEITO EM 27/08/2005

Na Figura C.1, o comportamento do tráfego indica varredura, podendo ser caracterizado como coleta da versão do sistema operacional.

```
12:38:01.932024 IP xxx.xxx.40.11.1028 > yyy.yyy.yyy.20.22: S 3338559404:
3338559404(0) win 16384 <mss 1460,nop,nop,sackOK>
12:38:01.937472 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1028: S 480998601:
480998601(0) ack 3338559405 win 16384 <mss 1460,nop,nop,sackOK>
12:38:04.926245 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1028: S 480998601:
480998601(0) ack 3338559405 win 16384 <mss 1460,nop,nop,sackOK>
12:38:10.927155 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1028: S 480998601:
480998601(0) ack 3338559405 win 16384 <mss 1460,nop,nop,sackOK>
12:38:22.929019 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1028: S 480998601:
480998601(0) ack 3338559405 win 16384 <mss 1460,nop,nop,sackOK>
12:38:46.932723 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1028: S 480998601:
480998601(0) ack 3338559405 win 16384 <mss 1460,nop,nop,sackOK>

12:40:36.909779 IP xxx.xxx.40.11.1030 > yyy.yyy.yyy.20.22: S 1400148276:
1400148276(0) win 16384 <mss 1460,nop,nop,sackOK>
12:40:36.912648 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1030: S 2937208221:
2937208221(0) ack 1400148277 win 16384 <mss 1460,nop,nop,sackOK>
12:40:36.958271 IP xxx.xxx.40.11.1030 > yyy.yyy.yyy.20.22: R 1400148277:
1400148277(0) win 0

12:40:36.957275 IP xxx.xxx.40.11.1031 > yyy.yyy.yyy.20.22: S 3200644453:
3200644453(0) win 16384 <mss 1460,nop,nop,sackOK>
12:40:36.957416 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1031: S 4053884694:
4053884694(0) ack 3200644454 win 16384 <mss 1460,nop,nop,sackOK>
12:40:39.950150 IP yyy.yyy.yyy.20.22 > xxx.xxx.40.11.1031: S 4053884694:
4053884694(0) ack 3200644454 win 16384 <mss 1460,nop,nop,sackOK>
12:40:40.051840 IP xxx.xxx.40.11.1031 > yyy.yyy.yyy.20.22: R 3200644454:
3200644454(0) win 0
```

FIGURA C.1 - Tráfego com o mesmo IP de origem de uma sessão suspeita em 27/08/2005.

ANEXO A

E-MAIL DE MARTIN ROESCH

Mensagem retirada da lista *"Snort-users"*, <http://archives.neohapsis.com/archives/snort/2005-07/0100.html>.

Re: [Snort-users] snort_decoder

From: Martin Roesch (roesch\@sourcefire.com)
Date: Sun Jul 17 2005 - 20:53:39 CDT

Actually, they're protocol "anomalies", which is a great illustration of the point that protocol anomaly detection is relies on people to care a whole lot about stuff that doesn't mean anything to most people...

-Marty

On Jul 17, 2005, at 7:26 PM, Joel Esler wrote:

> No, they are decoder "errors" telling you that a packet that has
> "tcp options" "with bad lengths" has been found and that (maybe
> another packet) that the tcp options have been truncated.
>
> Most people I know shut these off.
>
> You can find out to shut these off in the snort.conf or in the
> snort manual.
>
> joel
>
>
> On Jul 17, 2005, at 4:17 PM, Angelita de Cássia Corrêa wrote:
>
>> Do these alerts mean false positives?
>>

>> (snort_decoder): Tcp Options found with bad lengths
>> (snort_decoder): Truncated Tcp Options
>>
>>
>> Thanks
>>
>>
>>
>

--
Martin Roesch - Founder/CTO, Sourcefire Inc. - +1-410-290-1616
Sourcefire - Network Defense for the Real World - http://
www.sourcefire.com
Snort: Open Source Intrusion Detection and Prevention - http://
www.snort.org

ANEXO B

E-MAIL SOBRE ATAQUE AO SNORT

Mensagem retirada da lista *"Bleeding-sigs"*, <http://lists.bleedingsnort.com/pipermail/bleeding-sigs/2005-October/000953.html>.

[Bleeding-sigs] Snort-signature for snort DOS attack version -via mal formed sack!
Sumit Siddharth sumit.siddharth at gmail.com
Tue Oct 18 13:28:09 EST 2005

Hi List,
I have written a signature to detect a dos attack on snort versions <2.4.1 (http://www.frsirt.com/english/advisories/2005/1721).

The signature as such has no false positives but an attacker can always as by pass this by changing parameters like src port etc, .However this will be useful when the default exploit code(snortrigger.c) is run without editing.

This exploit code sends a wrong tcp checksum and hence when snort is running with default parameters , it will just drop the packet.However ,sometimes the TCP checksum verification is disabled when snort is monitoring a heavy traffic in order to reduce the overhead.

