UNIVERSIDADE PRESBITERIANA MACKENZIE
CURSO DE PÓS-GRADUAÇÃO
EM ENGENHARIA ELÉTRICA

DETECÇÃO DE INTRUSÃO EM REDES DE COMPUTADORES COM ESTATÍSTICAS PDF E CLASSIFICADORES NEURAIS

Leandro Silva Jorge

Orientador: Prof. Dr. Nizam Omar

São Paulo

2006

Leandro Silva Jorge

DETECÇÃO DE INTRUSÃO EM REDES DE COMPUTADORES COM ESTATÍSTICAS PDF E CLASSIFICADORES NEURAIS

Orientador: Prof. Dr. Nizam Omar

Dissertação de Mestrado Apresentada ao Curso de Pós-Graduação em Engenharia Elétrica Como Parte das Exigências Para a Obtenção do Grau de Mestre em Engenharia Elétrica, Área de Concentração em Engenharia de Computação.

São Paulo

2006

AGRADECIMENTOS

Ao meu orientador, Prof. Dr. Nizam Omar, pela competente orientação, dedicação e paciência.

Ao Prof. Dr. Wagner Zucchi, pela atenção e pelas contribuições para a realização deste trabalho.

À minha mulher, Patricia, por todo o apoio e compreensão.

Aos meus amigos da ITGROUP, pelo seu apoio e por toda a segurança e confiança transmitida para que eu pudesse executar este trabalho.

DEDICATÓRIA

Dedico esta Dissertação aos meus pais, Jorge e Silvia, pelo exemplo de vida, carinho e dedicação que sempre tiveram comigo.

Sumário

Índice de Figuras

Índice de Tabelas

Resumo

Este trabalho é um estudo sobre arquiteturas, métodos e resultados na detecção de intrusão em redes de computadores, no reconhecimento de ataques de negação de serviços (*DoS – Denial-of-Service*). As técnicas empregadas foram orientadas a pacotes, em janelas de observação, utilizando processos de detecção de intrusão hierárquicos, em múltiplas camadas, baseados em anomalias de rede, monitorando vários parâmetros de tráfego de rede simultaneamente. Foram utilizadas funções densidade de probabilidade (*PDF – Probability Density Functions*), estatisticamente comparadas com funções de referência de comportamento normal, utilizando-se de métricas de similaridade, combinando os resultados em vetores de status de anomalias que foram classificados por rede neurais. Dois conjuntos de dados foram utilizados para testar o desempenho dos métodos adotados: dados de simulações *OPNET* e dados de avaliação de detecção de intrusão *DARPA98*. Os resultados demonstraram detecção confiável de ataques *DoS* com grande precisão e taxas muito baixas de alarmes falsos nos conjuntos de dados analisados.

Palavras chave – Detecção de Intrusão, Redes de Computadores, Função Densidade de Probabilidade, Redes Neurais.

Abstract

This work is a study about architectures, methods and results on computer networks intrusion detection to avoid Denial-of-Services attacks. The main techniques used were: packet-oriented window observation, structured hierarchical processes, anomaly based network intrusion detection by monitoring several network traffic parameters simultaneously. The Probability Density Function (PDF) has been used and statistically compared it to normal behavior referenced functions using similarity metrics. The results have been combined into an anomaly status vector classified by a neural network classifier. Two data sets have been tested to measure the performance of the proposed approach: the OPNET simulations data and the DARPA98 intrusion detection evaluation data. The results showed reliable detection in DoS attacks with high accuracy and very low false alarm rates on all data sets.

Keywords – Intrusion Detection, Computer Networks, Density Probability Function, Neural Networks.

1 Introdução

1.1 Introdução à Detecção de Intrusão

A onipresença da Internet tem permitido aos seus atacantes elaborar ameaças sérias à segurança da infra-estrutura de computadores e à integridade das informações importantes. Intrusões em computadores na forma de uma série de atividades maliciosas têm como alvos típicos os sistemas ou os serviços que um hospedeiro fornece.

Estes ataques geralmente procuram obter privilégios não autorizados, reduzindo ou até mesmo bloqueando a disponibilidade de serviços e podem ser classificados dentro das seguintes categorias [Weber 1998]:

Tabela 1 – Categorias de Ataques.

TIPOS DE ATAQUES	DESCRIÇÃO
Reconhecimento	São ações iniciadas por atacantes para analisar uma rede "vítima" à procura de servidores vulneráveis e possíveis pontos de penetração.
Negação de Serviços (*DoS - Denial-of-Services*)	São tentativas destrutivas de interrupção ou degradação dos serviços fornecidos por um sistema de tal modo que seus usuários legítimos têm o acesso negado a estes serviços.
Aumento de Privilégios	São ações onde um atacante aumenta ilegalmente seus níveis de privilégios para acessar e controlar o sistema "vítima".
Interceptação e Alteração de Dados	São atividades que têm como objetivo interceptar e alterar informações importantes, sem a devida autorização para isso.
Uso do Sistema	São ações para capturar o sistema "vítima" com o objetivo de algum outro uso não autorizado, como por exemplo, converter um hospedeiro em um servidor FTP para armazenar músicas pirateadas, ou utilizar o sistema como um ponto de partida no disparo de ataques para outros sistemas.

Os sistemas de detecção de intrusão, conhecidos como *IDS - Intrusion Detection Systems,* são projetados para detectar de modo automático estas atividades maliciosas em um computador ou em uma rede de computadores.

A detecção de intrusão tem sido uma área ativa de pesquisas por mais de duas décadas desde que James Anderson publicou seu primeiro trabalho sobre segurança de computadores [Anderson 1980], como em [Tan e Maxion 2005].

Em 1987, Dorothy Denning [Denning 1987] estabeleceu uma metodologia estruturada para detectar intrusões baseadas em computadores. As premissas básicas de detecção de intrusão adotadas neste estudo são:

- O comportamento de usuários e programas é observável e pode ser modelado a partir de vários tipos de dados de auditoria, como por exemplo, a partir dos registros de sistema ou do tráfego da rede;

- Através do ajuste fino de um sistema, as atividades intrusivas podem ser diferenciadas e identificadas das atividades normais.

As técnicas de detecção de intrusão podem ser divididas em dois métodos complementares distintos:

- **Detecção de uso indevido:** Modelam ataques conhecidos e podem analisar dados de auditoria para a ocorrência destes padrões específicos [Vigna e Kemmerer 1998] e [Stolfo e Mok 1999].

- **Detecção de anomalias:** Mapeiam as intrusões através da observação de desvios significativos no comportamento típico ou esperado dos sistemas ou dos usuários [Javitz e Valdes 1993] e [Ghosh, Wanken e Charron 1998].

A maioria dos sistemas de detecção de intrusão é desenvolvida a partir de pelo menos um destes métodos, ou dos dois métodos simultaneamente. Por exemplo, o SNORT [Beale e Foster 2003], o Bro [Paxson 1999] e o JAM [Stolfo e Mok 1999] são sistemas de detecção de uso indevido. Projetos como o IDES [Lunt 1998] e o INBOUND [Tjaden, Welch et al. 2000] detectam ataques baseados em anomalias; alguns outros sistemas, como o NIDES [Valdes Anderson 1995] e o CMDS [Proctor 1994], utilizam simultaneamente técnicas de uso indevido e anomalias para a detecção de ataques.

1.2 Metodologia deste Estudo

Este estudo descreve os estudos realizados através da aplicação de estatísticas PDF e classificação de redes neurais nas áreas de detecção de anomalias de intrusão. Mais especificamente, este estudo tem o objetivo de responder as seguintes questões:

- As estatísticas PDF podem ser utilizadas para a detecção de anomalias de intrusão?

- Qual é o desempenho obtido por um IDS através da utilização de estatísticas PDF em sistemas de detecção de intrusão de anomalias?

- Como os diferentes algoritmos de classificação se comportam na detecção de ataques baseados em redes?

- Como podem ser identificados ataques utilizando-se estatísticas PDF e classificadores neurais?

Muitos sistemas de detecção de intrusão contemporâneos modelam as atividades dos usuários e dos atacantes utilizando contadores estatísticos sobre a utilização do sistema e as freqüências de eventos relevantes.

Este estudo propõe uma técnica para representar as estatísticas no formato de Funções Densidade de Probabilidade (PDF) para comparar os parâmetros observados com os modelos de referência utilizando métricas de similaridade PDF, e classificar as distâncias de similaridade medidas utilizando-se redes neurais.

Um protótipo de detecção de intrusão foi elaborado para validar estes métodos propostos utilizando-se modelos estatísticos PDF e classificadores neurais para detectar ataques de negação de serviços (*DoS - Denial-of-Service*).

1.3 Conceitos Básicos

Alguns termos básicos de detecção de intrusão, que são muito utilizados neste estudo, são apresentados nesta Seção. Os critérios de avaliação de sistemas de detecção de intrusão são descritos a seguir.

1.3.1 Terminologia

Nesta Secção, apresentamos os termos mais utilizados neste estudo, acompanhados de uma descrição dos mesmos:

Intrusão: Conjunto de ações que tentam comprometer a integridade, a confidência, ou a disponibilidade de um recurso [Heady et al. 1990].

Evento: Unidade de análise, ou unidade granular, de um IDS (Veja a Seção 2.5 para mais detalhes). Um evento pode ser definido como todos os pacotes que são

observados dentro de uma janela de tempo, uma transição de estado de sessão, ou um alerta gerado.

Detecção de Intrusão: Processo automático de detectar e emitir alarmes quando uma intrusão ocorre.

Cenário de Ataque: Seqüência de ataques que um atacante dispara para obter determinados propósitos.

Alerta: Mensagem que um IDS gera quando detecta uma intrusão em andamento.

Correlação de Alerta: Procedimento que automaticamente correlaciona alertas baseados em suas similaridades e também reconstrói os cenários de ataque para que mais informações contextuais e de ambiente possam ser fornecidas aos administradores.

Estatísticas PDF: Conjuntos de algoritmos estatísticos sobre como representar a distribuição de parâmetros descritivos e representativos do sistema monitorado e como estatisticamente comparar duas distribuições.

Classificador de Detecção: Algoritmo que um sistema de detecção de intrusão utiliza para classificar os dados de entrada em uma categoria normal ou anômala, baseando-se no conhecimento obtido através do treinamento de dados.

1.3.2 Critérios de Avaliação

O desempenho de um sistema de detecção de intrusão é geralmente avaliado através dos seguintes parâmetros:

- **TVP - Taxa de Verdadeiro-Positivo (ou taxa de detecção):** É a taxa que um evento de ataque irá detectar. Em um sistema de detecção de intrusão ideal é esperado operar com taxas de detecção tão próximas de 1 quanto possível.

- **TFP - Taxa de Falso-Positivo (ou taxa de alarme falso):** É a taxa em que eventos normais serão erroneamente classificados como ataques. Em um sistema de detecção de intrusão ideal é esperado operar com taxas de alarmes falsos tão próximos de zero quanto possível.

Em geral não é possível obter simultaneamente uma TVP de valor 1 e uma TFP de valor 0. Portanto, para um sistema de detecção de intrusão com parâmetros de detecção ajustáveis, um gráfico TFP x TVP também é utilizado para avaliar o desempenho do sistema. Este gráfico é conhecido como *ROC – Receiver Operating Characteristic*. A área abaixo do gráfico representa a probabilidade de distinguir corretamente um par (normal e ataque).

Um gráfico ROC simples é ilustrado na Figura 1. O ponto no canto superior esquerdo corresponde a um ótimo parâmetro de detecção, com alta taxa de detecção e baixa taxa de alarme falso.

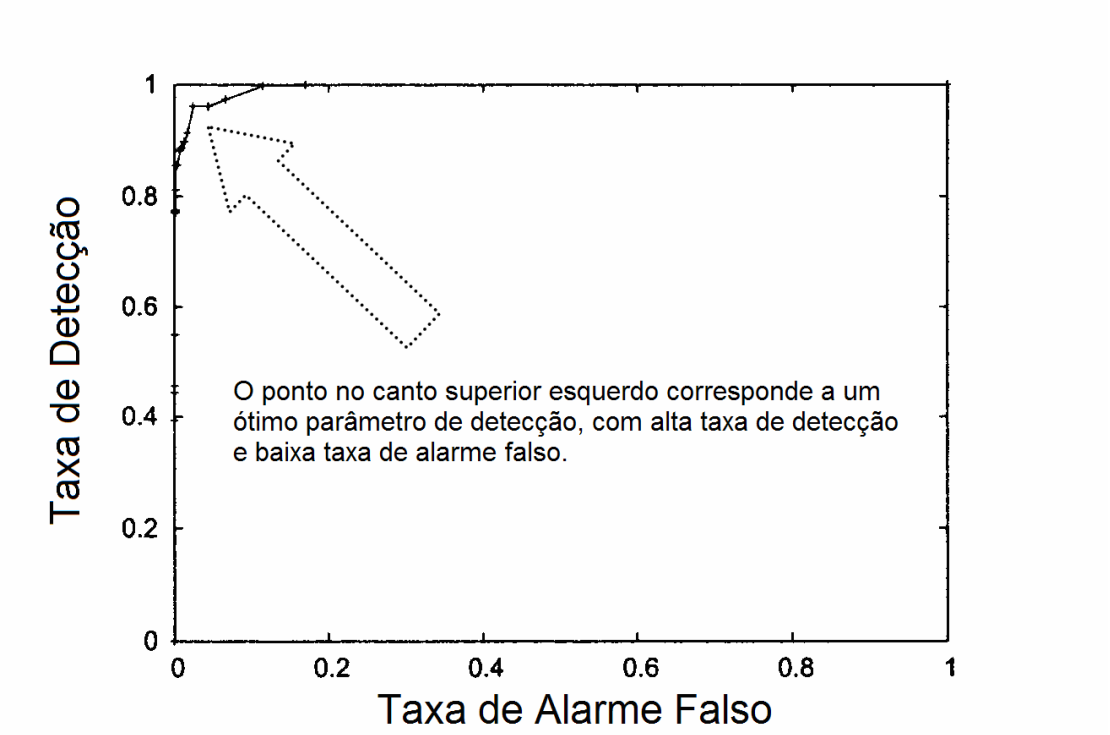

Figura 1 – Exemplo de um gráfico ROC.

Neste estudo, outro parâmetro também é utilizado para representar o desempenho do sistema:

- **Taxa de Não-Classificação (Erro):** é a probabilidade de uma observação ser erroneamente classificada. Ela é calculada como a relação entre o número de não-classificações, incluindo ambos os falsos positivos e os falsos negativos, e o numero total de observações.

A Taxa de Não-Classificação pode ser considerada como a métrica que descreve o desempenho geral da classificação. Ela pode ser utilizada como a função-objetivo para avaliar todos os classificadores testados.

1.4 Estrutura deste Estudo

O restante deste estudo está organizado da seguinte maneira descrita a seguir.

O Capítulo 2 introduz os conceitos básicos e mais complexos da segurança da informação, detecção de intrusão e correlação de alertas, além de questões relativas à avaliação de detecção de intrusão.

O Capítulo 3 descreve a arquitetura dos sistemas analisados e os modelos estatísticos empregados. Os resultados de classificação em três conjuntos de dados de testes são apresentados neste Capítulo.

O Capítulo 4 apresenta a pesquisa para identificar e selecionar conjuntos de características importantes para o sistemas detectarem ataques *DoS*.

O Capítulo 5 exibe os estudos de vários métodos para aperfeiçoar o desempenho da detecção dos sistemas, que incluem esquemas de partições PDF, métricas de distribuição de similaridades e algoritmos de classificação, além da aplicação da compressão de ondas (*wavelet compression*).

O Capítulo 6 apresenta dois métodos diferentes para treinar os classificadores neurais baseados em informações de ataque e tráfego de fundo (*background traffic*) coletado em uma rede de testes controlada e também em uma rede de produção.

O Capítulo 7 apresenta as conclusões deste estudo.

O Apêndice I lista as características estatísticas monitoradas pelos sistemas analisados e fornece uma descrição de cada característica.

O Apêndice II demonstra a utilização do OPNET nas simulações de detecção de intrusão.

2 Revisão de Literatura Correlata

2.1 Segurança da Informação

Uma vez que a computação e os recursos de informação também são patrimônios valiosos de uma organização, os mesmos deveriam ser protegidos de maneira apropriada. Os principais atributos da segurança de redes e de computadores são [Goolman 1999]:

- **Confidencialidade:** Mantém informações importantes apenas para pessoal autorizado.

- **Integridade:** Protege informações importantes contra modificações não autorizadas.

- **Disponibilidade:** Evita a manutenção não autorizada de informações e recursos.

Uma outra característica relativa ao uso da informação foi acrescentada por [Jonsson 1998]:

- **Registro de Responsabilidade (*Accountability*):** Evita o uso não autorizado dos recursos de computação

O objetivo da segurança da informação é evitar que estes quatro aspectos de segurança sejam violados. Uma intrusão baseada em computador é uma série de atividades maliciosas que têm como objetivo um computador, uma rede ou serviços para comprometer a segurança.

Seis métodos gerais, não exclusivos, de técnicas contra intrusão para proteger uma organização de ataques são citados por [Halme e Bauer 1995]:

- **Preempção:** Atacar a ameaça de segurança antes que ela tenha a chance de disparar seu ataque. Esta medida pró-ativa é difícil de ser praticada uma vez que muitos ataques não podem ser previstos.

- **Prevenção:** Eliminar ou reduzir de modo significativo todas as possibilidades de sucesso em uma intrusão específica. Exemplos deste método são as identificações e autorizações de usuário, controle de acesso e criptografia de informações. Estes métodos de prevenção são necessários, mas não necessariamente suficientes, uma vez que, na medida em que os sistemas têm se tornado mais complexos, poderão

existir fraquezas exploráveis nos sistemas, originadas à partir de vários erros de projeto e programação (*bugs*). A grande quantidade destes *bugs* e a proliferação de vírus que visam estes mesmos erros de projeto e programação nos diversos sistemas operacionais são exemplos clássicos.

- **Desencorajamento:** Intimidar os atacantes para que os mesmos desistam de seus ataques, através de aumento de riscos e conseqüências negativas. É obvio que se os recursos protegidos são muito importantes, ou se as violações são difíceis de serem descobertas, os atacantes podem não se assustar tão facilmente.

- **Detecção:** Identificar tentativas de intrusão, de modo que uma resposta apropriada possa ser invocada. Também podem ser fornecidas informações importantes sobre os incidentes para os administradores, para a recuperação de danos, a identificação e a captura do atacante.

- **Deflexão:** Encaminhar o invasor a um "pote de mel" previamente projetado e controlado para que nenhum dano real possa ocorrer, e iludi-lo de que foi bem sucedido. A maior dificuldade neste método é configurar o "pote de mel" do modo mais realístico possível para enganar um atacante experiente.

- **Contramedidas:** Ativamente responder às intrusões enquanto elas estão ocorrendo. As práticas mais comuns incluem "bloquear o tráfico no *firewall*", "desabilitar a conta do usuário" e, em alguns casos extremos, "executar uma parada completa do sistema (*shutdown*) temporariamente". A efetividade destas contramedidas é muito dependente da precisão da detecção de intrusão. Uma ação errônea em um usuário normal pode negar ao usuário o acesso legítimo aos serviços.

Estas técnicas de contramedidas podem cooperar umas com as outras na forma de proteção em múltiplas camadas nos recursos do sistema, como visto na Figura 2.

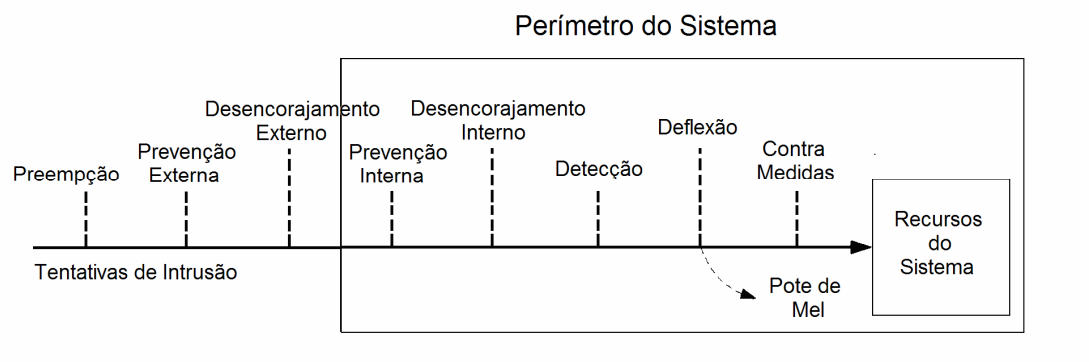

Figura 2 – Técnicas de Contra-Intrusão

Nesta taxonomia, muitos dos sistemas de detecção de intrusão estão quase que exclusivamente na categoria de detecção, embora alguns sistemas tenham iniciado a implantação de respostas automáticas.

2.2 Diagrama Genérico de Intrusão de Sistemas

Embora os sistemas de detecção de intrusão possam variar significante uns dos outros em termos de fontes de dados, características de extração e classificação, eles compartilham aspectos comuns de funcionalidade e estrutura. Para distinguir as diferenças entre atividades normais ou maliciosas, um sistema de detecção de intrusão tem de ser capaz de abstrair as atividades de um usuário para um conjunto de características estatísticas. Também é necessário manter uma base de conhecimento de padrões normais e de ataques. Um modelo genérico de sistemas de detecção de intrusão é apresentado na Figura 3.

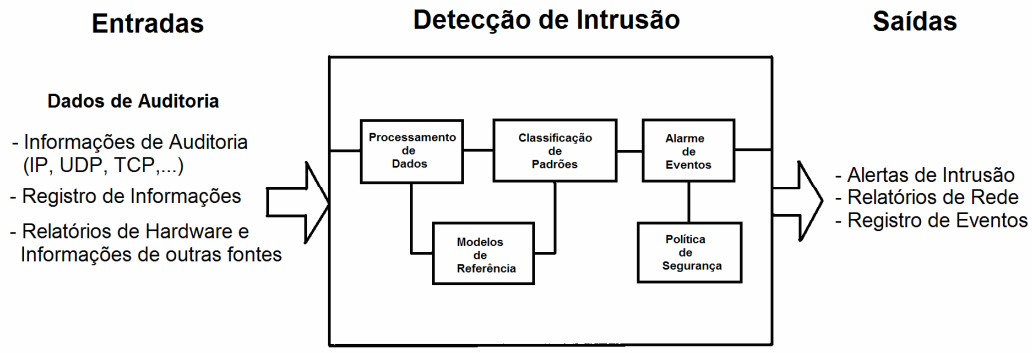

Figura 3 - Diagrama genérico de um sistema de detecção de intrusão.

Os dados de entrada de um sistema de detecção de intrusão podem ser obtidos de várias fontes: tráfego de rede, registros de sistemas e aplicações, informações de gerenciamento de rede, etc.

Estes dados de entrada são processados e reduzidos a conjuntos menores de características especificas.

Um banco de dados de modelos de referência mantém as informações sobre assinaturas de ataques conhecidos de uso indevido dos sistemas, ou padrões normais conhecidos de atividades de usuários para anomalias nos sistemas.

O mecanismo de classificação determina se dados recebidos de um processador de dados realmente contém atividades maliciosas.

Alarmes são disparados quando atividades suspeitas são detectadas. Os administradores também podem configurar os parâmetros de detecção e especificar as diretrizes de segurança para controlar as reações dos sistemas em diferentes tipos de ataques.

2.3 Técnicas de Detecção de Intrusão

A detecção de intrusão é essencialmente um processo de classificação, onde se tenta identificar padrões de ataques ou anomalias dentro de uma grande quantidade de dados de entrada.

Técnicas de classificação de padrões existentes em outros domínios de pesquisas, como estatísticas, redes neurais, mineração de dados, e até mesmo imunologia têm sido amplamente empregadas nesta área.

A “regra de associação” que foi originalmente proposta por [Agrawal e Srikant 1994] para encontrar conjuntos de itens mais freqüentes em dados de transações de usuários tem sido aplicada para descobrir automaticamente regras de detecção de ataques no JAM [Stolfo e Mok 1999] e em [Kantarcioglu e Clifton 2004].

O NIDES [Valdes e Anderson 1995] utiliza métricas estatísticas e sistemas especialistas para detectar anomalias e uso indevido.

Redes neurais e *VSM* (*Support Vector Machines*), que são normalmente utilizadas em disciplinas de aprendizagem de máquinas, têm sido utilizadas por pesquisadores na detecção de ataques [Bonifacio et al. 1998], [Sung e Mukkamala 2003].

No restante desta Seção, serão apresentadas diversas técnicas de detecção de intrusão encontradas na literatura e em produtos comerciais.

2.3.1 Detecção Estatística

Os algoritmos estatísticos são amplamente utilizados por vários sistemas de detecção de intrusão para extrair características estatísticas descrevendo padrões de atividade em dados de entrada, para definir as métricas esperadas de situações normais e de ataques e para comparar as características observadas com os padrões esperados.

O INBOUND [Tjaden, Welch et al. 2000] detecta anomalias de tráfego através da análise de desvios dos padrões de tráfego das médias normais dos usuários.

A aplicação de *HMMs (Hidden Markov Models)* para processos UNIX foi inicialmente estudada em [Gao et al.].

O NIDES [Valdes e Anderson 1995] representa o comportamento de usuários e de sistemas através de um conjunto de variáveis estatísticas e detecta o desvio entre as atividades observadas e as atividades padrão.

Em [Cabrera et al. 2000], as estatísticas de Kolmogorov-Smirnov são utilizadas para modelar e detectar ataques de negação de serviços e sondagens, assim como em [Cordeiro et al. 2005].

2.3.2 Redes Neurais

Um método eficiente para classificar padrões de modo adaptativo são as redes neurais. Em [Ghosh, Wanken e Charron 1998] e [Bonifacio et al.1998], redes neurais do tipo *BP* (*backpropagation*) foram utilizadas para detectar atividades anômalas dos usuários.

Mapas Auto-Organizantes (*SOMs – Self Organization Maps*) foram aplicados para detectar ataques em [Lichodzijewski et al. 2002].

A importância de características relevantes na detecção de intrusão utilizando *SVMs* (*Support Vector Machines*) foi testada em [Sung e Mukkamala 2003].

Se adequadamente treinados, os classificadores de redes neurais podem aprender a função entre dados de entrada e amostras e classificar dados de entrada não previstos, baseado-se no conhecimento adquirido. Entretanto, o conhecimento que os classificadores de redes neurais adquirem com o treinamento pode ser difícil de ser interpretado. O intensivo uso de computação com grandes ciclos de treinamento também pode dificultar a aplicação de redes neurais em sistemas de tempo real.

2.3.3 Reconhecimento de Padrões

O reconhecimento de padrões é a técnica mais comum utilizada na detecção do uso inadequado de sistemas, através da pesquisa de assinaturas de ataques específicos dentro de arquivos ou pacotes.

Exemplos de sistemas comerciais incluem o *RealSecure* da *Internet Security Sistems* [ISS 2006] e o *NFR Security* [NFR 2006].

Exemplos de software de código aberto incluem o dispositivo de detecção do *SNORT* [Beale e Foster 2003] e o *BRO* [Paxson 1999].

A vantagem da técnica de reconhecimento de padrões é que este método é direto e fácil de ser compreendido pelos administradores. Entretanto, o desenvolvimento e os testes de sistemas de reconhecimento de padrões para assinaturas de ataque consomem muito trabalho.

É muito difícil para estes sistemas permanecerem atualizados com a evolução das técnicas de ataques. E mais importante ainda: estes sistemas não podem detectar novos ataques, que não foram previstos anteriormente.

2.3.4 Técnicas de Mineração de Dados

Pelas dificuldades em desenvolver manualmente assinaturas de ataques e padrões normais, técnicas de mineração de dados (*Data Mining*) têm sido utilizadas por vários pesquisadores para automatizar a descoberta destes padrões.

O processo de mineração de dados (também conhecido como *KDD - Knowledge Discovery in Databases*) é definido como “A extração não trivial de informações,

potencialmente úteis, implícitas e previamente desconhecidas, a partir de conjuntos de dados" [Frawley et al. 1992].

Os algoritmos mais utilizados de mineração de dados em sistemas de detecção de intrusão incluem regras de associação [Stolfo e Mok 1999], árvores de decisão [Ye et al. 2000], [Gunneti e Ruffo 1999] e clusterização [Portnoy et al. 2001], [Shah et al. 2003].

Estas técnicas têm sido amplamente utilizadas, uma vez que elas podem descobrir automaticamente modelos detalhados de situações normais e de ataque, que podem ser facilmente compreendidos por seres humanos. Mas estes sistemas de mineração de dados tendem a gerar um número muito grande de modelos, especialmente para dados de entrada muito grandes. A intervenção humana e muitos cuidados adicionais são necessários para reduzir e refinar os modelos extraídos.

2.3.5 Sistemas Imunológicos de Computador

Os sistemas imunológicos naturais protegem os animais de patogenias perigosas, que incluem bactérias, vírus, parasitas e toxinas. O papel dos sistemas de segurança de computadores é análogo aos sistemas imunológicos naturais.

Inspirado pelos princípios da Imunologia, [Forrest et al. 2003] aplicou as idéias da Imunologia na elaboração de sistemas artificiais de imunologia para computadores. Nesta técnica, o problema de proteger sistemas de computadores contra ataques foi visualizado como uma instância de um problema mais geral de auto-reconhecimento (usuários legítimos, programas não infectados) de outros (usuários não autorizados, vírus e outros códigos maliciosos).

Outro sistema de detecção inspirado na imunologia utilizando-se de mecanismos de seleção na detecção de ataques foi apresentado por [Dasgupta e Gonzalez 2002].

Embora teoricamente estes sistemas sejam interessantes, a real eficácia desta técnica imunológica ainda aguarda ser comprovada

2.4 Cenários de Ataques e Correlação de Alertas

Um cenário de ataque é uma seqüência de ataques que um atacante realiza para atingir determinados propósitos maliciosos. Enquanto alguns ataques isolados ocorrem sem aviso,

muitas intrusões em redes se dão através de estágios bem definidos para executar uma varredura inicial e comprometer as vítimas.

Um cenário de ataque resultante de um ataque distribuído de negação de serviços (*DDoS – Distributed Denied-of-Service*), Figura 4, contém as seguintes fases distintas:

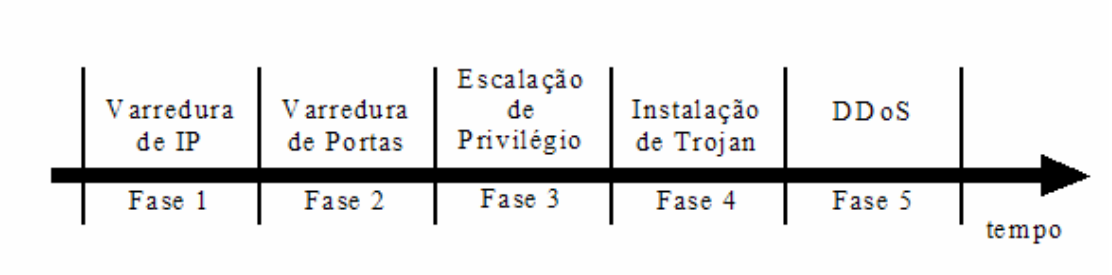

Figura 4 – Um cenário de ataque.

1. O atacante mapeia a topologia de rede da vítima através de uma varredura de IP;

2. Ataques de varredura de portas e sondagens são disparados para identificar as portas abertas, os serviços fornecidos, os sistemas operacionais e os possíveis pontos de penetração e métodos a serem empregados;

3. O atacante entra nos equipamentos das vítimas e obtém os privilégios necessários através da exploração das fraquezas previamente identificadas;

4. Programas *Trojan Horse* e *Back-Door* são instalados nos sistemas comprometidos;

5. Estes sistemas comprometidos são então utilizados como agentes para disparar um ataque de negação de serviços distribuído (*DDoS – Distributed Denied-of-Service*) para outra rede.

O desdobramento gradual característico deste cenário de ataque permite um amplo período de tempo para que um sistema de detecção de intrusão detecte e bloqueie o ataque em seus estágios iniciais de modo que danos mais severos possam ser evitados.

A detecção destas diferentes ameaças e a descoberta dos cenários de ataques o mais rapidamente possível permite fornecer muitas informações importantes aos administradores de sistemas, para que contramedidas de segurança possam ser tomadas de modo quase que imediato.

Sistemas de detecção de intrusão tradicionais são treinados para detectar todas as atividades que considerarem suspeitas. Entretanto, a utilidade destes alertas gerados é geralmente limitada pelos seguintes fatores:

- **Os alertas positivos são em geral granulares, de baixo nível:** Cada alerta corresponde a uma pista de ataque que o sistema de detecção de intrusão detectou em um pacote ou em uma sessão. A chegada dos alertas pode ser explosiva e sobreposta. Uma visão situacional de alto nível descrevendo os cenários completos de ataques é ausente.

- **A "eficácia" dos sistemas de detecção de intrusão atuais é baixa:** A "eficácia" de um IDS neste estudo é definida como a probabilidade que uma detecção de alerta positivo seja realmente verdadeira. Mas mesmo para um IDS com uma taxa de falso-positivo muito baixa, ainda é mínima a probabilidade de que um alarme indique uma intrusão real, pelo fato da quantidade de dados sobre atividades normais ser muito maior que a quantidade de informações sobre intrusões [Axelsson 1999] e [Bertino et al. 2005].

Esta questão de quantidade de dados de atividades normais quando comparada com a quantidade de dados de atividades de intrusão é um dos mais sérios problemas dos sistemas IDS atuais. Cada alerta necessita de tempo e discernimento humano para ser investigado.

Um grande número de falso-positivos pode distrair a atenção dos administradores de maneira que positivos críticos possam ser acidentalmente ignorados. E a quantidade de falso-positivo pode se tornar muito alta a ponto de fazer com que os administradores simplesmente ignorem todos os alertas.

Pelos motivos citados anteriormente, a correlação de alertas (também conhecida como "*fusão de dados*" em algumas literaturas) tem se tornado um ponto atraente de interesse de pesquisas, como uma solução para os problemas acima descritos.

A correlação de alertas relaciona os alertas de diferentes ataques e sensores de detecção de intrusão dentro dos cenários de ataques, que fornecem uma descrição de nível mais estruturado sobre os planos de intrusão. Estes cenários podem auxiliar no entendimento das intenções de intrusão e no tratamento das ameaças de segurança. E os falso-positivos

podem ser isolados e filtrados, baseando-se nas suas diferenças no espaço e tempo, além de outros aspectos.

Um número muito grande de projetos de pesquisa tem sido realizado no campo da correlação de alertas. O *GrIDS* [Staniford et al. 1996] utiliza conjuntos de regras pré-definidas para combinar alertas e dados de rede em uma estrutura gráfica para descobrir ataques coordenados em larga escala.

O *EMERALD* [Valdes e Skinner 2001] relaciona os alertas usando um método probabilístico através da comparação de similaridades entre os alertas.

Algoritmos para estimar a probabilidade de que um alerta pertença a um determinado cenário, relacionando o alerta a um cenário mais provável, são propostos por [Dain Cunningham 2001] e aplicados em [Wang Daniels 2005].

Técnicas de consolidar pacotes anômalos em cenários de varredura de portas são apresentadas em [Stainford et al. 2002].

2.5 Avaliação da Detecção de Intrusão

A detecção de intrusão é um processo de classificação, que envolve o ajuste de modelos a dados, e a análise destes modelos.

Geralmente o desempenho de um sistema de reconhecimento de padrões pode ser quantificado em duas medidas: a TVP - Taxa de Verdadeiro-Positivo (Probabilidade positiva/ataque) e a TFP – Taxa de Falso-Positivo (Probabilidade positiva/normal).

Como definido na Seção 1.3.2, geralmente não é possível obter uma TVP de valor 1 e uma TFP de valor 0. A prática comum nos algoritmos de avaliação de reconhecimento de padrões é escolher um padrão com o melhor TVP dentro das considerações de um TFP aceitável. Mas a avaliação de sistemas de detecção de intrusão, utilizando-se destes critérios, é problemática pelas seguintes motivos:

1. **Não existem padrões de definição de eventos de segurança monitorados:** Um sistema de detecção de intrusão típico procura por uma série de eventos e tenta identificar aqueles que representam uma intrusão. Os dados de entrada podem ser registros gravados de um ou mais serviços monitorados de um computador,

pacotes em uma rede, ou outros descritores de atividades dentro de um domínio monitorado. Um evento de segurança (a unidade de análise) pode ser definido como um registro gravado, um pacote de rede, ou uma sessão TCP. Esta questão de diversidade de aceitação de definições de eventos pode conduzir a grandes diferenças de valores TVP e TFP, tornando sem significado a comparação de sistemas com diferentes definições de eventos.

2. **O desempenho da detecção de intrusão é sensível ao tráfego e aos ataques:** Os sistemas de detecção de intrusão são treinados e testados através da coleta de dados de ambientes de testes contendo um número limitado de circunstâncias de ataques. Um sistema de detecção de intrusão com bom desempenho de treinamento e teste pode atingir uma alta taxa de falso-positivo (TFP) em outro ambiente com configurações completamente diferentes.

3. **Conjuntos de dados independentes e não tendenciosos são difíceis de se obter:** Os dois primeiros problemas descritos acima podem ser atenuados através de testes de sistemas de detecção de intrusão com um conjunto comum de dados. Mas, infelizmente, conjuntos de dados independentes são difíceis de se obter, pelos seguintes motivos:

   - Dados de rede são difíceis de serem identificados adequadamente (Podem existir ataques ocultos com padrões ainda não identificados).

   - Modelos de ambiente e modelos de ataques não são estacionários, e não podem ser claramente definidos, fazendo com que a utilidade de qualquer dado estático tenha um tempo de vida relativamente curto.

2.5.1 Projetos de Avaliação de Detecção de Intrusão do DARPA/MIT-LL

Em um esforço de avaliar de maneira confiável as capacidades de sistemas de detecção de intrusão existentes, o *MIT-LL (MIT Lincoln Labs)* foi designado pelo DARPA para elaborar uma rede de simulação em 1998.

O tráfego de rede foi simulado de acordo com as estatísticas de tráfego observadas em uma base da Força Aérea. Uma mistura de perfis diferentes de usuários e ataques distintos foi simulada na rede.

Até o momento, três conjuntos de dados (conhecidos como DARPA98, DARPA99 e DARPA2000) foram liberados e podem ser obtidos a partir de seus sites na Internet [IDEVAL 2005]. As definições de eventos de segurança são levemente diferentes entre estes três conjuntos de dados.

No DARPA98, um evento é definido como uma sessão, que pode ser TCP, UDP ou ICMP. Uma sessão é caracterizada por um horário de início, duração, serviço, fonte e destino (endereço IP e porta). Entretanto, utilizar a sessão como uma unidade de analise é insatisfatório uma vez que isto não fornece resultados precisos se:

- Um ataque simples necessita mais do que um serviço para ser utilizado ou envolve múltiplas sessões do mesmo serviço para ser completado, ou
- Uma determinação de alarme falso é baseada nos dados de mais de uma sessão.

No DARPA98 e no DARPA2000, os eventos são definidos como instâncias de ataque, que podem incluir múltiplas sessões. Uma instância de ataque é caracterizada pela data do ataque, o tempo de ataque e o endereço IP da vítima.

2.6 Estrutura de um EWIDS

Um *EWIDS* (*Early Warning Intrusion Detection System)* é projetado para detectar três tipos de ataques:

- Ataques de reconhecimento (*P&S*),
- Ataques de negação de serviço (*DoS*),
- Ataques de escalação de privilégios (*R2L ou U2R*).

Ele também correlaciona os alertas de reconhecimento de ataques, dentro de cenários de ataques, descobrindo os ataques coordenados distribuídos e alertando os administradores o mais rápido possível quando um ataque está sendo iniciado. Um sistema completo EWIDS geralmente é composto dos seguintes cinco subsistemas (Figura 5):

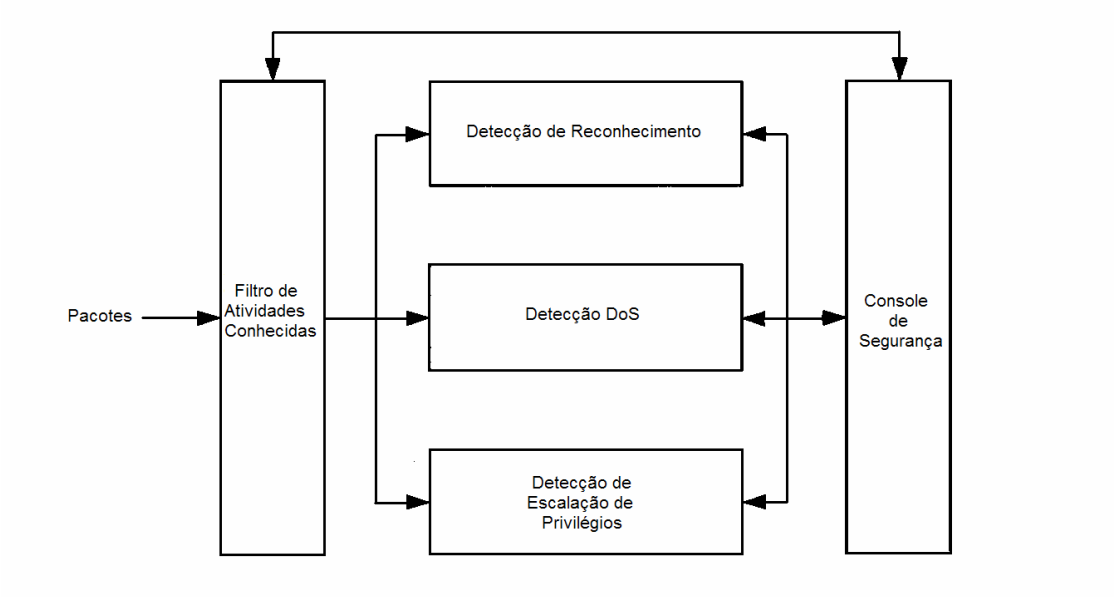

Figura 5 – Diagrama de sistema de um EWIDS.

- **Filtro de Atividades Conhecidas:** Atua como filtro de pacotes para o EWIDS. Ele permite os pacotes conhecidos e bloqueia os pacotes de ataques conhecidos. Um diagrama detalhado de um Filtro de Atividades Conhecidas é apresentado na Seção 2.6.1

- **Sistema de Detecção *DoS*:** Procura identificar este tipo de ataque em seu estágio inicial, para que o ataque possa ser bloqueado em sua entrada na rede, evitando-se a significativa degradação de serviços. O método de detecção *DoS* utilizado neste estudo é descrito no Capítulo 3.

- **Sistema de Detecção de Reconhecimento:** Detecta ataques conhecidos. Apesar de ataques de reconhecimento não causarem danos materiais nos sistemas vítimas, eles fornecem informações importantes sobre a eminência de ataques mais maliciosos. Através da detecção de ataques de reconhecimento, um sistema de detecção pode fornecer a proteção para evitar ataques com efeitos mais devastadores em seus primeiros estágios.

- **Sistema de Detecção de Escalação de Privilégios**: Evita que os atacantes ganhem privilégios não autorizados através da exploração das fraquezas do sistema. Este subsistema é uma parte importante de um sistema EWIDS, mas seu estudo está fora do escopo deste trabalho.

- **Console de Segurança:** É a interface entre os administradores e um sistema EWIDS. Ele permite visualizar os alertas de ataques detectados pelo sistema e fornece aos administradores a interface de configuração para administrar todo o sistema.

2.6.1 Filtro de Atividades Conhecidas

O módulo do Filtro de Atividades Conhecidas, exibido na Figura 6, processa os pacotes capturados na rede e filtra os pacotes em que os endereços de fonte e destino estão relacionados em listas negras ou brancas. Uma lista negra mantém os endereços IP de atacantes conhecidos. Se o Filtro de Atividades Conhecidas detecta pacotes destes atacantes, ele irá imediatamente informar ao Console de Segurança e bloquear o pacote. A lista branca mantém os endereços IP de usuários normais conhecidos. Com a filtragem de pacotes de status conhecidos, uma boa parte do poder de computação pode ser economizada para se concentrar na analise de tráfego desconhecido.

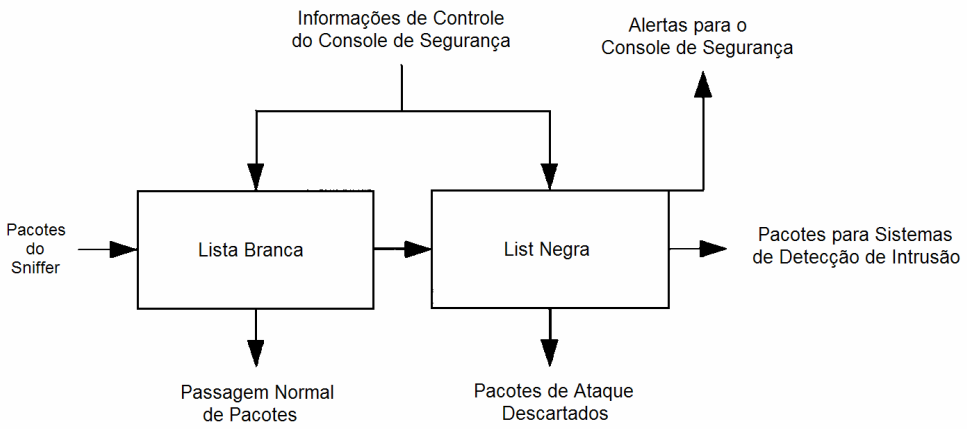

Figura 6 – Filtro de Atividades Conhecidas.

3 Análise Estatística de Anomalias

Este estudo está baseado em análise estatística de anomalias para a detecção de intrusão, através da representação de parâmetros estatísticos em formato de PDF, comparando os parâmetros observados com os modelos de referência utilizando-se de métricas de similaridade PDF, classificando as distâncias de similaridades medidas com classificadores de redes neurais.