This signature will work when snort is running with -k notcp switch which turns off tcp checksum verification.The exploit sends a malformed sack and so the snort_decoder alert "Tcp Options found with bad lengths" is also received .

Comments are invited
Thanks
Sumit

Here is the signature

```
:-
alert tcp $EXTERNAL_NET 31337 -> $HOME_NET 64876 (msg:"malformed Sack
--SnortDoS-by-$um$id";seq:0;ack:0;window:65535;dsize:0;sid:5000009211;
rev:1;classtype: snort-DOS attempt;)

root at nfrbc root]# snort -c /snort/snort-2.4.1/etc/snort.conf -k notcp
-A console

10/17-17:10:57.285982  [**] [1:2147483647:1] malformed Sack --Snort DoS
-by-$id
[**] [Classification: Attack on snort version<2.4.1] [Priority: 2] {TCP}
200.31.XX.XXX:31337 -> 192.168.0.YY:64876
10/17-17:10:57.285982  [**] [116:54:1] (snort_decoder): Tcp Options found
with bad lengths [**] [Classification: Attack on snort version<2.4.1]
 [Priority: 2]
{TCP} 200.31.XX.XXX:31337 -> 192.168.0.YY:64876

--
Sumit Siddharth
```

ANEXO C

E-MAIL SOBRE DECODER ERRORS

Mensagem retirada da lista *"Snort-users"*, <http://www.mcabee.org/lists/snort-users/Apr-03/msg01145.html>.

Re: [Snort-users] (snort_decoder): Truncated Tcp Options

* Date: Sun, 27 Apr 2003 08:00:27 -0400
* From: MH <procana@xxxxxxxxxxxxx>
* To: Jason Beveridge <jason@xxxxxxxxx>, snort-users@xxxxxxxxxxxxxxx
* Subject: Re: [Snort-users] (snort_decoder): Truncated Tcp Options

Hi Jason,

What the Truncated TCP options means is that a certain tcp option was set in the segment (identified by an option "Kind") but did not use a corresponding length or reported an incorrect length.
For example if a maximum segment size (MSS) option, kind = 2, is used it is followed by the length of that option including that option's data (Length = 4) . This way the stack knows to look at 4 bytes total for this particular option to find the option's data.
The packet trace for an MSS of 1460 might look like this ... 02 04 05 b4 ...

Take a look at your snort dump or a packet trace that tripped this alert and look for the offending "Kind" of option that was set. Next to that you will see what it is reporting as the length of the option. The reported length would place the data for that option beyond the allotted space to the options within the segment. Reference the parameters list here: http://www.iana.org/assignments/tcp-parameters

Clear as mud right?

You can turn this off within your snort.conf file by adding the line "config disable_tcpopt_alerts"

Hope this helps,
Mike
-
( ) ASCII ribbon campaign
X against HTML email
/ \

At 04:53 PM 4/26/2003 -0400, Jason Beveridge wrote:

Hi, I am a newbie. I keep getting a lot of alerts listed as:
(snort_decoder): Truncated Tcp Options.

There's no snort ID for them - it seems they are junk. What is this and how can I get rid of it? Any info is appreciated.

Jason

PUBLICAÇÕES TÉCNICO-CIENTÍFICAS EDITADAS PELO INPE

Teses e Dissertações (TDI)

Teses e Dissertações apresentadas nos Cursos de Pós-Graduação do INPE.

Notas Técnico-Científicas (NTC)

Incluem resultados preliminares de pesquisa, descrição de equipamentos, descrição e ou documentação de programa de computador, descrição de sistemas e experimentos, apresentação de testes, dados, atlas, e documentação de projetos de engenharia.

Propostas e Relatórios de Projetos (PRP)

São propostas de projetos técnico-científicos e relatórios de acompanhamento de projetos, atividades e convênios.

Publicações Seriadas

São os seriados técnico-científicos: boletins, periódicos, anuários e anais de eventos (simpósios e congressos). Constam destas publicações o Internacional Standard Serial Number (ISSN), que é um código único e definitivo para identificação de títulos de seriados.

Pré-publicações (PRE)

Todos os artigos publicados em periódicos, anais e como capítulos de livros.

Manuais Técnicos (MAN)

São publicações de caráter técnico que incluem normas, procedimentos, instruções e orientações.

Relatórios de Pesquisa (RPQ)

Reportam resultados ou progressos de pesquisas tanto de natureza técnica quanto científica, cujo nível seja compatível com o de uma publicação em periódico nacional ou internacional.

Publicações Didáticas (PUD)

Incluem apostilas, notas de aula e manuais didáticos.

Programas de Computador (PDC)

São a seqüência de instruções ou códigos, expressos em uma linguagem de programação compilada ou interpretada, a ser executada por um computador para alcançar um determinado objetivo. São aceitos tanto programas fonte quanto executáveis.