A Seção 3.1 revisa a literatura existente na detecção de ataques *DoS*. A Seção 3.2 introduz a estrutura das técnicas empregadas. A Seção 3.3 descreve as definições de eventos e as características estatísticas monitoradas. A Seção 3.4 apresenta os modelos estatísticos utilizados. Os resultados de detecção são exibidos na Seção 3.5.

3.1 Revisão de Literatura sobre Detecção de Ataques *DoS*

Os ataques *DoS* se apresentam como ameaças sérias na infra-estrutura de computadores de organizações comerciais e governamentais. Qualquer interrupção ou queda nos níveis de serviços fornecidos por uma organização pode causar grandes perdas financeiras e comprometer a reputação pública de uma organização. A recente tendência de união entre vírus de computadores e ataques *DoS*, como por exemplo, os ataques *DoS* nos sites de Internet da Microsoft e da SCO disparados por máquinas infectadas pelo vírus *Mydoom* [NAI 2006] tornou esta tendência ainda mais intensa.

Existem muitas variedades de ataques *DoS*. Alguns ataques *DoS* (por exemplo: ataques *mailbomb*, *neptune* e *smurf*) se aproveitam de características legítimas. Outros ataques (por exemplo: *teardrop* e *ping of death*) criam pacotes mal formados que têm como alvo as pilhas de protocolo TCP/IP das máquinas. Outros tipos de ataques (*back*, *syslogd*) exploram os *bugs* de serviços específicos de rede e aplicações.

Muitos sistemas de detecção de intrusão comerciais e de código aberto, como o *RealSecure* [ISS 2006], *SNORT* [SNORT 2006] e o *Bro* [Paxson 1999], utilizam assinaturas e contadores de eventos relevantes para detectar ataques *DoS*. Estas técnicas são úteis na detecção de ataques conhecidos *DoS* que utilizam pacotes mal formados e que exploram *bugs* de aplicações especificas, mas eles têm dificuldade em determinar novos ou velhos ataques *DoS* que se aproveitam de características legítimas.

Muitas atividades de pesquisa têm sido realizadas para detectar ataques *DoS*. O *NIDES* [Valdes e Anderson 1995] desenvolveu algoritmos estatísticos sofisticados para medir as distribuições utilizando testes de χ^2 para medir a similaridade entre dois perfis de utilização.

Sessões foram representadas utilizando-se 41 características quantitativas e qualitativas e aplicando-se regras de associação para descobrir automaticamente padrões de ataques em [Stolfo e Mok 1999]. Uma parte do conjunto de dados DARPA98 foi processada utilizando estas 41 características e os arquivos de dados resultantes foram publicados como dados do concurso *KDD CUP 1999* [KDD 1999]. A importância destas 41 características foi avaliada por [Sung e Mukkamala 2003] utilizando classificadores de redes neurais e *SVMs* (*Support Vector Machines*).

A eficácia das classificações dos dados do KDD CUP 1999 foi testada por [Giacinto e Roli 2002] utilizando classificadores neurais.

Técnicas baseadas em Imunologia foram utilizadas por [Dasgupta e Gonzalez 2002] para detectar os ataques de rede do conjunto de dados do DARPA'99.

3.2 Arquitetura de Sistema

O método de análise proposto neste estudo está distribuído de modo hierárquico em diversas camadas, cada uma delas com *ADIs - Agentes de Detecção de Intrusão*. Os *ADI*s são componentes de detecção de intrusão que monitoram as atividades de um computador ou de uma rede.

O diagrama de um Agente de Detecção de Intrusão está ilustrado na Figura 7. Ele é composto pelos seguintes componentes:

- Sondagem.
- Pré-Processador de Eventos.
- Processador Estatístico.
- Classificador de Redes Neurais.
- Pós-Processador.

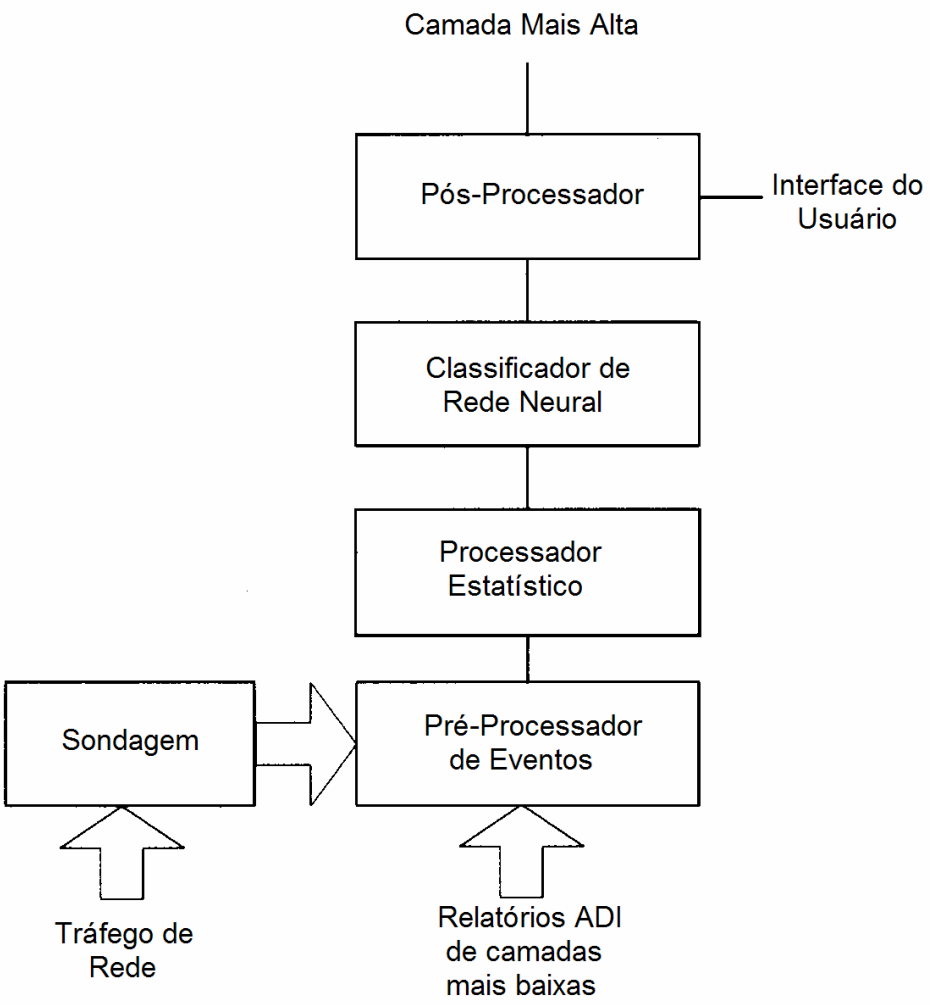

Figura 7 – Diagrama do Agente de Detecção de Intrusão (ADI).

As funcionalidades destes componentes são descritas a seguir:

- **Sondagem:** Coleta o tráfego de rede de um computador ou de uma rede, abstrai o tráfego em um conjunto de variáveis estatísticas para refletir o status da rede, e periodicamente gera relatórios para o Pré-Processador de Eventos (Figura 8).

- **Pré-Processador de Eventos:** Recebe relatórios da Sondagem e dos ADIs das camadas mais baixas e converte a informação em um formato requerido pelo modelo estatístico (Figura 8).

- **Processador Estatístico:** Mantém os modelos de referência das atividades típicas de rede, compara os relatórios do Pré-Processador de Eventos para os modelos de

referência, e forma um vetor de estímulo para alimentar os Classificadores de Redes Neurais.

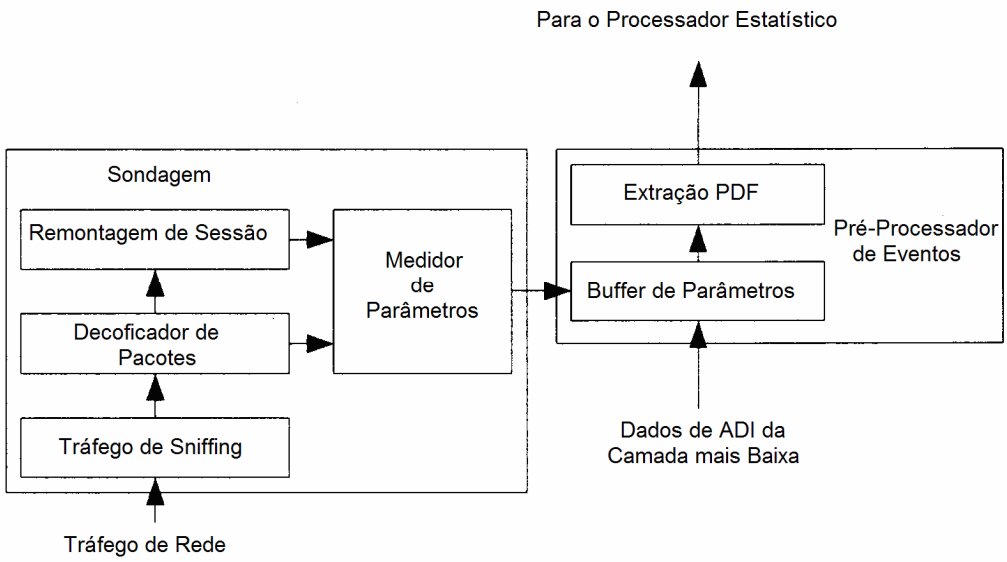

Figura 8 – A sondagem e o pré-processador de eventos.

- **Classificadores de Redes Neurais:** Analisa o vetor de estímulo do modelo estatístico para decidir se um tráfego de rede é normal ou não.

- **Pós-Processador:** Gera relatórios para os agentes nas camadas mais altas.

3.3 Eventos Monitorados e Descrição das Características

Este estudo realiza uma análise de pacotes em janelas de tempo. O evento (unidade de análise) é definido como "o padrão estatístico agregado de todo o tráfego de rede, incluindo pacotes de entrada e saída, observados dentro de uma janela de tempo".

Ao final de cada janela de tempo, todas as características estatísticas monitoradas, que representam as estatísticas de tráfego da janela de tempo analisada, são extraídas pelo Pré-Processador de Eventos e encaminhadas para os estágios seguintes para detectar possíveis ataques.

O Classificador de Redes Neurais irá emitir um valor contínuo descrevendo a normalidade ou não do tráfego de rede analisado.

É possível monitorar muitas estatísticas de tráfego para descrever com precisão o status da rede (uma descrição detalhada de todas as características estatísticas é apresentada no Apêndice I). Entretanto, monitorar todas estas características é computacionalmente dispendioso e um subconjunto destas características já é suficiente para detectar ataques *DoS*.

3.4 Modelo Estatístico PDF

Os modelos estatísticos têm sido muito utilizados em sistemas de detecção de intrusão. Muitos destes sistemas simplesmente medem a média e o desvio padrão de parâmetros estatísticos e detectam se um limite específico pré-determinado foi ultrapassado, como o *INBOUND* [Tjaden, Welch et al. 2000].

O *NIDES* [Valdes e Anderson 1995] do SRI desenvolveu um algoritmo mais sofisticado utilizando um teste de χ^2 para medir a similaridade entre perfis de utilização.

Em [Cabrera et al. 2000], estatísticas Kolmogorov-Smirnov (KS) foram utilizadas para modelar e detectar ataques de negação de serviços (DoS) e sondagem (P&S). O modelo estatístico utilizado neste estudo utiliza um algoritmo KS modificado.

Algumas informações básicas sobre χ^2 e estatísticas KS serão apresentadas neste Capítulo. Neste estudo, as atividades do usuário são representadas pro um número de funções densidade de probabilidades.

Seja S o espaço de amostragem de uma variável randômica e os eventos $E_1, E_2,...E_K$ partições mutuamente exclusivas de S.

Assume-se p_i como a probabilidade esperada da ocorrência do evento E_i, e p_i' a freqüência da ocorrência de E_i durante um dado intervalo de tempo.

Seja N o número total de ocorrências. O teste de χ^2 (Equação 3.1) é utilizado como o componente estatístico para determinar a similaridade entre as distribuições reais e esperadas.

$$Q = N \times \sum_{i=1}^{K} \frac{(p_i' - p_i)^2}{p_i} \tag{3.1}$$

Se N é grande e os eventos $E_1, E_2,...E_K$ são independentes, Q segue aproximadamente uma distribuição χ^2 com $K - 1$ graus de liberdade. Entretanto, para uma aplicação em tempo real, as duas afirmações acima não podem ser garantidas, e Q pode não seguir uma distribuição χ^2.

O teste de Kolmogorov-Smirnov, (Equação 3.2) é aplicável para medir a similaridade quando a distribuição não é conhecida.

Assume-se que $F(x)$ é a distribuição de probabilidade acumulada esperada (*CDF – Cumulative Distribution Function*); $F'(x)$ é a CDF medida; Fi e Fi' são as CDFs esperadas e medidas no evento Ei.

$$D_N = \max_{-\infty}^{\infty} \left| F(x) - F'(x) \right| = \max_{i=0}^{K} \left| F_i - F_i' \right| \tag{3.2}$$

Utilizando-se o teste de KS, assume-se que $F(x)$ é contínua e a distribuição de D_N não depende da distribuição de $F(x)$, mas apenas do número de amostras N.

Pelo fato das redes neurais serem usadas como classificadores para identificar possíveis intrusões, não é preciso conhecer a real distribuição de D_N.

Entretanto, uma vez que o tráfego de rede não é estacionário e ataques de rede podem ter durações de tempo diferentes, variando de alguns segundos a diversas horas ou mais, um algoritmo robusto e eficiente é necessário para monitora o tráfego de rede com janelas de tempo diferentes.

Baseando-se nas observações anteriores, adotou-se um modelo estatístico de janelas em camadas (Figura 9), com cada camada-janela correspondendo a uma fatia de tempo monitorada, de tamanho incrementado.

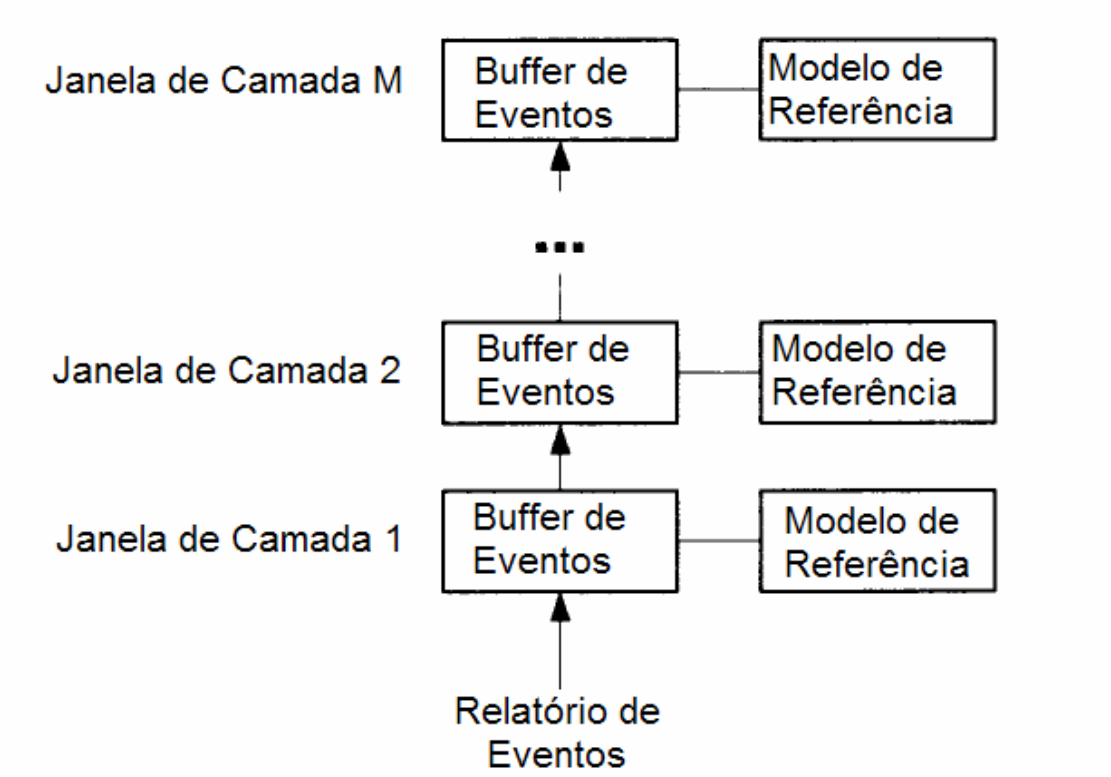

Figura 9 – Modelo Estatístico.

Os novos eventos que surgem são armazenados inicialmente no buffer de eventos da camada 1.

Os eventos armazenados são comparados com o modelo de referência dessa camada e os resultados são então alimentados no classificador de rede neural para decidir o status da rede durante esta janela de tempo.

O buffer de eventos é limpo quando se torna cheio, e as médias dos eventos armazenados são então encaminhadas para o buffer de eventos da camada 2.

Este processo se repete recursivamente até que o nível mais alto é atingido onde os eventos são descartados após o processamento. O algoritmo de similaridade utilizado neste estudo é dado pela equação 3.3

$$Q = f(N) \times \left[\max_{i=1}^{K} (|p_i' - p_i|) + \sum_{i=1}^{K} |p_i' - p_i| \right]$$

(3.3)

O lado direito da equação é a combinação do teste de *KS* e a diferença de área entre as distribuições.

Esta equação é conhecida como teste *AKS* [Cabrera et al. 2000]. A função *f(N)* é a função do número de amostras *N*.

Uma atualização do algoritmo de referência também é projetada para atualizar o modelo de referência baseada no novo tráfego em tempo real observado.

Seja M_{old} o modelo de referência antes da atualização, M_{new} o modelo de referência após a atualização e *K* a atividade de usuário observada dentro de uma janela de tempo.

A fórmula para atualizar o modelo de referência é dada pela Equação 3.4.

$$\mathrm{M}_{new} = s \times \alpha \times \mathrm{K} + (1 - s \times \alpha) \times \mathrm{M}_{old}$$

(3.4)

Onde α é uma taxa de adaptação pré-definida e *s* é o valor gerado pela saída do classificador neural.

Assumindo-se que a saída do classificador de rede neural é uma variável contínua *c* entre -1 e 1, onde -1 significa intrusão com absoluta certeza e 1 significa nenhuma intrusão com completa confiança. Entre eles, os valores de níveis proporcionais de certeza.

A função para calcular *s* é:

$$s = \begin{cases} c \text{ se } c \geq 0 \\ 0 \text{ se } c < 0 \end{cases}$$

Através das equações acima, é possível garantir que o modelo de referência irá ser ativamente atualizado para o tráfego típico enquanto se mantém inalterado quando ocorrem ataques.

Os eventos de ataques podem ser separados e armazenados, como *scripts* de ataques, para futuras investigações.

3.5 Resultados de Detecção.

Dois conjuntos de dados diferentes foram utilizados para avaliar o desempenho deste estudo de detecção de intrusão.

O primeiro conjunto de dados foi gerado através de simulações com o OPNET.

O segundo conjunto de dados foi obtido através do projeto de avaliação de detecção de intrusão DARPA98.

3.5.1 Dados de Simulação OPNET

Uma rede virtual utilizando ferramentas de simulação foi utilizada para gerar cenários de ataques. O ambiente experimental que foi construído utilizando-se o OPNET, uma poderosa ferramenta de simulação, é exibido na Figura 3.4. O ambiente de teste é uma LAN 10-BaseX composto de onze estações e um servidor.

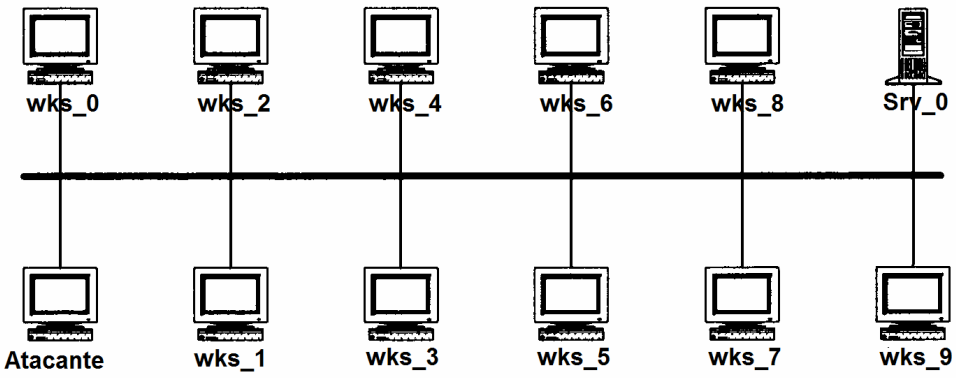

Figura 10 – Ambiente de simulação OPNET.

Na rede de simulação, três tipos de tráfego simulado foram utilizados: HTTP, FTP e SMTP (E-mail).

Para gerar suficiente tráfego UDP de fundo, o UDP foi configurado como o protocolo de transporte dos serviços SMTP. Embora esta configuração não seja completamente realística, ela não compromete as conclusões obtidas neste estudo.

Os ataques *flooding* de UDP foram simulados dentro do ambiente. Múltiplos cenários de simulação, com diferentes tráfegos de fundo e tráfegos de ataque foram aplicados. A Tabela 2 lista as especificações dos cenários simulados.

Tabela 2 – Especificações de Carga de Tráfego da Simulação OPNET.

TRÁFEGO DE FUNDO	TRÁFEGO DE ATAQUE
600 Kbps	10 Kbps, 20 Kbps, 30 Kbps, 40 Kbps, 50 Kbps, 70 Kbps, 100 Kbps, 200 Kbps
2 Mbps	10 Kbps, 50 Kbps, 150 Kbps, 200 Kbps

Para cada cenário de simulação, dez mil registros de tráfego de rede foram coletados. Estes dados foram divididos em dois conjuntos separados: um conjunto de 6.000 dados para treinamento e outro de 4.000 dados para testes.

Os resultados de classificação dos testes são apresentados na Figura 11, a seguir.

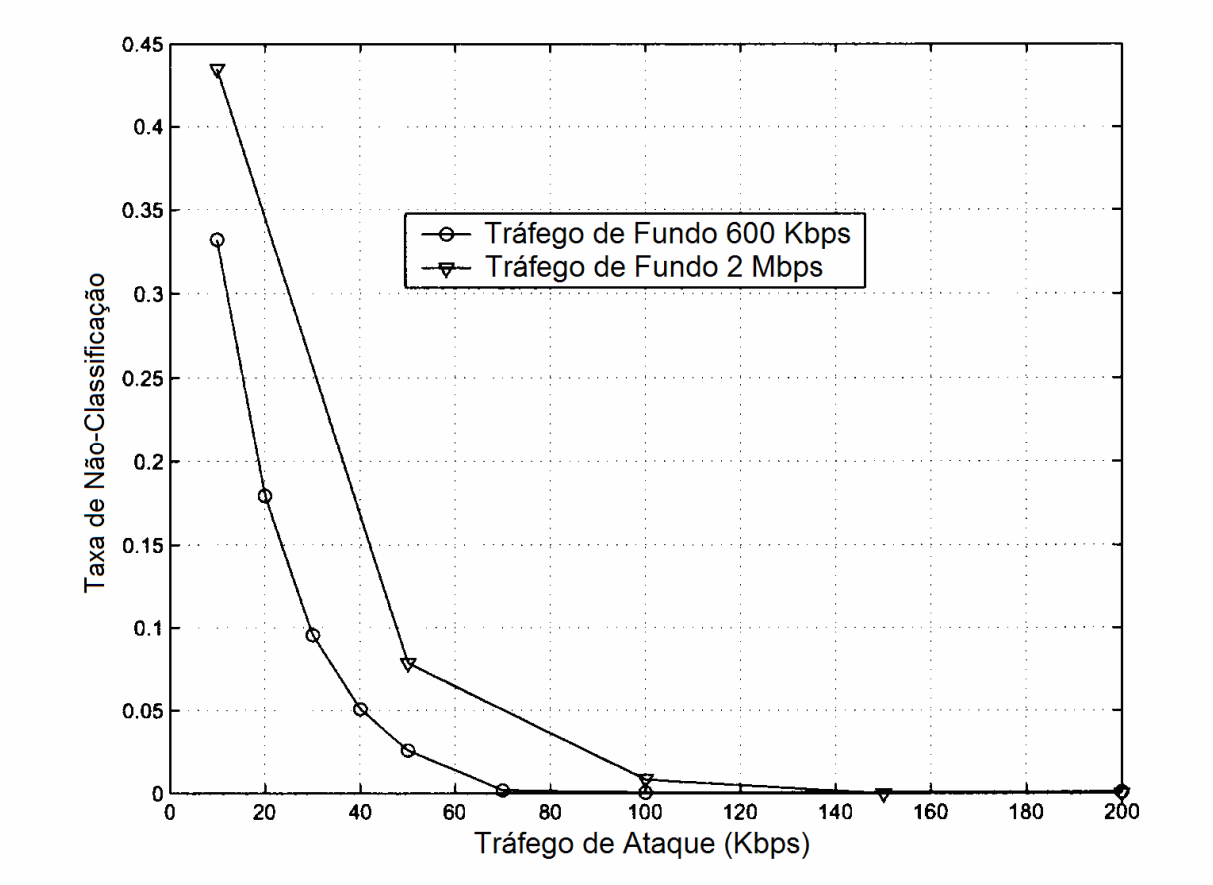

Figura 11 – Resultados de detecção com dados OPNET.

É possível observar que as taxas de não-classificação decrescem à medida que o nível de ataque aumenta. Isto ocorre porque os padrões de tráfego de uma grande quantidade de ataques atingem diferenças muito maiores em relação ao modelo de referência que as diferenças criadas por um baixo volume de ataques.

Também pode ser observado que para um determinado nível de ataque, o desempenho do tráfego de fundo de 600 Kbps é consistentemente melhor que o de 2 Mbps. Uma explicação plausível é que os invasores podem encobrir mais facilmente seus padrões de comportamento em ambientes de alto tráfego de fundo.

De uma maneira geral, os resultados indicam que o processo proposto pode detectar efetivamente ataques de *flooding* UDP com intensidade de tráfego, na faixa de cinco a dez por cento da intensidade de tráfego de fundo.

3.5.2 Dados de Avaliação de Detecção de Intrusão DARPA98

Este conjunto de dados foi gerado pelo MIT Lincoln Labs em 1998 através de um projeto de avaliação de detecção *offline* patrocinado pelo DARPA [IDEVAL 2005]. Estes dados foram coletados em um ambiente de rede controlado e isolado (Figura 12).

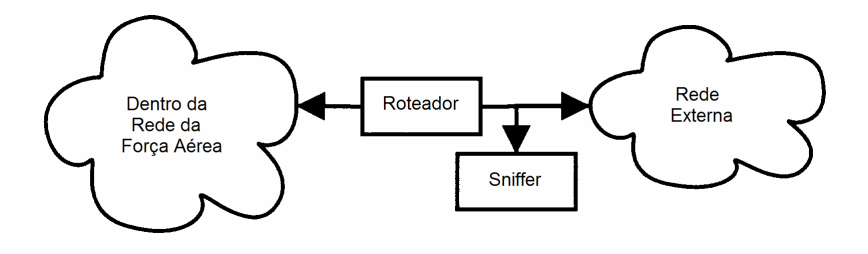

Figura 12 - A rede DARPA IDEVAL.

O ambiente de rede controlado e isolado consistia de dois segmentos: a rede interna simulava a rede de uma base da Força Aérea dos Estados Unidos, e a rede externa simulava a Internet. Estes dois segmentos eram conectados através de um roteador. Um *sniffer* foi utilizado para capturar os pacotes que passavam através do roteador.

O conjunto de dados DARPA98 possui sete semanas de dados de treinamento e duas semanas de dados de testes que contém os seguintes tipos de ataques:

- **Ataques de Sondagem e Reconhecimento *P&S (Probe & Scan)*:** São usados para obter informações sobre a topologia de rede da vítima, os serviços de rede, e os potenciais pontos de penetração.

- **Ataques de Negação de Serviços *DoS* :** São ataques que têm como objetivo derrubar um servidor ou uma rede de tal modo que os usuários não possam ser capazes de acessar seus serviços.

- **Ataques Remotos para Local R2L (*Remote to Local*):** São ataques com o objetivo de ganhar acesso não autorizado a um servidor local a partir de uma máquina remota.

- **Ataques Usuário para Raiz *U2R* (*User to Root*):** São utilizados por um usuário local sem privilégios para ganhar acesso não autorizado a privilégios de super-usuários locais.

Apenas os ataques *DoS* foram considerados neste estudo. O restante dos pacotes de ataques foi filtrado. Os ataques *DoS* considerados incluíram:

- ***Neptune:*** Um ataque do tipo *SYN flooding denial-of-service* em uma ou mais portas TCP para sobrecarregar o buffer de conexão TCP através da criação de um grande número de conexões TCP de portas "meio-abertas".

- ***Pod* (ou *Ping of Death*):** É um ataque de negação de serviços para derrubar servidores vítimas através do envio de grande quantidade de pacotes IP sobrecarregados.

- ***Smurf:*** É um ataque DoS para "inundar" o servidor da vítima através da criação de um fluxo grande e contínuo de respostas ICMP do tipo "ECHO".

- ***Teardrop:*** É um ataque DoS que explora as fraquezas de programas de desfragmentação IP de muitos sistemas operacionais mais antigos.

Os resultados da detecção são apresentados na Tabela 3.

Tabela 3 – Resultados Resumidos do Conjunto de Dados DARPA98.

Número Total de Amostras	59.226
Número Total de Ataques	1.074
Número Total de Não-Classificações	32
Número Total de Falso-Positivos	25
Número Total de Falso-Negativos	7
Taxa de Não-Classificação	0,000540
Taxa de Falso-Positivo	0,000422
Taxa de Falso-Negativo	0,00652
"Eficácia"	0,977

No total, 59.226 amostras de dados significativas foram coletadas. Entre elas, 1.074 amostras são ataques enquanto as 58.152 amostras remanescentes são normais. Portanto, a linha de base da Taxa de Não-Classificação para este conjunto de dados DARPA98 é 1.074/59.226 = 0,0181.

O desempenho da detecção deste estudo em diferentes ataques *DoS* é apresentado na Tabela 4, onde pode ser observado que taxas muito baixas de falso-negativo foram obtidas para todos os quatro tipos de ataques *DoS* monitorados.

Tabela 4 – Resultados do Conjunto de Dados DARPA98.

ATAQUE	AMOSTRAS	FALSO-NEGATIVOS	TFN
Neptune	798	6	0,00752
Pod	24	0	0
Smurf	267	1	0,00375
Teardrop	12	0	0

3.5.3 Resultados DARPA de outros *IDSs*

Como os conjuntos de dados DARPA98 são acessíveis ao público e amplamente utilizados em várias literaturas de detecção de intrusão, uma dúvida natural que pode surgir é como o desempenho do método utilizado neste estudo se compara com outros sistemas de detecção de intrusão. Uma comparação completa é difícil porque as definições de eventos entre a metodologia empregada neste estudo e os projetos de avaliação do MIT-LL são diferentes.

Considerou-se como unidade básica de análise uma janela de tempo, e os projetos DARPA do MIT-LL usam sessões ou instâncias de ataque como eventos básicos. Alguns dos resultados existentes utilizando-se os conjuntos de dados DARPA do MIT-LL são apresentados a seguir:

- Os melhores resultados de detecção *DoS* no projeto DARPA98 IDEVAL atingiram "65% de todos os ataques DoS, mas com muitos poucos alarmes falsos" [Lippman et al. 1999]. O projeto DARPA98 IDEVAL utiliza sessões como unidades de eventos.

- Os melhores resultados de detecção *DoS* no projeto DARPA99 IDEVAL atingiram "85% de todos os ataques *DoS*, com 10 alarmes falsos por dia [Lippman et al.2000] O projeto DARPA99 IDEVAL utiliza instâncias de ataque como unidades de eventos.

- [Sung e Mukkamala 2003] atingiu uma taxa de detecção de 99,25% em ataques *DoS* utilizando *SVMs*. A sessão é utilizada como unidade básica em [Sung e Mukkamala 2003].

- [Dasgupta e Gonzalez 2002] detectou 96,2% de ataques selecionados dos tipos *P&S* e *DoS* no conjunto de dados DARPA99. Em [Dasgupta e Gonzalez 2002], uma janela de tempo de 1 minuto foi utilizada como evento básico.

Dos resultados listados, [Dasgupta e Gonzalez 2002] adotou uma definição semelhante à utilizada neste estudo. A taxa de detecção de 99,4% aqui obtida é maior que a reportada em [Dasgupta e Gonzalez 2002].

Os outros resultados mencionados não podem ser diretamente comparados com este estudo. Eles foram apresentados aqui apenas para ilustrar o nível de desempenho das detecções *DoS* atualmente utilizadas.

4 Características Relevantes

4.1 Introdução

O processo de detecção de intrusão envolve múltiplas variedades de classificações baseadas nas características monitoradas. Geralmente os sistemas de detecção de intrusão são projetados para utilizar todas as características que seus projetistas possam imaginar como aplicáveis.

O JAM [Stolfo e Mok 1999] é um protótipo de sistema de detecção de intrusão da Columbia University que monitorou 41 parâmetros diferentes em cada sessão, enquanto este estudo utilizou 45 medidas de tráfego observadas em uma janela de tempo.

Os resultados experimentais nestes sistemas têm comprovado que todas estas características selecionadas podem ser utilizadas para detectar com precisão as atividades de ataque. Surgem três questões sobre estas técnicas de classificação de múltiplas variantes:

- Todas estas características são igualmente úteis?
- Se nem todas estas características são igualmente úteis, quais são as mais importantes?
- É possível remover algumas destas características sem comprometer o desempenho?

A questão da seleção de características de subconjuntos *FSS (Feature Subset Selection)* tem sido estudada em várias áreas, como Aprendizagem de Máquinas [Inza et al.1999], Reconhecimento de Padrões [Martinez 2002], Estatística [Miller 1990] e Mineração de Dados [Liu e Motoda 1998]. Geralmente, pesquisas nestas áreas tratam os problemas de *FSS* através de dois modelos:

1. **Modelo de Filtro** [Kohavi e John 1997]: Posiciona e seleciona as características através de avaliação das propriedades estatísticas e da relevância do subconjunto analisado. Os métodos de filtragem mais utilizados são:
   - **Critério de correlação** [Furey et al. 2000]: Classifica as características baseado nos valores absolutos dos coeficientes de correlação entre as características e o alvo.
   - **Método de classificação teórica** [Quinlan 1993]: Seleciona as características de acordo com informações comuns entre cada variável e o alvo.

- **Critério de Fisher** [Furey et al.2000]: Classifica as características calculando a diferença estatística de uma característica especifica em classes positivas e negativas:

$$F(r) = \left| \frac{\mu_p - \mu_n}{\sigma_p^2 + \sigma_n^2} \right|$$

(4.1)

μ_n e σ_n^2 : média e variância da característica *r* na classe negativa.

μ_p e σ_p^2 : média e variância da característica *r* na classe positiva.

- **Classificação de características simples** [Guyon e Elisseeff 2003]: Seleciona características de acordo com o poder de previsão das características individuais.

2. Modelo de Encapsulamento [Kohavi e John 1997]: Procura um subconjunto de características através da utilização de algoritmos de aprendizagem como uma caixa preta para classificar subconjuntos de características baseando-se em seu desempenho.

[Liu e Motoda 1998] avaliou as estratégias de pesquisa de características no modelo de encapsulamento: pesquisa completa, pesquisa heurística e pesquisa não-determinística.

A estratégia de pesquisa completa avalia todas as possíveis combinações de conjuntos de características. Esta estratégia garante encontrar o conjunto de características ideal, mas rapidamente se torna computacionalmente impraticável quanto o número de características aumenta.

A estratégia de pesquisa heurística evita a necessidade da pesquisa completa, através da utilização de heurísticos. Este método é computacionalmente mais eficiente que a pesquisa completa, mas pode ter como resultado um subconjunto de características abaixo do ideal se "travar" na máxima local.

A estratégia de pesquisa não-determinística tenta escapar da máxima local através da aplicação de aleatoriedade no processo de pesquisa.

O modelo de filtro é mais rápido e pode ser utilizado como um método genérico para a seleção de características, sem depender de um método de aprendizagem especifico. Por outro lado, o modelo de encapsulamento, embora mais lento, é capaz de atingir subconjuntos de características de modo mais preciso.

O objetivo deste Capítulo é identificar características importantes do método empregado neste estudo, e avaliar seu desempenho quando utilizando diferentes subconjuntos de características. Deste modo, três subconjuntos diferentes de características foram concebidos utilizando os métodos de filtros:

- **Classificação simples para todos os ataques:** Este método avalia o desempenho de classificação de características individuais em todos os ataques detectados. Estes resultados são comparados com as referências das taxas de classificação (Veja Seção 3.5.2 para definição das referências das taxas de classificação). Estas características, que possuem desempenho melhor que as taxas de referência, são consideradas como relevantes e selecionadas para o primeiro subconjunto de características.

- **Classificação simples para ataques individuais:** Este método avalia o desempenho de classificação para características individuais quando classificando ataques, um ataque de cada vez. As características relevantes (características que têm desempenho melhor que as taxas de referência) são selecionadas no segundo subconjunto de características.

- **Seleção com árvore de decisão:** Este método utiliza uma arvore de decisão, que é capaz de selecionar automaticamente as características baseado nas informações obtidas [Quinlan 1993]. Estas características selecionadas pela árvore de decisão são utilizadas como o terceiro subconjunto de características.

Três classificadores de redes neurais foram treinados utilizando os subconjuntos de características selecionados. O desempenho da classificação foi comparado ao método adotado neste estudo, quando utilizando o conjunto completo de características (Seção 3.5)

4.1.1 Estudos Relacionados em Detecção de Intrusão

Embora os sistemas de detecção de intrusão envolvam classificações de múltiplas características, poucos artigos têm sido publicados sobre problemas *FFS (Feature Subset Selection).* [Stolfo e Mok 1999] propôs um processo automatizado para descobrir características importantes utilizando regras de associação e freqüência de episódios. [Sung e Mukkamala 2003] classificou 41 características fornecidas pelo conjunto de dados do KDD Cup 99 [KDD 1999] utilizando *SVMs* e redes neurais.

4.2 Funções-Objetivo

A função-objetivo é a função de medida de custo para avaliar um subconjunto especifico de conjunto de dados. As funções-objetivo mais utilizadas em problemas de classificação são as de medidas de classificação e de precisão de predição, como erro quadrático médio, taxas de não-classificação, taxas de falso positivo e taxas de falso negativo.

Neste Capítulo, seis funções objetivo diferentes são utilizadas para estudar o impacto do processo *FSS*.

A primeira função-objetivo (Equação 4.1) mede a taxa de não-classificação de um subconjunto de características.

$$J_1(s) = \frac{N_{\mathrm{ofp}}(s) + N_{\mathrm{ofn}}(s)}{N} \tag{4.1}$$

Onde s é um dado conjunto de características; N é o número total de amostras; $N_{ofp}(s)$ é o numero de saídas falso-positivas de um classificador que utiliza o subconjunto de características s; $N_{ofn}(s)$ é o número de saídas falso-negativas de um classificador que utiliza o subconjunto de características s.

A segunda função objetivo (Equação 4.2) considera o tamanho $|s|$ do subconjunto de características, como um fator de custo.

$$J_2(s) = J_1(s) + J_1(s) \log_\alpha |s|$$

(4.2)

Na equação, o segundo termo mede o custo de computação relacionado ao subconjunto de características *s*, que é um produto de $J_1(s)$ e log_α |*s*| para excluir os subconjuntos de características com altas taxas de não-classificação e pequeno número de características. A constante α é determinada pelo custo máximo de computação, que é especificado pelos usuários. Por exemplo, se max |*s*| é o tamanho do conjunto completo de características, e se o custo máximo de computação é ajustado para *c*, α pode ser calculada utilizando-se a Equação 4.3.

$$\alpha = (\max |s|)^{1/c}$$

(4.3)

Várias curvas de |*s*| e log_α |*s*| com diferentes *c*'s e α's foram elaboradas na Figura 13 para ilustrar o impacto de α's nos custos computados.

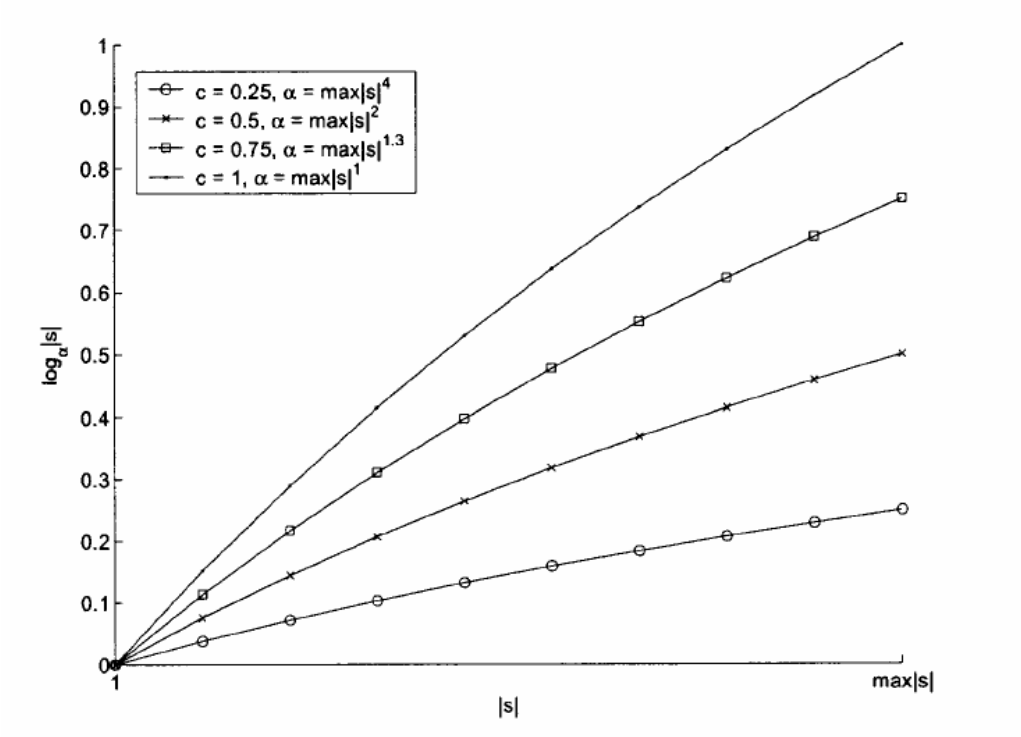

Figura 13 – Custo de computação com diferentes α´s.

Neste Capítulo, o máximo custo de computação *c* está fixado em 0,25, e consequentemente α está configurado como sendo *max* $|s|^4$.

Para comparar o impacto de funções de custo de computação diferentes, outra função-objetivo é utilizada para incluir o custo de computação como um fator quadrático (Equação 4.4).

$$J_3(s) = J_1(s) + \beta J_1(s) \sqrt{\frac{|s| - 1}{\max |s| - 1}} \tag{4.4}$$

Na equação, β é uma constante para modular o custo de computação. Ela está ajustada neste Capítulo para conformidade com o custo de computação *c*, que está fixado em 0,25.

Pelo fato dos falso-positivos gerarem um impacto maior no desempenho de um sistema de detecção de intrusão, três outras funções objetivo foram utilizadas para reduzir os falso-positivos (Equações 4.5 a 4.7).

$$J_4(s) = \frac{(1+\gamma)N_{\text{ofp}}(s) + (1-\gamma)N_{\text{ofn}}(s)}{N} \tag{4.5}$$

$$J_5(s) = J_4(s) + J_4(s)\log_\alpha |s| \tag{4.6}$$

$$J_6(s) = J_4(s) + \beta J_4(s)\sqrt{\frac{|s|-1}{\max|s|-1}} \tag{4.7}$$

Nas equações acima, γ é uma constante ajustável para destacar a importância de reduzir os falso-positivos. Neste Capítulo, γ está fixado em 0,6. Os subconjuntos de características desejáveis são os que possuírem os valores mais baixos nas seis funções-objetivo

4.3 Classificação utilizando Características Simples.

O desempenho de características individuais pode ser medido como as taxas mais baixas de não-classificação em curvas *ROC*, que são definidas como os gráficos entre as taxas de verdadeiro-positivo (TVP) e as taxas de falso-positivo (TFP) em diferentes limiares de detecção [Dhawan 2003].

O relacionamento entre a taxa de não-classificação, erro, TVP e TFP é dada pela Equação 4.8.

$$\text{error} = \frac{N_{\text{ofp}}(s) + N_{\text{ofn}}(s)}{N} = \frac{N_{\text{tp}}(1-\text{TPR}) + N_{\text{tn}}\text{FPR}}{N} \tag{4.8}$$

Onde N é o número total de amostras; N_{tp} é o número de amostras de ataques; N_{tn} é o número de amostras normais; N_{ofn} é o número de falso-negativos; e N_{ofp} é o número de falso-positivos.

A taxa de não-classificação mínima de uma característica pode ser calculada através do exame de todos os possíveis limiares de dados de treinamento.

Pode ser computacionalmente dispendioso examinar todos os possíveis limiares, mas para determinados registros de dados de treinamento, existem *m-1* possíveis limiares.

Uma vez que os dados de treinamento tenham sido ordenados, o processo de pesquisa pode ser executado em apenas uma passagem.

4.4 Resultados Experimentais

Os arquivos de dados utilizados neste Capítulo são arquivos similarmente distantes que foram gerados no ambiente de testes, ao monitorar 45 características de tráfego no conjunto de dados DARPA98 (Veja o Apêndice I para a descrição detalhada destas 45 características). Como listado na Seção 3.5.32, existem 59.226 registros, incluindo 1.074 registros de ataques e 58.152 registros normais, nos arquivos de dados.

Uma estratégia de procura heurística é empregada neste experimento, que se inicia pelas 45 características e após examinar o desempenho de características individuais. As características consideradas irrelevantes são removidas.

4.4.1 Filtragem de Características Baseada em Todos os Ataques

O desempenho de classificação destas características para detectar todos os quatro tipos de ataques é listado na Tabela 5, onde “Erro” corresponde às taxas de não-classificação das características individuais.

Tabela 5 – Características para todos os ataques.

#	ÍNDICE	NOME	ERRO
1	12	in.tcp-con-new-opened	0,00554
2	9	in.tcp-syn-pkt-rate	0,00572
3	3	in.ip-byt-rate	0,00591
4	2	in.ip-pkt-rate	0,00642
5	14	in.tcp-con-new-aborted	0,00775
6	17	in.tcp-con-diff-src	0,00864
7	13	in.tcp-con-new-closer	0,00954
8	18	in.tcp-com-dif-dst	0,00971
9	1	in.ip-pkt-len	0,00984
10	8	in.tcp-pkt-rate	0,0106
11	19	in.tcp-con-anomalous-entropy	0,0117
12	7	in.tcp-pkt-len	0,0122
13	22	in.icmp-byt-rate	0,0136
14	23	in.icmp-diff-src	0,0136
15	45	io.icmp-anomalous-echo-reply	0,0136
16	6	in.ip-csum-error-rate	0,0136
17	21	in.icmp-pkt-rate	0,0136
18	15	in.tcp-con-half-opened-ratio	0,0147
19	24	in.icmp-diff-dst	0,0156
20	16	in.tcp-con-duration	0,0170
21	11	in.tcp-rst-pkt-rate	0,0174
22	4	in.ip-frag-error-rate	0,0175
23	5	in.ip-defrag-error-rate	0,0176
24	28	in.udp-diff-src	0,0178
25	44	in.udp-diff-dst	0,0180
26-45	...	...	0,0181

Apenas 25 características foram consideradas relevantes na detecção de ataques *DoS*. As outras 20 características, classificadas de 26 a 45, têm a mesma taxa de não-classificação de 0,0181, que é a referência de taxa de não-classificação dos arquivos de dados DARPA98. Deste modo, apenas as primeiras 25 características são selecionadas no conjunto de dados. Para simplificação, este conjunto de características foi chamado de CJ25.

4.4.2 Filtragem de Características Baseada Ataques Individuais

Este experimento examina o desempenho de 45 características em ataques individuais, um ataque de cada vez.

Ataque *Neptune* : A classificação de desempenho de características individuais em ataques N*eptune* é apresentada na Tabela 6.

Tabela 6 – Características para o ataque *Neptune*.

#	ÍNDICE	NOME	ERRO
1	12	in.tcp-con-new-opened	0,00125
2	9	in.tcp-syn-pkt-rate	0,00138
3	14	in.tcp-con-new-aborted	0,00338
4	17	in.tcp-com-diff-src	0,00432
5	13	in.tcp-con-new-closed	0,00516
6	18	in.tcp-com-dif-dst	0,00545
7	8	in.tcp-pkt-rate	0,00640
8	18	in.tcp-con-anomalous-entropy	0,00748
9	7	in.ip-pkt-len	0,00790
10	3	in.ip-byt-rate	0,00992
11	15	in.tcp-com-half-opened-ratio	0,0102
12	2	in.ip-pkt-len	0,0105
13	16	in.tcp-con-duration	0,0127
14	11	in.tcp-rst-pkt-rate	0,0128
15	28	in.udp-diff-src	0,0131
16	44	out.udp-diff-dst	0,0133
17-45	...	...	0,0135

Como existem 798 amostras de ataques *Neptune* dentro do conjunto de DARPA98, a referência de taxa de não-classificação é 798/59.226 = 0,0135.

A partir da tabela 6 pode-se observar que apenas 16 características (entre 45) têm taxas de não-classificação mais baixas que a taxa de referência.

Como as taxas de não-classificação dos dois últimos resultado relevantes na tabela, *in.udp-diff-src* e *out.udp-diff-dst*, estão muito próximos da taxa de referência, e como estes parâmetros são tráfego *UDP*, que intuitivamente não deveriam ter relação com o tráfego *TCP*, estes dois parâmetros também não foram considerados.

Ataque *Smurf*: O desempenho de classificação na detecção do ataque *Smurf* está listado na Tabela 7. Como existem 267 amostras de ataques *Smurf* nos conjuntos de dados DARPA98, a taxa de referência é 267/59.226=0,00451.

Tabela 7 – Características para o ataque *Smurf*.

#	ÍNDICE	NOME	ERRO
1	22	in.icmp-con-byt-rate	0,000354
2	23	in.icmp-diff-src	0,000354
3	45	io.cimp-anomalous-echo-reply	0,000354
4	6	in.ip-csum-error-rate	0,000388
5	21	in.icmp-pkt-rate	0,000388
6	24	in.icmp-diff-dst	0,002126
7-45	...	...	0,00451

Pode-se observar que apenas seis parâmetros possuem taxas de não-classificação mais baixas que a taxa de referência. Portanto, estes seis parâmetros serão considerados.

Ataques *Pod* e *Teardop*: Como os ataques *Pod* e *Teardrop* possuem padrões de tráfego similares, estes dois ataques serão agrupados e analisados em conjunto. O desempenho de classificação das 45 características destes dois ataques está listado na Tabela 8.

Tabela 8 – Característica para o ataque *Pod*.

#	ÍNDICE	NOME	ERRO
1	4	in.ip-frag-rate	0,000118
2	5	in.ip-defrag-error-rate	0,000152
3-45	...	...	0,000608

Como existem 24 amostras de ataques *Pod* e 12 amostras de ataques *Teardrop* no conjunto de dados DARPA98, a referência de ataques para *Pod* e *Teardrop* é 36/59.226=0,000608. A tabela 8 indica que existem apenas duas características, "*in.ip-frag-rate*" e "*in.ip-defrag-error.rate*" que são relevantes para detectar estes ataques.

Subconjunto de Características Selecionado: Baseando-se no conjunto de tabelas anteriores, um subconjunto de vinte e duas características foi selecionado e listado na Tabela 9. Este subconjunto de características será chamado de CJ22 para referência.

Tabela 9 – Características em CJ22.

ÍNDICE	NOME
2	in.ip-pkt-rate
3	in.ip-byt-rate
4	in.ip-frag-rate
5	in.ip-defrag-error-rate
6	in.ip.csum-error-rate
7	in.tcp-pkt-len
8	in.tcp-pkt-rate
9	in.tcp-syn-pkt-rate
11	in.tcp.rst.pkt-rate
12	in.tcp-con.new-opened
13	in.tcp.con-new-closed
14	in.tcp-con-new-aborted
15	in.tcp-con-half-opened-ratio
16	in.tcp-con-duration
17	in.tcp-con-diff-src
18	in.tcp-com-diff-dst
19	in.tcp-com-anomalous-entropy
21	in.icmp-pkt-rate
22	in.icmp-byt-rate
23	in.icmp-diff-src
24	in.icmp-diff-dst
45	io.icmp-anomalous-echo-reply

4.4.3 Subconjunto de Dados da Árvore de Decisão

Uma árvore de decisão [Borgelt 2006] foi utilizada com os dados do DARPA98. Apenas 8 características foram selecionadas pela árvore (Tabela 10). Observou-se que as características selecionadas pela árvore de decisão são completamente diferentes das características listadas na Tabela 9. Para referência, este subconjunto de dados será chamado de CJ8.

Tabela 10 – Características selecionadas pela árvore de decisão.

ÍNDICE	NOME
4	in.ip-frag-rate
12	in.tcp-con.new-opened
13	in.tcp.con-new-closed
14	in.tcp-con-new-aborted
17	in.tcp-con-diff-src
22	in.icmp-byt-rate
33	out.tcp-pkt-len
34	out-tcp-pkt-rate

4.4.4 Comparação dos Diferentes Conjuntos de Características

As redes neurais foram treinadas utilizando-se os subconjuntos de características nos experimentos acima. As taxas de não-classificação destas redes neurais, após serem treinadas em 2000 "*epochs*" estão tabuladas na Tabela 11, juntamente com os valores de seis funções-objetivo, que são definidas na Seção 4.2.

Tabela 11 – Desempenho dos diferentes conjuntos de características.

S	CJ45	CJ25	CJ22	CJ8	CJ1
|s|	45	25	22	8	1
Error	0,000540	0,000371	0,000355.	0,000321	0,00554
TFP	0,000422	0,000310	0,000327	0,000275	0,000396
TFN	0,006520	0,003720	0,00186	0,00279	0,28400
J1(s)	0,000540	0,000371	0,000355	0,000321	0,00554
J2(s)	0,000675	0,000499	0,000427	0,000365,	0,00554
J3(s)	0,000675	0,000440	0,000416	0,000353	0,00554
J4(s)	0,000710	0,000514	0,000527	0,000542	0,00268
J5(s)	0,000888	0,000623	0,000634,	0,000514	0,00268
J6(s)	0,000888	0,000609	0,000618	0,000497	0,00268

Na tabela o "CJ45" corresponde ao desempenho do sistema quando se utilizam as 45 características (Veja Seção 3.5.2).

Os resultados apresentam o "CJ8" com a mais baixa taxa de não-classificação e os valores mínimos nas funções-objetivo dentre os quatro conjuntos. Isto demonstra que a precisão da classificação e a eficácia de computação podem melhorar de modo significativo através da seleção apropriada de um subconjunto de características.

4.5 Conclusões

Este capítulo construiu três subconjuntos de características diferentes utilizando-se de métodos de filtragem de características, e avaliou o desempenho correspondente de classificação.

Os resultados experimentais do conjunto de dados DARPA98 indicaram que a precisão da classificação e a eficácia de computação podem melhorar de modo significativo através da seleção apropriada de um subconjunto de características.

Dentre os quatro conjuntos de características, o CJ8 obteve o melhor desempenho, o qual inclui oito características: *in.ip.frag-rate*, *in.tcp-com-new-opened*, *in.tcp-com-new-close*,

in.tcp-new-aborted, *in.tcp-com-diff-src*, *in-icmp-byt-rate*, *out.tcp-pkt-len* e *out.tcp-pkt-rate.* Utilizando-se as oito características em CJ8, é possível reduzir efetivamente os valores das quatro funções-objetivo pela metade. Isto prova a importância da escolha apropriada do conjunto de características a serem monitoradas em um sistema de detecção de intrusão para maximizar a precisão de classificação e minimizar os custos computacionais. Entretanto, é importante observar que este conjunto de características pode não ser o melhor em termos de taxa de não-classificação e número de características para a detecção de ataques *DoS*.

Como relatado por [John et al.1994], o processo de seleção de subconjuntos de características não depende apenas das características e das funções-objetivos, mas também do algoritmo de aprendizagem.

O processo de identificação de características é baseado em dados, no qual os subconjuntos de características selecionadas são muito dependentes dos dados de treinamento. Portanto, atenção especial é necessária para evitar o "*over-fitting*" do subconjunto de características para um conjunto de dados de treinamento específico.

Como existe uma constante flutuação e evolução no tráfego de fundo e de ataque na Internet, o objetivo real da seleção de características não é encontrar um subconjunto de características, que pode ser ótima para um conjunto especifico de dados, e inadequada ou pobre para outros conjuntos de dados.

É importante selecionar um subconjunto de características que seja capaz de operar em condições ótimas, ou próximas de ótimas, sobre uma faixa ampla de configurações de rede. O próximo passo neste estudo é aplicar este processo de seleção de características aos dados coletados das várias fontes para entender as correlações entre os métodos adotados neste estudo e os ataques DoS.

5 Otimização

Os resultados da Seção 3.5 indicam que os método utilizados neste estudo podem efetivamente identificar ataques *DoS* utilizando-se estatísticas PDF e classificadores neurais com grande precisão. Esta seção relata uma série de pesquisas para aperfeiçoar os métodos utilizados neste estudo.

O estudo de vários algoritmos de particionamento PDF é relatado na Seção 5.1. A Seção 5.2 explora a possibilidade de reduzir o esforço de computação e a complexidade de armazenamento através de compressão *Wavelet*. A Seção 5.3 compara o desempenho de diferentes métricas de similaridade. A Seção 5.4 testa o desempenho de classificação em diferentes redes neurais.

5.1 Estudo de Algoritmos de Particionamento PDF

A representação das medições de rede em formatos PDF envolve o particionamento de espaços amostrais em um conjunto completo e não sobreposto de partições e o cálculo das freqüências dos eventos observados dentro das partições.

Os algoritmos PDF tradicionais montam histogramas PDF utilizando partições de tamanhos iguais, e o número de partições utilizado é determinado pela experimentação.

O NIDES [Javitz e Valdes 1993] utiliza 32 partições escolhidas para serem grandes o suficiente, de modo que os detalhes não sejam perdidos, com o custo adicional de complexidade computacional. Mas a escolha da estrutura de particionamento pode ter conseqüências significativas na precisão e na eficácia da representação numérica de um parâmetro monitorado. Poucas investigações mais profundas parecem ter sido realizadas para contornar este problema. Existem três esquemas diferentes de particionamento: uniforme, logarítmico e percentil, que foram testados em [Zhang et al. 2002a]:

Esquema de Particionamento Uniforme: O algoritmo de particionamento uniforme divide o espaço amostral em partições de igual tamanho. Assumindo-se x_0, x_1,...x_N como partições de um espaço amostral com x_0 como o mínimo e x_N como o máximo, onde N é o número total de partições. A partição pode ser calculada com a Equação 5.1

$$\begin{aligned} x_0 &< x_1 < x_2 < \ldots < x_N \\ x_1 - x_0 &= x_2 - x_1 = \ldots = x_N - x_{N-1} \\ x_i &= x_0 + i \times \frac{x_N - x_0}{N} \end{aligned}$$

(5.1)

Esquema de Particionamento Logarítmico: No esquema de particionamento logarítmico, o espaço amostral é segmentado utilizando-se a equação 5.2.

$$\begin{aligned} \frac{i}{N} &= \log_{(1+x_N-x_0)}(1 + x_i - x_0) \\ x_i &= x_0 - 1 + (1 + x_N - x_0)^{\frac{i}{N}} \end{aligned}$$

(5.2)

Esquema de Particionamento Percentil: Seja *F(x)* a função de distribuição acumulativa (*CDF – Cumulative Distribution Function*) de PDF. O algoritmo de particionamento percentil gera partições com valores iguais de probabilidade (ou massa), de acordo com a equação 5.3.

$$\begin{aligned} F(x_0) &= 0 \qquad F(x_N) = 1 \\ F(x_1) - F(x_0) &= \ldots = F(x_N) - F(x_{N-1}) \\ F(x_i) &= F^{-1}(\tfrac{i}{N}) \end{aligned}$$

(5.3)

Amostras de PDFs utilizando estes três esquemas de particionamento são ilustrados na Figura 14. Pode-se observar que o particionamento logarítmico possui comprimentos menores para os valores mais baixos e comprimentos maiores para valores mais altos. O particionamento percentil utilizar partições pequenas para representar o meio da distribuição, mas utiliza partições muito grandes para representar as duas áreas que se destacam. O particionamento uniforme dá uma ênfase igual para todas as partições.

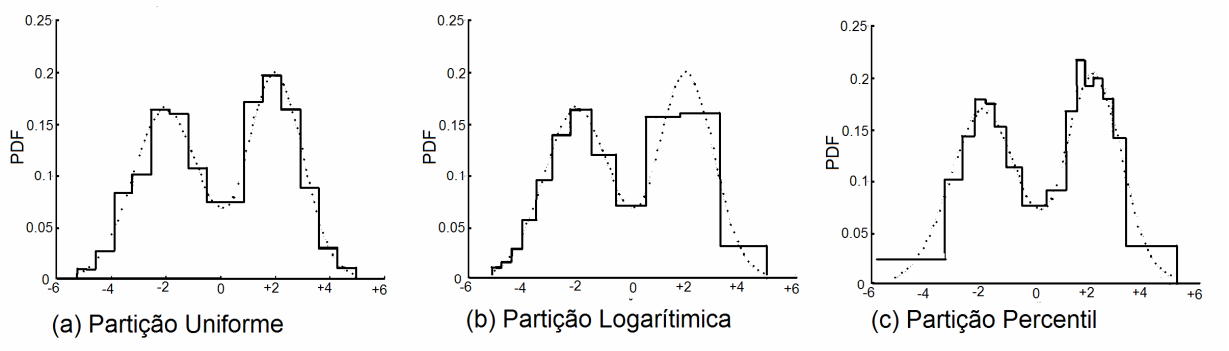

Figura 14 – Amostras PDF com diferentes algoritmos de particionamento.

As taxas de não-classificação em 8 e 16 partições, representando as funções dos níveis de ataques são exibidas na figura 15.

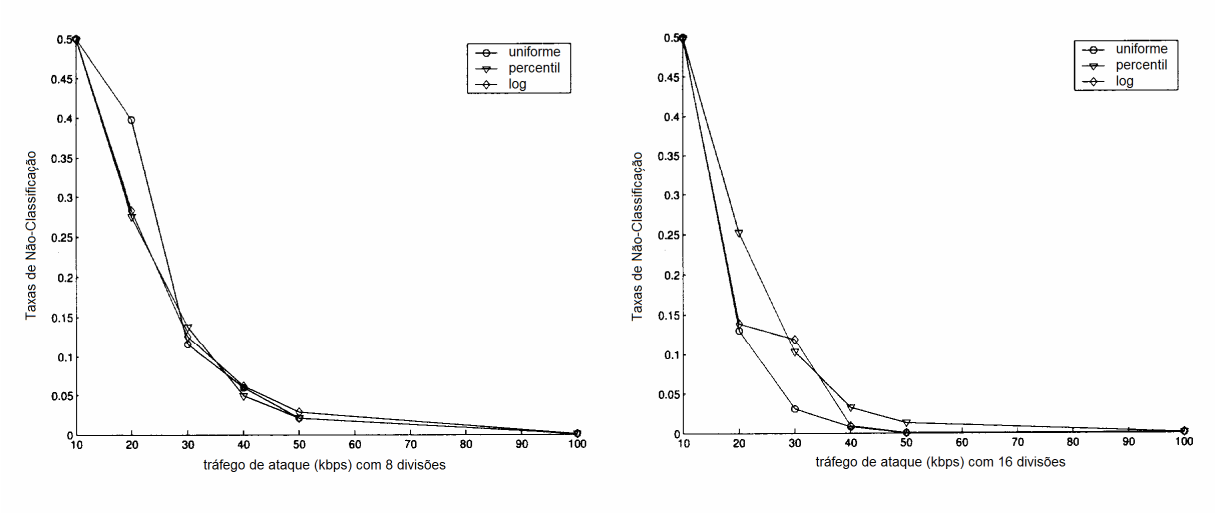

Figura 15 – Taxas de Não-Classificação versus Ataques.

É possível observar que à medida que o nível de ataque aumenta, a probabilidade de não-classificação dos sistemas diminui como já era esperado.

Quando os níveis de ataque são baixos (por exemplo, 40 kbps), é difícil para os esquemas PDF com poucas partições (por exemplo, 4 partições) detectar os ataques, enquanto os PDFs com mais partições (por exemplo, 16 partições) apresentam baixas taxas de não-classificação. Isto comprova a teoria de que a utilização de PDFs para representar variáveis de sistema melhora a precisão de classificação uma vez que isto abstrai informações mais detalhadas que podem ser úteis para sistemas de detecção de intrusão.

À medida que o numero de partições aumenta, o comportamento dos três algoritmos de particionamento se torna mais equivalente (com 64 partições, todos possuem o mesmo

desempenho). Isto pode ser explicado pela observação de que as diferenças entre os parâmetros monitorados reais e seus histogramas diminuem quando mais partições são utilizadas para representá-los.

Com o número total de partições variando de 4 partições a 64 partições, a taxa de não-classificação inicialmente diminui (o que representa melhoria de desempenho), mas quando o número total de partições se aproxima de 16, as curvas se tornam quase planas, demonstrando então que não houve melhorias de desempenho. Este comportamento sugere que as representações PDF com 16 ou mais partições podem ser as melhores escolhas para tratar a complexidade e o desempenho do sistema [Zhang et al. 2002a].

5.2 Compressão de PDFs com Compressão de Onda

A montagem e a manipulação de PDFs consome muito do poder de processamento dos sistemas, além de recursos de memória e armazenamento. Portanto, um algoritmo de compressão de dados mais eficiente pode ter o potencial de melhorar significativamente a eficiência do sistema. Esta seção apresenta os experimentos na compressão PDF de dados, utilizando compressão *wavelet*, como utilizado em [Vilegas et al. 2006].

Entre os diversos tipos de tipos de *wavelets* existentes [Ogden 1997] [Mallat 1999] [Daubechies 1992] [Donoho e Ducan 2004], uma *wavelet* foi selecionada de cada uma das seguintes famílias: a *Harr*, a *Symlets*, a *Coiflets*, e a *Daubechies*. Amostras de PDFs comprimidos utilizando cada uma destes quatro algoritmos de compressão wavelet são exibidas na Figura 16.

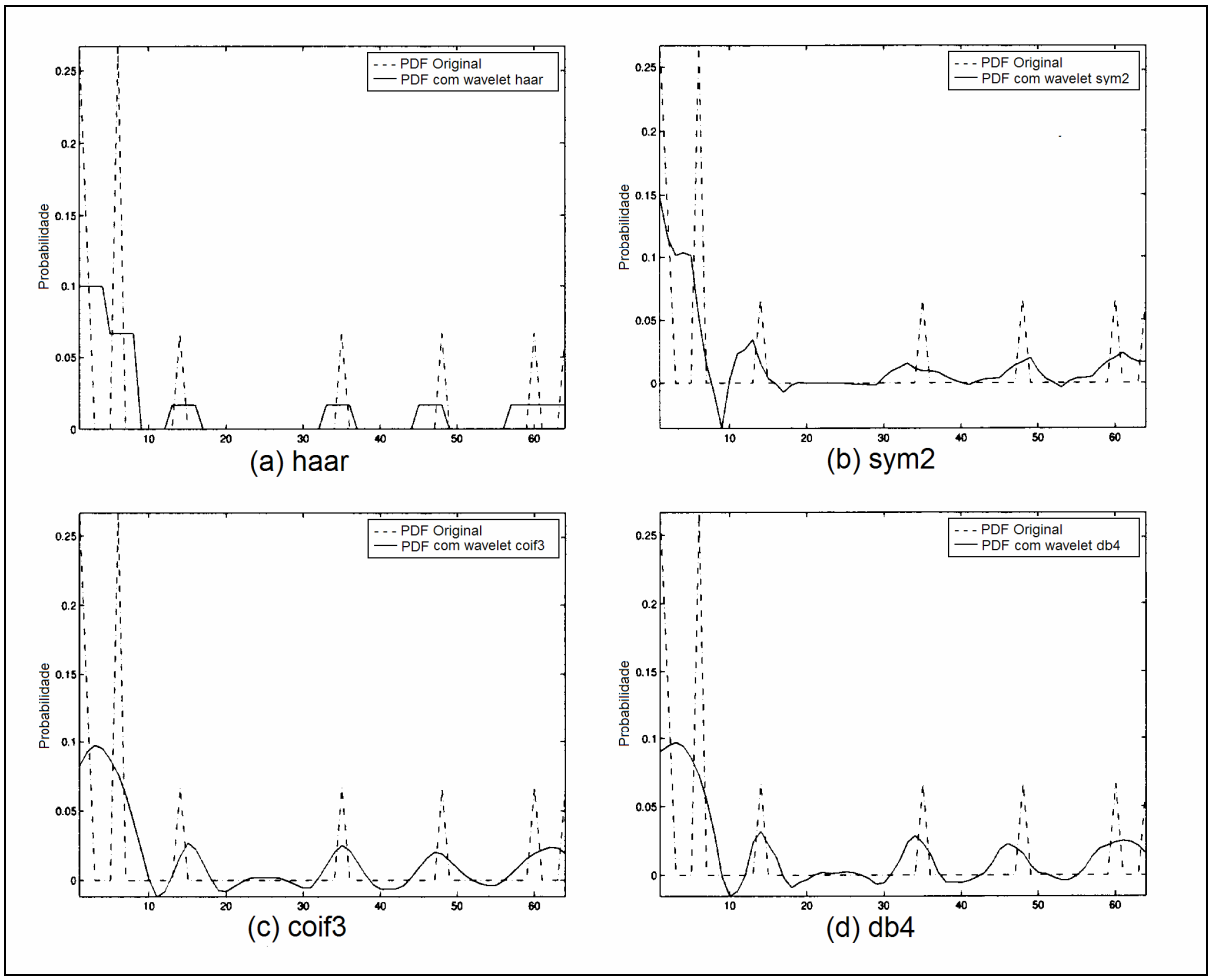

Figura 16 – Amostras de PDF com diferentes compressões wavelet.

Experimentos com várias faixas de compressão wavelet (Tabela 12), são relatadas em [Zhang et al. 2002b]. Nesta tabela, PDFs com compressão 1 correspondem a PDFs sem compressão. Observou-se que que *wavelets coif* e *db* requerem maiores números de coeficientes em cada faixa, enquanto as *wavelets haar* e *sym* apresentam as maiores quantidades de compressão.

Tabela 12 – Número de coeficiente Wavelet por faixas.

WAVELET	FAIXA 1	FAIXA 2	FAIXA 3	FAIXA 4	FAIXA 5
Harr	64	32	16	8	4
Sym	64	33	18	10	6
Coif	64	33	21	14	10
DB	64	40	28	22	19

Os algoritmos de *wavelet* mencionados foram utilizados para comprimir os arquivos de dados PDF coletados nas simulações OPNET (Seção 3.5.1), nos quais os PDFs são

representados utilizando-se o algoritmo de particionamento uniforme (Seção 5.1) com 64 partições dentro de conjuntos de coeficientes *wavelet*.

Para comparar o desempenho do sistema em várias faixas de compressão *wavelet*, os PDFs comprimidos foram reconstruídos dos coeficientes *wavelet* encontrados e processados pelos módulos estatísticos e pelos classificadores de redes neurais.

As taxas de não-classificação obtidas utilizando estes PDFs reconstruídos são representadas na Figura 17, na qual o eixo x representa as cinco faixas de compressão e o eixo y representa as taxas de não-classificação.

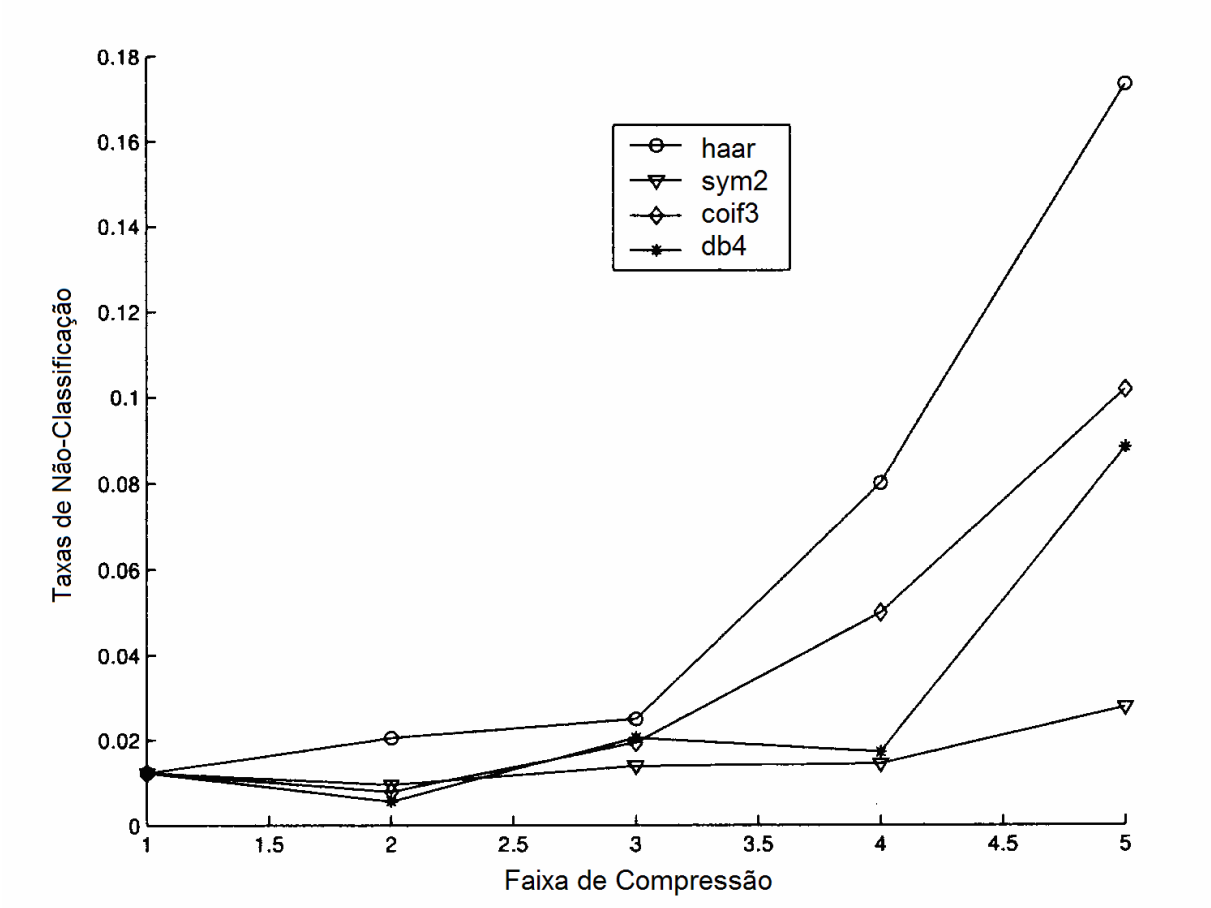

Figura 17 – Taxas de não-classificação com diferentes faixas de compressão wavelet.

É possível observar no gráfico que a curva do algoritmo de compressão *wavelet sym2* cresce lentamente, e mesmo para a faixa de compressão 5, o algoritmo ainda mostra grande desempenho com taxa de não-classificação em torno de 2%. Para as outras três *wavelets*, o desempenho é satisfatório para as faixas de 1 a 3, mas com deterioração nas faixas 4 e 5.

Os resultados experimentais indicaram que o algoritmo de compressão *wavelet sym2* é a escolha mais apropriada para compressão PDF. Na prática, observou-se que a compressão *wavelet sym2* na faixa de 3 (18 coeficientes *wavelet*) é a escolha mais adequada para o sistema manter um desempenho satisfatório enquanto aumenta a eficiência dos recursos em todos os métodos.

5.3 Eficácia das Métricas de Similaridade

Esta seção investiga a eficácia das diferentes métricas de similaridade na detecção de intrusão. As quatro métricas estudadas foram: o teste $\chi 2$ (equação 3.1); o teste *KS* (Equação 3.2); o teste *AKS* (Equação 3.3); e a estatística simplificada de número único *SG (Single-number)* (Equação 5.4). O teste *SG* foi utilizado para simular a detecção de intrusão utilizando parâmetros de número único.

$$D = \frac{p(r)}{p_{max}} \tag{5.4}$$

Onde *r* é o valor médio do parâmetro monitorado dentro de uma janela de tempo; *p(r)* é a probabilidade de *r* na distribuiçao em referência; e p_{max} é a máxima probabilidade da distribuição em referência. O desempenho de classificação destas quatro métricas, quando detectados no conjunto de dados OPNET, é apresentado na Figura 18.

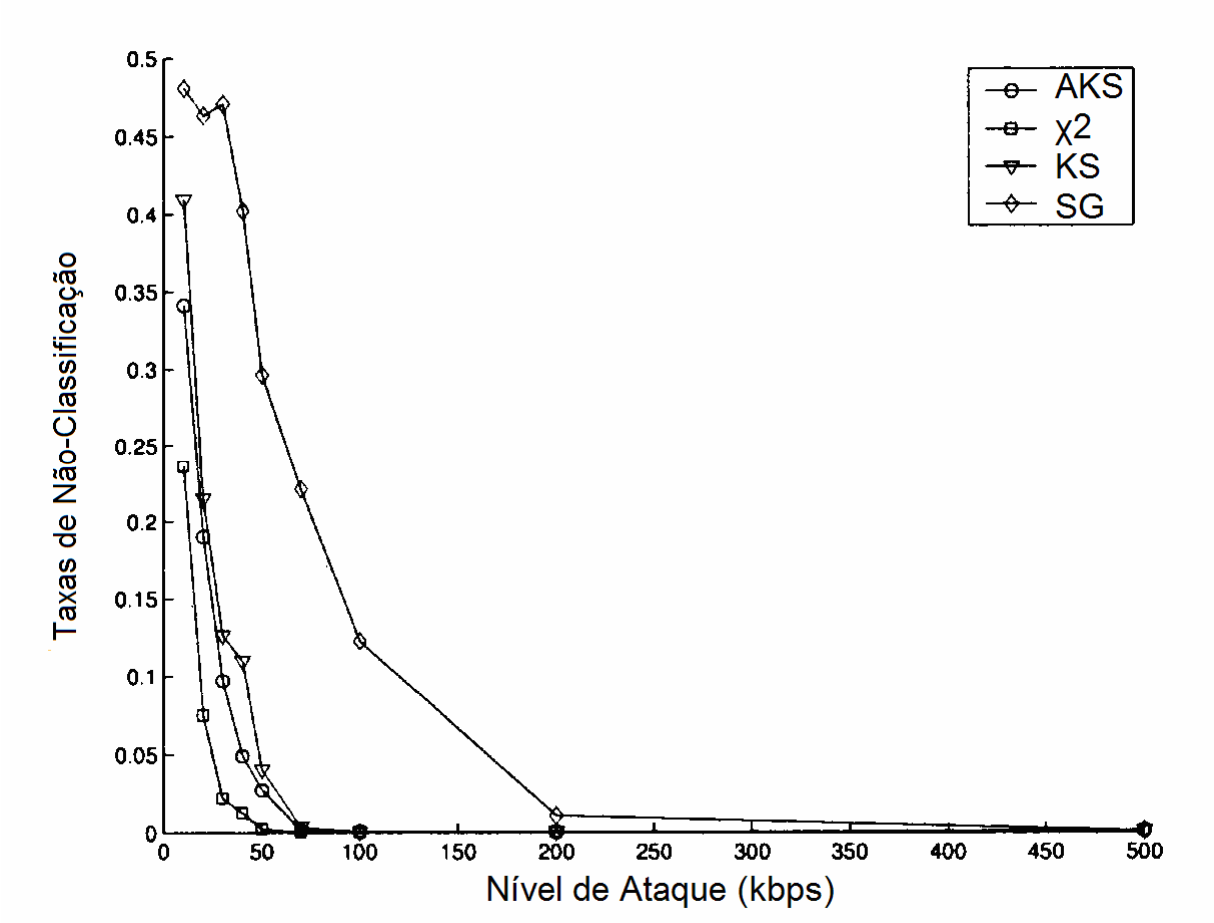

Figura 18 – As taxas de não-classificação de diferentes métricas de similaridade.

A métrica *SG* tem o pior desempenho, especialmente quando a intensidade de ataque varia de baixa para média. Isto indica que na utilização de estatísticas PDF, os sistemas de detecção de intrusão podem melhorar potencialmente a taxa de detecção inicial de ataques *DoS*.

A métrica *AKS* teve um desempenho ligeiramente melhor que a estatística *KS*.

O teste $\chi 2$ obteve as mais baixas taxas de não-classificação quando a intensidade de ataques era baixa. Uma explicação aceitável para esta observação é que, pelo fato das distribuições das simulações OPNET serem "*Poisson*" e "*Exponencial*", que são "sem-memória" e individualmente independentes, as distâncias resultantes $\chi 2$ seguem estas distribuições teóricas. Portanto, o teste $\chi 2$ pode ser preciso em apontar a diferença entre PDFs normais e de ataques. Entretanto, nos ambientes do mundo real, onde as distribuições de tráfego são frequentemente caóticas [Leland et al. 1994], as considerações sobre "*Poisson*" e "*Exponencial*" não são mais válidas. Pela inerente sensibilidade de distribuição, o teste de $\chi 2$

pode não ser uma boa escolha para a detecção em uma rede verdadeira. Portanto, o teste *AKS* foi escolhido como a métrica de similaridade para a maior parte deste estudo.

5.4 Classificadores de Redes Neurais

As redes neurais são consideradas eficientes para classificar padrões adaptativamente, mas a alta intensidade de computação, além de longos ciclos, tem dificultado a utilização destas aplicações em sistemas de tempo-real.

Em [Kim, Jo e Suh 2006], [Ghosh, Wanken e Charron 1998] e [Bonifacio et al. 1998], redes neurais *BP* foram utilizadas para detectar atividades anômalas de usuários.

Em [Zhang et al. 2002b], um paradigma de rede neural híbrida, chamada de rede *perceptron-backpropagation-hybrid* (*PBH*) [Dillon e Manikopoulos 1991], que é uma superposição de uma rede *perceptron* e uma pequena rede *backpropagation*, foi aplicada para detectar intrusões.

A decisão mais importante na elaboração de módulos de anomalias de muitos sistemas de detecção de intrusão é a escolha das técnicas de classificação, que têm um grande impacto no desempenho e eficiência, mas poucas pesquisas têm sido realizadas para comparar a eficácia de classificadores de redes neurais aplicadas a problemas de detecção de intrusão.

Para investigar o desempenho das redes neurais, três tipos diferentes de redes neurais são examinados nesta seção: *Perceptron*, *BP* e *PBH*.

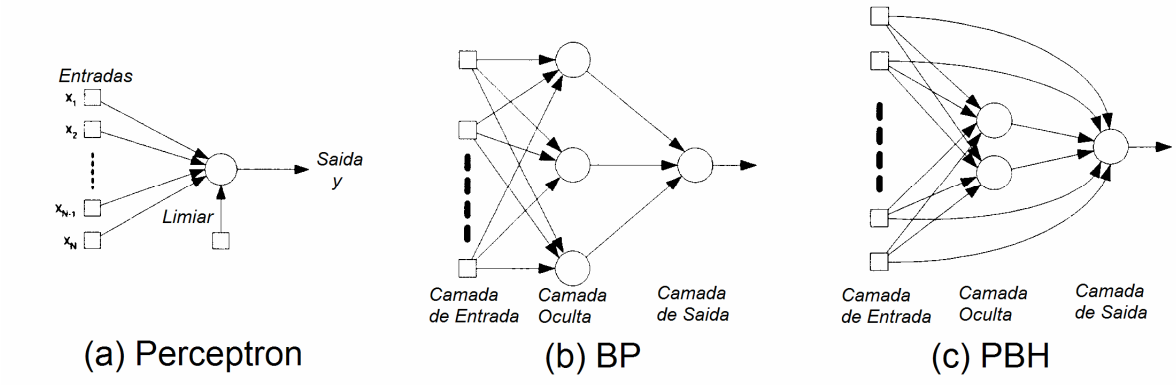

Figura 19 – Os modelos de classificadores neurais testados.

O *perceptron* [Haykin 1994], Figura 19 (a), é a forma mais simples de uma rede neural usada para a classificação de padrões linearmente separáveis. Ela consiste de um neurônio único com sinapses e limiares ajustáveis. Embora os conjuntos de dados em geral não sejam linearmente separáveis, o *perceptron* foi utilizado como referência para medir o desempenho das outras redes neurais.

A rede *backpropagation* [Haykin 1994], Figura 19 (b), ou *BP*, é uma rede multicamadas de alimentação à frente, que contém uma camada de entrada, uma ou mais camadas ocultas, e uma camada de saída. As *BPs* possuem grande capacidade de generalização e têm sido aplicadas com sucesso para resolver problemas complexos Foi testado o desempenho de classificação de redes *BP*, com o número de neurônios ocultos variando de 2 a 8.

A rede *perceptron-backpropagation híbrida* [Dillon e Manikopoulos 1991], ou *PBH*, Figura 19 (c), é uma superposição de uma rede *perceptron* e uma pequena rede *backpropagation*. As redes *PBH* são capazes de explorar correlações lineares e não lineares entre vetores de estímulo e valores de saída. Foi testado o desempenho de classificação de redes *PBH*, com o número de neurônios ocultos variando de 1 a 8. As taxas de não-classificação das redes neurais *BP* e *PBH* são apresentadas na Figura 20.

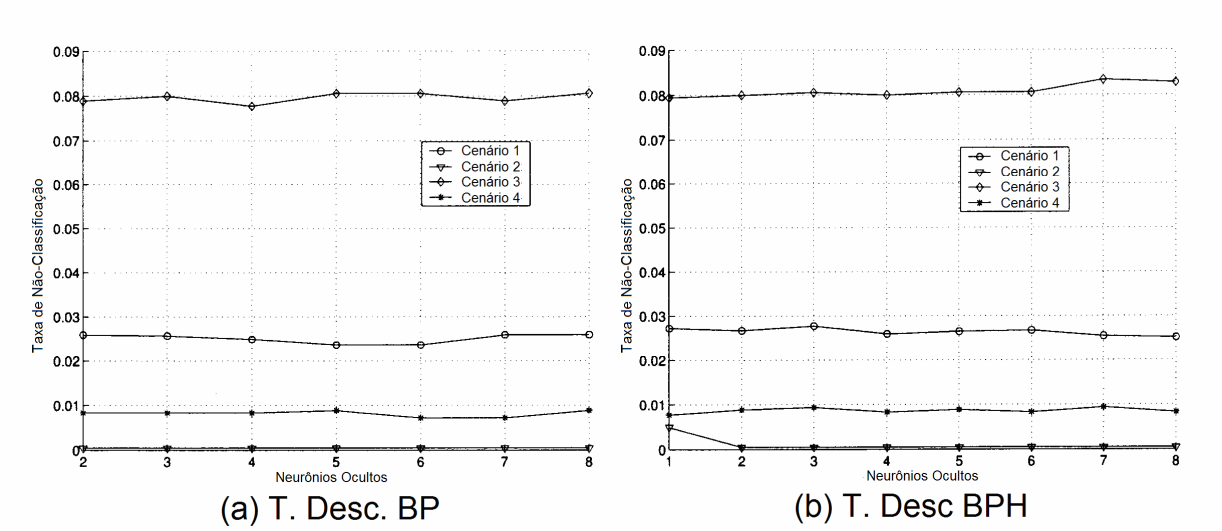

Figura 20 – Os resultados dos classificadores BP e PBH.

A conclusão é que as redes *BP* e *PBH* possuem desempenhos de detecção muito semelhantes, e ambas possuem desempenho melhor que a *perceptron*. As taxas de não-classificação destas duas redes não diminuem à medida que o número de neurônios ocultos

aumenta. Uma explicação para isto é que os espaços amostrais de padrões de tráfego de ataques e de tráfego normal foram efetivamente separados após terem sido processados pelas estatísticas PDF. Estes padrões não representaram grandes desafios para as redes *BP* e *PBH*. Portanto, a rede *BP* foi escolhida como classificador neste estudo.

6 Treinamento de Classificadores em uma Rede de Produção com Informações de Teste

6.1 Treinamento

Em um ambiente de produção, durante sua operação, é esperado que um sistema de detecção de intrusão descubra atividades anômalas. Na Figura 3, o classificador de padrões processa os padrões de atividades correntes e os classifica em uma tabela de similaridades comparando-os com atividades normais e anômalas. De maneira semelhante a muitas outras técnicas de aprendizagem com treinamento supervisionado, o classificador adotado, que é uma rede neural *back-propagation*, precisa ser treinado com uma variedade de registros de dados típicos (normais) e de ataques (anômalos) em quantidades suficientemente grandes.

Como é esperado distinguir diferentes classes de ataques ou ataques individuais, os dados para todos os tipos de ataques precisam estar disponíveis para treinamento. Estes dados irão permitir que o classificador de redes neurais aprenda a diferença entre atividades normais e anômalas, obtendo assim altas taxas de detecção e baixas taxas de alarmes falsos.

Em uma rede de testes de laboratório, pode-se investigar e registrar o disparo destes ataques conhecidos, selecionando-os de uma biblioteca de ataques, com identificadores precisos. Estas investigações de ataques podem ser executadas e registradas em vários tipos e níveis de tráfegos de fundo.

O desafio encontrado é que enquanto dados normais e de ataque são facilmente encontrados em uma rede de testes, apenas dados normais estão normalmente disponíveis em uma rede de produção. Deste modo, o classificador poderá ser treinado de modo inadequado para um novo ambiente de rede, e, portanto inaceitável para utilização modificações. Deve-se então considerar as duas seguintes premissas:

- A nova rede pode ser relativamente diferente da rede de testes, e o novo tráfego de fundo pode não ser semelhante ao tráfego de fundo da rede de testes.

- O IDS depende de treinamento que inclui dados de ataque para a nova rede.

Uma destas condições, ou mais frequentemente as duas, são casos mais reais que um novo sistema de detecção de intrusão pode encontrar. Portanto, o classificador de um sistema

de detecção de intrusão deverá ser treinado novamente em alguma fase, durante ou logo após sua instalação.

Uma ferramenta de detecção de intrusão que vai ser instalada em um ambiente de rede novo e real pode depender de um período inicial de adaptação, que irá fornecer estimativas suficientes do comportamento normal da rede. Em uma configuração típica, a maior parte do tráfego que um sistema de detecção de intrusão visualiza é normal, com apenas alguns registros de tráfego de ataque. Deste modo, algoritmos devem ser utilizados para separar o tráfego normal dos ataques que podem ocorrer na nova rede, baseando-se em considerações isoladas de comparação. Estes algoritmos, que empregam técnicas de clusterização, também devem ser investigados [Portnoy et al. 2001].

Existe a possibilidade de que durante a fase inicial, todo o tráfego na nova configuração seja efetivamente normal. Mesmo que algum tráfego de ataque possa ser encontrado na nova rede, será difícil neste estágio caracterizar que tipo de ataque está ocorrendo. Portanto, apenas modelos de tráfego normal na nova configuração de rede podem ser observados, mas nenhuma descrição mais clara de ataques de rede pode ser esperada.

Embora seja possível em teoria, não é adequado disparar ataques na rede a ser protegida, apenas para se beneficiar na fase de aprendizagem do sistema de detecção de intrusão. Isto inclusive pode não ser permitido. Então, de algum modo, usando dados de tráfego normal e de ataque da rede de teste, e dados de tráfego normal da nova rede, é necessário garantir que o classificador seja adequadamente treinado.

Duas técnicas alternativas de solução para este desafio, o classificador *re-use* e o classificador *grafted*, foram propostas em [Zhang et al. 2002b]:

- O método classificador *re-use* é direto e consiste na utilização do classificador da rede de testes na nova rede, como treinado anteriormente apenas na rede de testes, sem modificações no inicio. Este método funciona adequadamente se a nova rede é semelhante em arquitetura e utilização à rede de testes, mas irá ter uma queda de desempenho se as duas configurações de rede forem muito diferentes.

- O método classificador *grafted* (transplantado) consiste basicamente na abstração de um banco de dados de ataques, num formato mais “puro”, da rede de testes, e

disparando estes ataques no tráfego normal da nova rede, “transplantando” assim os modelos de ataque da rede de testes para a rede de produção.

Deste modo, o método classificador *grafted* irá produzir amostras de tráfego de ataque e de tráfego normal nas configurações da nova rede, permitindo o treinamento do classificador na nova rede. Testes foram executados para investigar a eficácia destas duas técnicas em uma configuração experimental, onde os dois se demonstraram eficientes.

O método classificador *grafted* irá trabalhar adequadamente para a classe de parâmetros onde o efeito dos ataques é adicionar características ao tráfego de fundo, que é o caso dos ataques *DoS*. O objetivo destas técnicas não é gerar o melhor modelo final de um ataque, mas fornecer informações básicas iniciais para um modelo, permitindo que o sistema se adapte aprendendo a partir destas informações iniciais, gerando então um modelo de ataque mais adequado a partir de uma estimativa compatível.

Este estudo emprega um algoritmo adaptativo que ajusta os modelos normais e de ataques, usando as descrições dos dados normais e de ataques. O método classificador *grafted* como descrito acima, geralmente depende da natureza dos parâmetros e dos tipos de ataques monitorados. Basicamente, ele depende do pressuposto de que os modelos de tráfego normal variam muito de rede para rede enquanto os modelos de tráfego de ataque não variam na mesma proporção. Esta afirmação é aceitável intuitivamente e pode ser comprovadamente observada em muitos tipos de redes e ataques. O tráfego de rede de fundo normal muda ao longo do tempo e de rede para rede.

Por outro lado, os padrões de comportamento de vários tipos de ataques são comparativamente estacionários e apresentam similaridades de rede para rede. De modo mais detalhado, este método envolve a extração de padrões de ataques, para cada parâmetro monitorado, a partir da rede de testes, e sua aplicação na nova rede de produção. Em primeiro lugar, as amostras de tráfego de ataque e de fundo são extraídas dos dados coletados na rede de testes. Depois, as instâncias de ataques são calculadas removendo-se os componentes do tráfego normal dos modelos de ataques. Assim as amostras de ataques para a nova rede podem ser simuladas pela combinação e mistura das amostras normais da rede de produção com as instâncias de ataques que foram extraídas da rede de testes.

Finalmente, o classificador de detecção de intrusão pode ser treinado utilizando dados de ataques simulados na etapa anterior, junto com as amostras de dados normais. As

definições de amostras de ataques, amostras normais, instâncias de ataques e padrões de ataques, para cada parâmetro, são apresentadas a seguir:

- **Ataque Híbrido:** é o padrão de tráfego observado, que é representado no formato PDF, com tráfego normal e de ataque.
- **Amostra Normal:** é o padrão de tráfego observdo apenas com tráfego normal.
- **Ataque Abstrato:** é a instância do tráfego de ataque, na sua forma "pura", removendo-se os componentes de tráfego normal das amostras de ataque.
- **Padrões de Ataque:** é o banco de dados de todas as instâncias de ataque.

6.2 Modelagem e Simulação de Ataque

Os modelos de tráfego normal são em geral consideravelmente variados em uma rede, ao longo do tempo, e de rede para rede sendo difícil prever ou estimar modelos de tráfego normal. Entretanto, pelo fato dos esquemas e padrões de ataques específicos geralmente possuírem características especificas, seus efeitos no tráfego normal podem ser consistentes. Assim, é possível estimar modelos de ataques baseando-se no conhecimento dos dados coletados a partir de uma rede de testes, com o propósito de obter "informações iniciais" para alimentar a nova configuração de rede. O diagrama de fluxo de modelagem e simulação é apresentado na Figura 21.

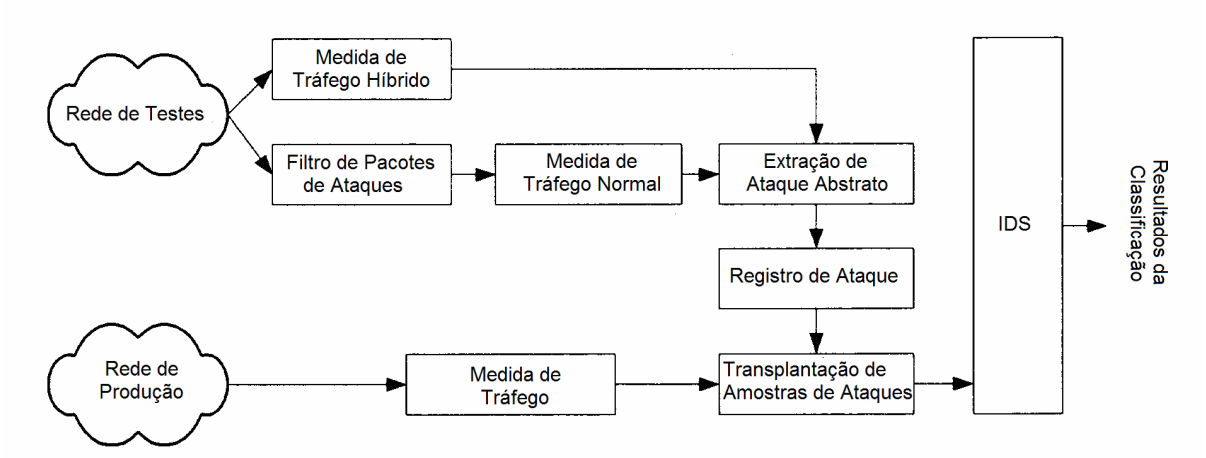

Figura 21 – Estimativa das fases de um ataque.

Os seguintes passos estão envolvidos neste processo:

1. **Amostragem de Tráfego Híbrido na Rede de Testes:** Mede o tráfego de rede, incluindo pacotes normais e de ataques, da rede de testes, em amostras de dados PDF.

2. **Filtro de Pacotes de Ataques:** Bloqueia os pacotes de ataques, para que apenas pacotes normais passem através deste bloqueio.

3. **Amostragem de Tráfego Normal na Rede de Testes:** Mede o tráfego de rede, incluindo apenas pacotes normais da rede de testes, em amostras de dados PDF (os pacotes de ataques são filtrados pelo Filtro de Pacotes de Ataques).

4. **Extração Abstrata de Ataques:** Extrai as instâncias de ataques através da remoção dos componentes de tráfego normal das amostras de ataques.

5. **Registro de Ataques:** É o banco de dados de todas as instâncias abstratas de ataques extraídos da etapa anterior.

6. **Amostragem de Tráfego Normal na Rede Alvo:** Mede os padrões de tráfego na rede alvo.

7. **Amostra de Ataque Transplantada:** Neste estágio, as instâncias de ataques abstratos, que foram coletadas na rede de testes, e as amostras normais que foram coletadas na rede alvo, tornam-se disponíveis. Entretanto, o objetivo real destes dados é simular amostras de ataques suficientes para serem usadas em treinamento, testes e validação de dados. Explicitamente este algoritmo precisa ser randomicamente gerado para presumir amostras de ataques que são PDFs que estatisticamente prevêm padrões de tráfego, pressupondo que os mesmos ataques serão observados na rede alvo. Assim estas amostras de ataques simulados, juntamente com as amostras de tráfego normal da rede alvo são utilizadas para treinar os classificadores.

6.2.1 Extração de Conteúdo de Ataque

Assume-se que *AH* é uma amostra de ataque, que contém tráfego normal e de ataque. *BN* é uma amostra normal, que contém apenas o tráfego normal. As equações para calcular a instância de ataque são dadas na Equação 6.1.

$$\begin{aligned} \mathrm{AA}(i) &= \frac{N_{\mathrm{AH}_t} * \mathrm{AH}_t(i) - N_{\mathrm{BN}_t} * \mathrm{BN}_t(i)}{N_{\mathrm{AH}_t} - N_{\mathrm{BN}_t}} \\ N_{\mathrm{AA}} &= N_{\mathrm{AH}_t} - N_{\mathrm{BN}_t} \end{aligned} \tag{6.1}$$

Onde:

- i: $i^{enésima}$ partição de uma amostra PDF.
- AH_t: Amostra de tráfego de ataque na rede de testes.
- BN_t: Amostra de tráfego normal na rede de testes.
- N_{AHt}: Número de amostras de *AHt.*
- N_{BNt}: Número de amostras de *BNt*.
- *AA*: Instância de abstração de ataques calculada.
- N_{AA}: Número de amostras de *AA*.

6.2.2 Amostras de Ataques Transplantadas

O transplante de amostras de ataque é um processo reverso de uma etapa de extração de instância de ataque. Primeiro, uma abstração de ataques, *AA*, é randomicamente escolhida nos registros de ataques; então a amostra de ataque a ser transplantada, *ATg*, é gerada pelo transplante da abstração de ataque selecionada dentro de uma amostra normal coletada na rede alvo. A equação de transplante de ataque é dada pela Equação 6.2.

$$\begin{aligned} \mathrm{AT}_g(i) &= \frac{N_{\mathrm{AA}} * \mathrm{AA}(i) + N_{\mathrm{BN}_p} * \mathrm{BN}_p(i)}{N_{\mathrm{AA}} + N_{\mathrm{BN}_p}} \\ N_{\mathrm{AT}_g} &= N_{\mathrm{AA}} + N_{\mathrm{BN}_p} \end{aligned} \tag{6.2}$$

Onde:

- i: $i^{enésima}$ partição de uma amostra PDF.
- *AA:* Instância de abstração de ataques calculada.
- N_{AA}: Número de amostras de *AA*.
- *BNp*: Amostra de tráfego normal na rede alvo de produção.
- N_{BNp}: Número de amostras de *BNp*.

6.2.3 Aplicação de PDF nas Amostras de Ataque

Exemplos de PDFs de amostras medidas de tráfego normal e de ataques, abstrações de ataques extraídos e amostras resultantes de ataques transplantados são apresentadas na Figura 22 para alguns parâmetros. Na Figura 22(a) e na Figura 22(b) estão amostras PDF medidas normal e de ataque. Na Figura 22(c) está a instância abstrata de ataque extraída, baseada nas Figura 22(a).e Figura 22(b). Na Figura 22(d) e na Figura 22(e) estão as amostras normal e de ataque na rede alvo. Na Figura 22(f) está a amostra transplantada de ataque simulado na rede alvo.

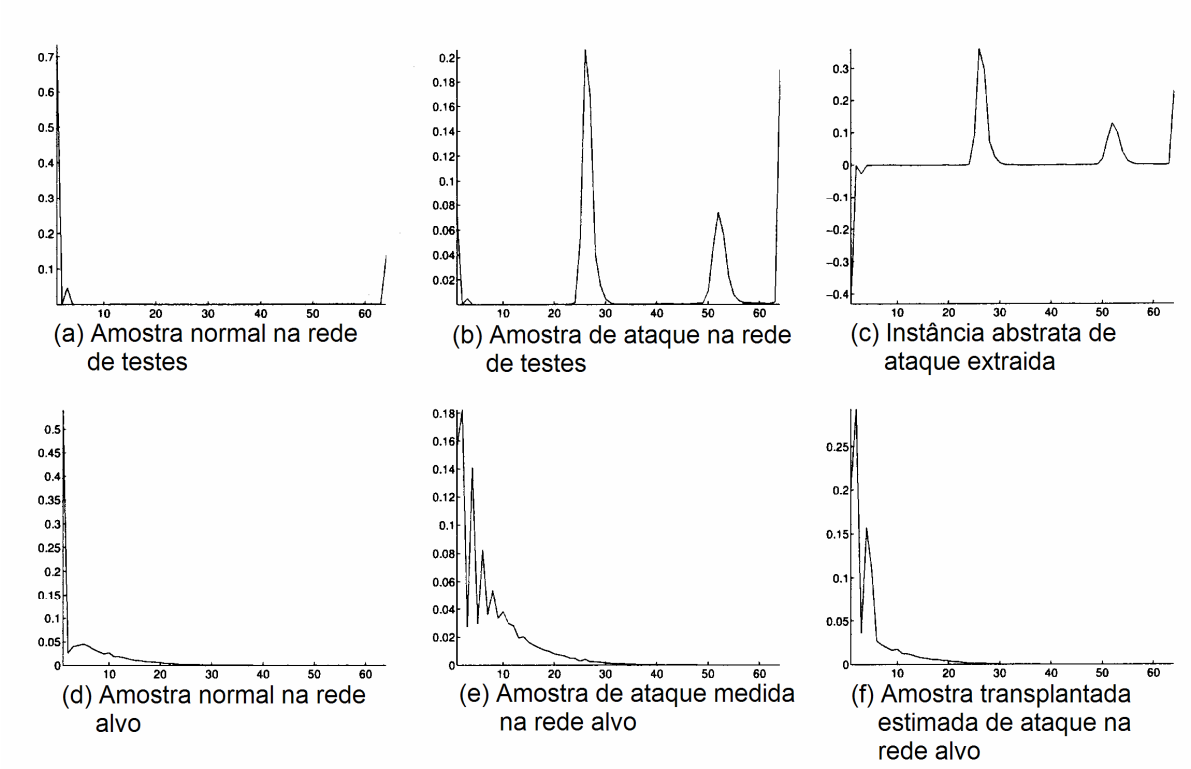

Figura 22 – Histogramas PDF correspondentes às etapas utilizadas na estimativa de uma amostra transplantada de ataque de um parâmetro.

Pelas figuras, é possível observar que embora as amostras de tráfego normal nas duas redes sejam completamente diferentes (como pode ser observado comparando-se a Figura 22(a) e Figura 22(d)), a amostra transplantada de ataque simulado é visivelmente semelhante à amostra de ataque medida na rede alvo (como pode ser observado comparando-se a Figura 22(f) com a Figura 22(e)).

6.3 Resultados com OPNET

A primeira experiência é projetada para avaliar o desempenho dos algoritmos *graft*ed e *re-use* quando a rede de testes e a rede alvo possuem configurações de rede semelhantes. Conjuntos de dados de simulação OPNET (Seção 3.5.1) são utilizados neste experimento.

A rede de 600 kbps (Tabela 2) é utilizada como "Rede de Testes A". Dados de tráfego de fundo e de ataque da rede de testes são utilizados para treinar o classificador de testes. Este classificador de testes é utilizado para investigar a eficácia do método de classificação *re-use*. Estes dados também são utilizados para calcular as instâncias abstratas de ataques através dos algoritmos descritos anteriormente que são parte do método de classificação *grafted*.

A rede de 2Mbps (Tabela 2) é utilizada como a "Rede de Produção B". O tráfego de fundo da rede B é utilizado para computar os modelos de fundo e então estimar os modelos de ataque na rede de produção.

Os modelos de ataque são gerados simulando amostras transplantadas para treinamento.

6.3.1 Modelos de Similaridades

Para quantificar melhor as similaridades entre os modelos medidos e estimados, três medidas de similaridade diferentes são calculadas utilizando-se a equação de similaridades apresentada na Equação 3.3:

1. A similaridade entre o modelo real de fundo e o modelo real de ataque medido (A_aB_a), onde o termo real se refere às quantidades medidas, ao invés das estimadas, na rede simulada.

2. A similaridade entre o modelo de fundo real e o modelo de ataque estimado (A_eB_a).

3. A similaridade entre o modelo de ataque real e o modelo de ataque estimado (A_eA_a).

4. A similaridade entre o modelo de fundo de 600 kbps e o modelo de fundo de 2Mbps ($B_{a600k}B_{a2M}$). As médias dos resultados de similaridade são apresentadas na Tabela 6.1.

Tabela 13 – Resultados da Similaridade Média

ATAQUE (kpbs)	A_aB_a	A_eB_a	A_eA_a	$B_{a600k}B_{a2M}$
10	0,958	0,982	0,962	0,565
50	0,904	0,927	0,941	0,603
100	0,837	0,840	0,896	0,555
200	0,706	0,677	0,839	0,599

Analisando-se a Tabela 13, pode-se observar que quando um nível de ataque é muito baixo comparando-se com o fundo, por exemplo, de 10 a 50 kbps, todos os modelos estão próximos uns dos outros, e na verdade, nenhum dos classificadores pode detectar ataques de maneira confiável.

Para níveis mais altos de ataques, por exemplo, de 100 a 200 kbps ou mais, os modelos estimados de ataques permanecem muito semelhantes aos modelos de ataques medidos, mas diferentes dos modelos normais reais. Isto sugere que os modelos de ataques podem ser estimados com adequada precisão porque eles são muito similares aos modelos reais. A ultima coluna indica que o fundo em redes de 600 kbps e 2Mbps são consistentemente muito diferentes uns dos outros.

6.3.2 Desempenho da Classificação

Três experimentos de classificação são executados para examinar o desempenho de classificação para redes, através das duas técnicas, *re-use* e *grafted:*

1. Os dados reais PDF da rede de produção B são utilizados para testes de treinamento e validação do classificador. Os resultados são utilizados como referência para o desempenho do sistema, devendo ser os melhores que podem ser obtidos para a rede real de 2Mbps. Para referência, este experimento é chamado de *optimum*.

2. Os dados PDF gerados, utilizando-se o classificador *grafted* são usados para treinamento do classificador, enquanto os dados reais são usados para testes. Os resultados refletem o desempenho do método "*grafted*".

3. Os dados PDF de 600 kbps são utilizados para treinamento e testes do classificador. Este classificador é utilizado para validar os dados PDF da rede de produção. Isto corresponde a examinar a eficiência do método *re-use*.

As curvas ROC (Receiver Operating Characteristic) dos três experimentos sob diferentes níveis de ataques e apresentado na Figura 23.

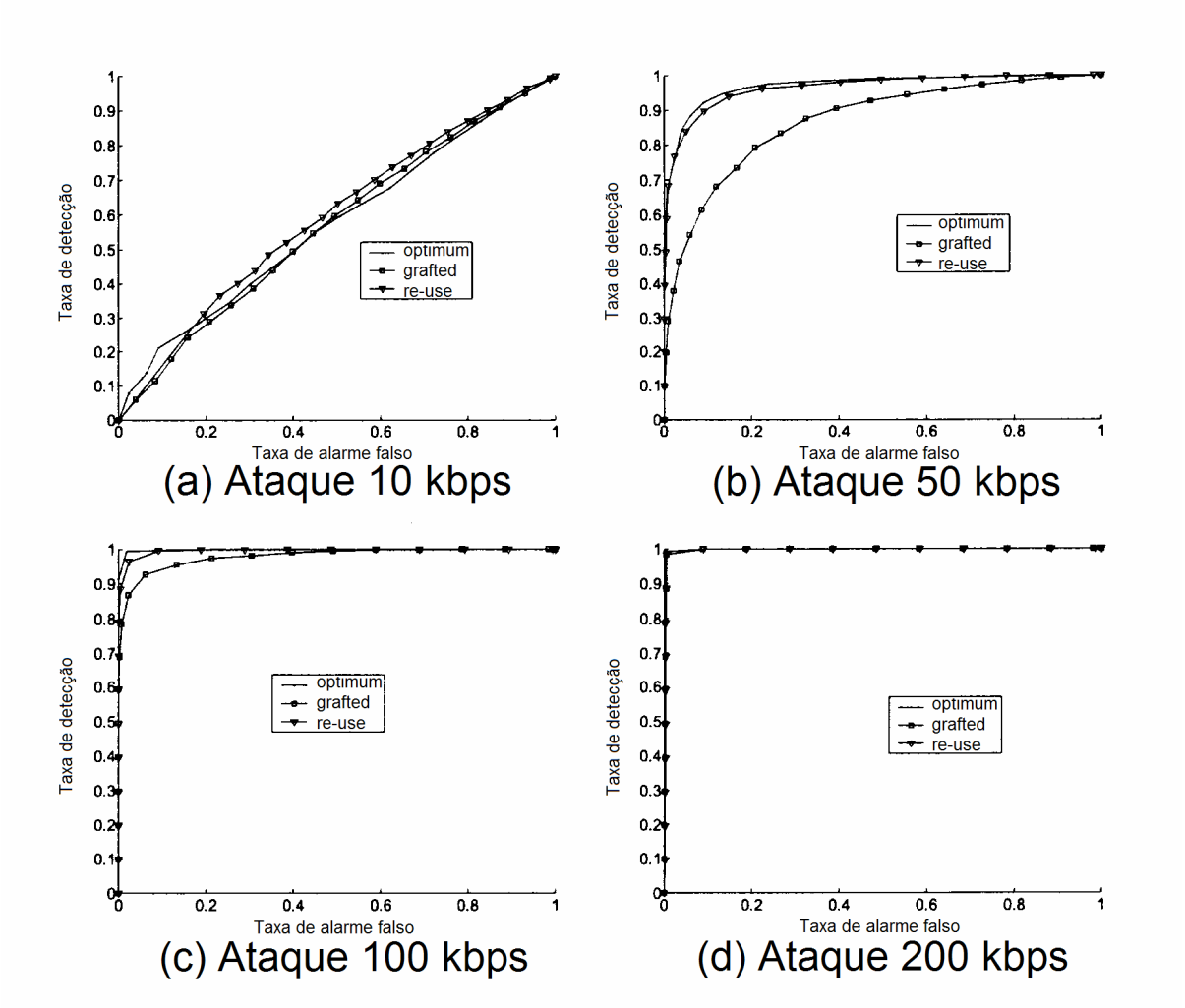

Figura 23 – Curvas ROC para diferentes níveis de ataques.

Através dos gráficos, pode-se observar que o desempenho de classificação melhora à medida que a intensidade dos ataques é maior, como esperado.

O desempenho das medidas *optimum* é o melhor. Como mencionando anteriormente, para níveis de ataques muito baixos, todos os classificadores basicamente falham e qualquer resultado de classificação medida é de pouco significado; este é o caso do gráfico de 10 kbps.

Para níveis mais altos de taxas de ataques, mais significativos, o desempenho da classificação dos dois métodos (classificadores *re-use* e *grafted*) são mais significativos e basicamente se assemelham, sendo um pouco inferior aos resultados *optimum*.

As taxas de não-classificação para *optimum*, *grafted* e *re-use* são apresentadas na Tabela 14. Os valores observados na tabela acompanham o que foi observado na Figura 23.

Tabela 14 – Taxas de não-classificação após treinamento para 300 epochs.

ATAQUES (kbps)	OPTIMU	GRAFTED	RE-USE
10	0,432	0,446	0,424
50	0,0783	0,204	0,0944
100	0,00778	0,0633	0,0267
200	0	0,00278	0,00556

As taxas de não-classificação para o experimento *optimum* são melhores, mas estão próximas dos experimentos com classificadores *grafted* e *re-use*. Isto indica que estes dois últimos modelos e seus algoritmos de estimativas podem ser utilizados como efetivos pontos de partida, quando se migra um sistema de detecção de intrusão de anomalias de uma rede de testes para uma nova configuração de rede não tão semelhante, onde modelos de ataques mais precisos não estão disponíveis. Entretanto, o método *grafted* possui desempenho melhor quando o novo ambiente de rede se torna cada vez mais diferente do ambiente de testes.

6.4. Conclusões

A classificação de um sistema de detecção de intrusão, que é adequadamente treinada em um ambiente de testes, pode ser treinada de modo inadequado em uma rede de configurações diferentes. Isto ocorre porque embora os dados de tráfego de fundo típicos estão disponíveis na nova rede, os dados de ataque normalmente não estão.

Para resolver este problema, duas técnicas, classificadores *re-use* e *grafted*, são propostas e apresentadas neste capitulo para disparar os classificadores em uma rede de produção através da utilização de informações de ataques obtidas na rede de testes.

Um experimento foi executado em um protótipo de detecção de intrusão, para comparar as técnicas *optimum* (quando o classificador conhece os ataques na nova rede) com os métodos de classificação *re-use* e *grafted*. Foram utilizados dados gerados por simulações OPNET em várias configurações de tráfego de ataque e tráfego de fundo.

Foi constatado que os resultados (Seção 6.3) do desempenho da classificação para os classificadores *grafted* e *optimum* melhoram à medida que a intensidade dos ataques aumenta, enquanto o desempenho da classificação optimum é a melhor em todos os cenários, como esperado. Isto indica que tanto os algoritmos *grafted* e *re-use* podem ser utilizados para

disparar métodos de treinamento de classificadores para um novo ambiente quando as informações sobre ataques forem insuficientes.

7 Conclusões

Este estudo apresentou os esforços de pesquisa na aplicação de estatísticas PDF e classificação de redes neurais na detecção de intrusão e correlação de alertas. A seguir, são descritos os tópicos de pesquisa analisados e reportados neste estudo.

7.1 Arquitetura de Sistema e o Modelo Estatístico

O sistema foi testado em dois diferentes conjuntos de dados: uma simulação OPNET e dados de avaliação DARPA98. Os resultados indicaram que o sistema pode detectar com segurança ataques *DoS* com taxas de falso positivo e negativo muito baixas.

7.2 Seleção de Conjunto de Características

Experiências para selecionar subconjuntos importantes para o sistema foram elaboradas e relatadas no Capitulo 4. Os resultados experimentais indicaram que através da seleção adequada de subconjuntos de características, é possível reduzir o número de características monitoradas de 45 para 8.

7.3 Pesquisa para Aperfeiçoamento de Desempenho

Várias atividades de pesquisa para aperfeiçoar o desempenho do sistema foram executadas. Os resultados e conclusões foram relatados no Capítulo 5, que incluem:

1. **Estudo de esquemas PDF:** Foram analisados três algoritmos, utilizados para construir e representar funções de densidade de probabilidade em sistemas de computadores. Estes algoritmos foram apresentados na Seção 5.1. Os resultados mostraram que a partição em 16 divisões pode ser uma escolha adequada para complexidade computacional e o desempenho do sistema.

2. **Compressão de PDF utilizando *Wavelet*:** Quatro algoritmos diferentes de *wavelet*, *haar*, *sym2*, *coif3* e *db4*, com vários limites de compressão foram testados para comprimir PDFs geradas pelo sistema (Seção 5.2). Os resultados mostraram que todos os quatro podem comprimir as PDFs de 64 divisões para 16 e 28 coeficientes de *wavelet* (faixa de compressão 3), sem grandes queda de desempenho. Entre os quatro algoritmos de *wavelet*, o *sym2* demonstrou o melhor desempenho.

3. **Estudo de Métricas de Similaridade:** A eficácia de quatro métricas de similaridade, que são usadas para medir as distâncias entre duas PDFs foi analisada na Seção 5.3. Pela comparação e análise dos resultados destas métricas, o *AKS* (Equação 3.3) foi escolhido como a melhor métrica de similaridade.

4. **Estudo dos Classificadores de Redes Neurais:** Por serem o núcleo principal dos sistemas de detecção de intrusão, os classificadores têm um grande impacto no desempenho e na eficiência dos sistemas. Testes foram realizados e relatados na Seção 5.4 para avaliar três tipos diferentes de redes com vários números de neurônios ocultos. Os resultados indicam que as redes *BP* e *BPH* permitem resultados de classificações mais eficientes que a alternativa *Perceptron*.

7.4 Métodos de Treinamento de Classificadores em Ambiente de Produção

Dois algoritmos diferentes para treinar classificadores de detecção de intrusão em um ambiente de produção que utilizaram as informações obtidas de uma rede de testes, *graft* e *re-use*, foram estudados no Capítulo 6. Os resultados apresentados pelos dois algoritmos podem ser utilizados como entradas iniciais para o treinamento de classificadores em um novo ambiente com informações insuficientes de ataques.

7.5 Futuros Estudos

Como estudos futuros, ficam algumas idéias que podem vir a ser úteis no desenvolvimento dos métodos até agora empregados. Para investigar de modo mais abrangente o desempenho das redes neurais na detecção de intrusão, além do *Perceptron*, do *BP* e do *PBH* já analisados neste estudo, experimentos com as redes *Fuzzy ARTMAP* e *RBF* (*Radial Basis Function*) devem acrescentar mais informações relevantes para o objetivo de obter melhor desempenho e precisão. Este estudo foi baseado na orientação a pacotes para a detecção de ataques *DoS*, e outra alternativa interessante é a análise estatística orientada por sessão, visando a identificação de ataques do tipo *P&S*

Referências Bibliográficas

[Agrawal e Srikant 1994] AGRAWAL, R. and SRIKANT, R. *Fast algorithms for mining association rules*. In Proceedings of the 20th VLDB Conference, (Santiago, Chile), 1994.

[Anderson 1980] ANDERSON, J. P. *Computer security threat monitoring and surveillance*. Tech. rep., James P. Anderson Co., Fort Washington, PA, Abril 1980.

[Axelsson 1999] AXELSSON, S. *On a difficulty of intrusion detection*. In Proceedings of the 2nd Intl. Workshop on Recent Advances in Intrusion Detection (RAID'99), (http://www.raid-symposium.org/raid99/PAPERS/Axelsson.pdf), September 1999.

[Beale e Foster 2003] BEALE, J. and FOSTER, J. *Snort 2.0 - Intrusion Detection*. Syngress Publishing, Inc., 2003.

[Bertino et al. 2005] BERTINO, E., KAMRA, A., TERZI, E. and VAKALI, *A. Intrusion Detection in RBAC-administered Databases*, *acsac*, pp. 170-182, 21st Annual Computer Security Applications Conference (ACSAC'05), 2005.

[Bonifacio et al. 1998] BONIFACIO, J. and et al. *Neural networks applied in intrusion detection systems*. In Proceedings of the 1998 IEEE Internal Joint Conference on Neural Networks, pp. 205-210, May, 1998.

[Borgelt 2006] BORGELT, C. Neural *Networks and Fuzzy Systems: Decision Trees.* In http://fuzzy.cs.uni-magdeburg.de/borgelt/software.html, 2006.

[Cabrera et al. 2000] CABRERA, J.et al, *Statistical traffice modeling for network intrusion detection*. In Proceedings of 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 466-473, August 2000.

[Cordeiro et al. 2005] CORDEIRO, O. C. and et al., *Exploiting Multithreaded Programming on Cluster Architectures*, *hpcs*, pp. 90-96, 19th International Symposium on High Performance Computing Systems and Applications (HPCS'05), 2005.

[Dain Cunningham 2001] DAIN, O. M. and CUNNINGHAM, R. K. *Fusing a heterogeneous alert stream into scenarios*. In Proceedings of the Eighth ACM Conference on Computer and Communications Security, pp. 1-13, 2001.

[Dasgupta e Gonzalez 2002] DASGUPTA, D. and GONZALEZ, F. *An immunity-based technique to characterize intrusions in computer networks*. IEEE Transactions on Evolutionary Computation, vol. 6, pp. 281-291, June 2002.

[Daubechies 1992] DAUBECHIES, I. *Ten Lectures on Wavelets*. Philadelphia, PA: SIAM, 1992.

[Denning 1987] DENNING, D. E. An intrusion detection model, IEEE Transactions on Software Engineering (SE-13), pp. 222-232, February 1987.

[Dhawan 2003] DHAWAN, A. *Medical Image Analysis*. John Wiley and Sons, Inc., 2003.

[Dillon e Manikopoulos 1991] DILLON, R. and MANIKOPOULOS, C. *Neural net nonlinear predication for speech data*. IEEE Eletronics Letters, vol. 27, pp. 824-826, May 1991.

[Donoho e Ducan 2004] DONOHO, D.and DUCAN, M. *WAVELAB 802 for Matlab 5.x*, In http://www.mathworks.com, 2004.

[Forrest et al. 2003] FORREST et al. *Technical Report: Computer Immune Systems*. In http://www.cs.unm.edu/immsec, 2003.

[Frawley et al. 1992] FRAWLEY, W. et al. *Knowledge discovery in databases: an overview*. AI Magazine, vol. 13, pp. 57-70, 1992.

[Furey et al. 2000] FUREY, T. et al. *Support vector machine classification and validation of cancer tissue samples using microarray expression data*. Bioinformatics, no. 16, pp. 906-914, 2000.

[Gao et al. 2002] GAO, B. et al. *HMMS (hidden markov models) based on anomaly intrusion detection method.* In Proceedings of the 2002 International Conference on Machine Learning and Cybernetics, (Beijing, China), pp. 381-385, 2002.

[Ghosh, Wanken e Charron 1998] GOSH, A., WANKEN, J. and CHARRON, F. *Detecting anomalous and unknown intrusions against programs*. In Proceedings of IEEE 14th Annual Computer Security Applications Conference, pp. 259-267, 1998.

[Giacinto e Roli 2002] GIACINTO, G. and ROLI, F. *Intrusion detection in computer networks by multiple classifier systems*. In Proceedings of the 16th International Conference on Pattern Recognition (ICPR2002), vol 2, pp. 390-393, 2002.

[Goolman 1999] GOLLMAN, D. Gollman. *Computer Security*. John Wiley and Son Ltd., 1st ed., 1999.A21

[Gunneti e Ruffo 1999] GUNNETI, D. and RUFFO, G. *Intrusion detection through behavioral data*. In Proceedings of Intelligent Data Analysis (IDA'99), (Amsterdam, Netherlands), August 1999.

[Guyon e Elisseeff 2003] GUYON, I. and ELISSEEFF, A. *An introduction to variable and feature selection*. Journal of Machine Learning Research (JMLR), no. 3, pp. 1157-1182, 2003.

[Halme e Bauer 1995] HALME, L. and BAUER, K. *AINT misbehaving – a taxonomy of anti-intrusion techniques*. In Proceedings of the 18th National Information System Security Conference, (Baltimore, MD, USA), pp. 163-172, 1995.

[Haykin 1994] HAYKIN, S. *Neural Network: a Comprehensive Foundation*. Macmillan College Publishing Company, 1994.

[Heady et al. 1990] HEADY, R. et al. *The architecture of a network level intrusion detection system*. Tech. rep., Computer Science Department, University of New Mexico, August 1990.

[IDEVAL 2005] IDEVAL. *Intrusion Detection Evaluation Data Set*. http://www.ll.mit.edu/IST/ideval, 2005.

[Inza et al. 1999] INZA, I. et al. *Feature subset selection by Bayesian networks based optimization*. Tech. rep., University of the Basque Country, Spain, 1999.

[ISS 2006] ISS. Internet Security Systems. In: http://www.iss.net, 2006.

[Javitz e Valdes 1993] JAVITZ, H. and VALDES, A. *The NIDES statistical component: description and justitication*. Tech. rep., SRI International, March 1993.

[John et al. 1994] JOHN, G. et al. *Irrelevant features and the subset selection problem*. In Proceedings of the Eleventh International Conference on Machine Learning, pp. 121-129, 1994.

[Jonsson 1998] JONSSON, E. *An integrated framework for security and dependability*. In Proceedings of the New Security Paradigms Workshop, (Charlottesville, VA, USA), pp. 22-29, September 1998.

[Jung et al. 2004] JUNG, J. et al. *Fast portscan detection using sequential hypothesis testing*. in Proceedings of IEEE Symposium on Security and Privacy, http://www.sds.lcs.mit.edu/papers/portscan-oakland04.html, 2004.

[Kantarcioglu e Clifton 2004] KANTARCIOGLU, M. and CLIFTON, C. *Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data*, IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 9, pp. 1026-1037, Sept., 2004.

[KDD 1999] KDD. *KDD cup 1999 data: Intrusion Detector Learning*. In http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[Kim, Jo e Suh 2006] KIM, Y., JO, J. and SUH, K. *Baseline Profile Stability for Network Anomaly Detection*, *itng*, pp. 720-725, Third International Conference on Information Technology: New Generations (ITNG'06), 2006.

[Kohavi e John 1997] KOHAVI, R. and JOHN, G. *Wrappers for feature selection*. Artificial Intelligence, pp. 273-324, December 1997.

[Leland et al. 1994] LELAND,W. et al. *On the self-similar nature of ethernet traffic*. IEEE/ACM Transactions on Networking, pp. 1-15, February 1994.

[Lichodzijewski et al. 2002] P. Lichodzijewski, A. and et al. *Host-based intrusion detection using self-organizing maps*, in Proceedings of the 2002 International Joint Conference on Neural Networks, (Honolulu, HI), pp. 1714-1719, May 2002.

[Lippman et al. 1999] LIPPMAN, R. P. et al. *Results of the darpa 1998 offline intrusion detection evaluation*. In Proceedings of the RAID 1999 Conference, (West Lafayette, Indiana), September 1999.

[Lippman et al. 2000] LIPPMAN, R. P. et al. *The 1999 darpa off-line intrusion detection evaluation*. Computer Networks, vol. 34, pp. 579-595, October 2000.

[Liu e Motoda 1998] LIU, H. and MOTODA, G. *Feature Selection for Knowledge Data Mining*. Kluwer Academic Publishers, 1998.

[Lunt 1998] LUNT, T. and et al. *IDES: an enhanced prototype, a real-time intrusion detection system*. Tech. rep., SRI International, October 1998.

[Mallat 1999] MALLAT, S. *A Wavelet Tour of Signal Processing*. Academic Press, 2 ed., 1999.

[Martinez 2002] MARTINEZ, R. *A Pattern Recognition Feature Optimization using the Visual Empirical Region of Influence Algorithm*. http://www.sandia.gov/imrl/XVisionScience/XVERIFeatureOp.htm, 2002.

[Miller 1990] MILLER, A. *Subset Selection in Regression*. Washington, DC. Chapman and Hall, 1990.

[NAI 2006] Network Associates International, McAfee AVERT Labs Threat Library: MyDoom. In http://vil.nai.com/vil/content/v_100983.htm, January 2006.

[NFR 2006] NFR. Security. In http://www.nfr.com, 2006.

[Ogden 1997] OGDEN, R. *Essential Wavelets for Statistical Applications and Data Analysis*. Birhauser: Boston, 1997.

[Paxson 1999] PAXSON, V. *Bro: a system for detection network intruders in real-time*, Computer Networks, no. 31, pp. 2435-2563, 1999.

[Portnoy et al. 2001] PORTONOY et al. *Intrusion detection with unlabeled data using clustering*. In Proceedings of ACM Workshop on Data Mining for Security Applications, pp. 1-14, November 2001.

[Proctor 1994] PROCTOR, P. *Audit reduction and misuse detection in heterogeneous environments: Framework and applications*. In Proceedings of the 10th Annual Computer Security Applications Conference, pp 117-125, December 1994.

[Quinlan 1993] QUINLAN, J. *C4.5 Programs for Machine Learning*. Mogan Kaufmann Publisher, Inc., 1993.

[Robertson et al. 2003] ROBERTSON, S. et al. *Surveillance detection in high bandwidth environments*, in Proceedings of the 2003 DARPA DISCEX III Conference, pp. 130-138, April 2003.

[Shah et al.2003] SHAH, H. et al. *Fuzzy clustering for intrusion detection*. In The Proceedings of the 12th IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1274-1278, May 2003.

[SNORT 2006] SNORT. *Snort*. http://www.snort.org, 2006.

[Stainford et al. 2002] STANIFORD, S. and et al. *Practical automated detection of stealthy portscans*. ‚Journal of Computer Security, vol. 10, pp. 105-126, 2002.

[Staniford et al. 1996] STANIFORD, S. et al. *GRIDS – a graphic based intrusion detection system for large networks*. In Proceedings of 19th National Information Systems Security Conference, pp. 361-370, October 1996.

[Stolfo e Mok 1999] STOLFO,W. Lee and MOK, K. *A data mining framework for building intrusion detection models*. In Proceedings of 1999 IEEE Symposium of Security and Privacy, pp. 120-132, 1999.

[Sung e Mukkamala 2003] SUNG, A. and MUKKAMALA, S. *Identifying important features for intrusion detection using support vector machines and neural networks*. In Proceedings of the 2003 Symposium on Applications and Internet (SAINT'03), pp. 209-216, 2003.

[Tan e Maxion 2005] TAN, K. M. C. ane MAXION, A. R. *The Effects of Algorithmic Diversity on Anomaly Detector Performance*, *dsn*, pp. 216-225, 2005 International Conference on Dependable Systems and Networks (DSN'05), 2005.

[Tjaden, Welch et al. 2000]. TJADEN, L., WELCH, L., and et al. *INBOUND: The integrated network-based Ohio university network detective service*. In Proceedings of 4th World Multiconference on Systemics, Cybernetics and Informatics (SCI2000), (Orlando, Florida), July, 2000.

[Valdes e Anderson 1995] VALDES, A. and ANDERSON, D. *Statistical methods for computer usage anomaly detection using NIDES*. Tech. rep., SRI International, January 1995.

[Valdes e Skinner 2001] VALDES A. and SKINNER, K. *Probabilistic alert correlation*. In Recent Advances in Intrusion Detection (RAID 2001), no. 2212 in Lecture Notes in Computer Science, pp. 54-68, Springer-Verlag, 2001.

[Vigna e Kemmerer 1998] VIGNA, G. and KEMMERER, R. A. *Netstat: a network-based intrusion detection approach*. In Proceedings of the 14th Annual Computer Security Applications Conference, pp. 25-34, 1998.

[Vilegas et al. 2006] VILEGAS, O. et al. *Apple Classification System with EZW and Daubechies D4 Lossy Image Compression*, *conielecomp*, p. 50, 16th International Conference on Electronics, Communications and Computers (CONIELECOMP'06), 2006

[Wang Daniels 2005] WANG., W. and DANIELS, T., *Building Evidence Graphs for Network Forensics Analysis*, *acsac*, pp. 254-266, 21st Annual Computer Security Applications Conference (ACSAC'05), 2005.

[Weber 1998] WEBER, D. *A taxonomy of computer intrusions, Master's thesis*. Massachusetts Institute of Technology, Cambridge, MA, 1998.

[Ye et al. 2000] YE, N. et al. *Decision trees for signature recognition and state classification*. In Proceedings of IEEE SMC Information Assurance Security Workshop, (West Point, NY), pp. 189-194, June 2000.

[Zhang e Manikopoulos 2001a] ZHANG, Z. and MANIKOPOULOS, C. *Neural networks in statistical anomaly intrusion detection*. Neural Network World, International Jornal on Non-Standard Computing and Artificial Intelligence, vol. 11, no 3, pp. 305-316, 2001.

[Zhang et al. 2002a] ZHANG, Z. et al. *Experimental comparisons of binning schemes in representing network intrusion detection data*. In The 36th Conference of Information Sciences and Systems (CISS2002), March 2002.

[Zhang et al. 2002b] ZHANG, Z. et al. *Methods of Classifier training for anomaly network intrusion detection in a deployed network using test network information*. In Proceedings of the 3rd Annual IEEE Systems, Mans, Cybernetics Information Assurance Workshop (IAW 2002), June 2002.

Apêndice I Características Estatísticas Monitoradas

Este apêndice fornece descrições detalhadas das características estatísticas monitoradas neste estudo, que foram categorizadas baseando-se nos protocolos de tráfego de rede.

A.1 Características IP

1. **IP Packet Lenght** (ip-pkt-len): Mede as médias e as distribuições de comprimentos de pacotes IP dentro de uma janela de tempo.

2. **IP Packet Rate** (ip-pkt-rate): Mede as médias e as distribuições das taxas de pacotes de todos os pacotes IP observados dentro de uma janela de tempo.

3. **IP Byte Rate** (ip-byt-rate): Mede as médias e as distribuições das taxas de byte de todos os pacotes IP observados dentro de uma janela de tempo.

4. **IP Fragment Rate** (ip-frag-rate): Mede as médias e as distribuições das taxas de fragmentos IP que ocorrem dentro de uma janela de tempo.

5. **IP Defragmentation Error Rate** (ip-defrag-error): Mede as médias e as distribuições de taxas de erros de desfragmentação IP que ocorrem dentro de uma janela de tempo.

6. **IP Checksum Error Rate** (ip-csum-error): Mede as médias e as distribuições de taxas de error de checksum IP que ocorrem dentro de uma janela de tempo.

A.2 Características TCP

1. **TCP Invalid Packet Rate** (tcp-pkt-invalid): Mede as médias e as distribuições das taxas de pacotes TCP com combinações inválidas de flags de controle TCP.

2. **TCP Packet Lenght** (tcp-pkt-len): Mede as médias e distribuições dos comprimentos de pacotes IP dentro de uma janela de tempo.

3. **TCP Packet Rate** (tcp-pkt-rate): Mede as médias e distribuições de taxas de pacotes TCP dentro de uma janela de tempo.

4. **TCP SYN Packet Rate** (tcp-syn-pkt-rate): Mede as médias e as distribuições de taxas de pacotes de controle TCP com flag set SYN dentro de uma janela de tempo.

5. **TCP FIN Packet Rate** (tcp-fin-pkt-rate): Mede as médias e as distribuições de taxas de pacotes de controle TCP com flag set FIN dentro de uma janela de tempo.

6. **TCP RST Packet Rate** (tcp-rst-pkt-rate): Mede as médias e as distribuições de taxas de pacotes de controle TCP com flag set RST dentro de uma janela de tempo.

7. **TCP Connection Open Rate** (tcp-com-new-opened): Mede as médias e as distribuições de taxas de connection open TCP dentro de uma janela de tempo.

8. **TCP Connection Close Rate** (tcp-com-new-closed): Mede as médias e as distribuições de taxas de close connection close TCP dentro de uma janela de tempo.

9. **TCP Connection Abort Rate** (tcp-com-new-aborted): Mede as médias e as distribuições de taxas de connection abort TCP (conexões fechadas por RESET ou TIMEOUT diferentes do normal three-way hand shaking) dentro de uma janela de tempo.

10. **TCP Connections from Different Source Address** (tcp-com-diff-srd): Mede as distribuições de conexões TCP de endereços fonte IP diferentes dentro de uma janela de tempo.

11. **TCP Connections from Different Destination Source Address** (tcp-com-diff-dst): Mede as distribuições de conexões TCP de endereços destino IP diferentes dentro de uma janela de tempo.

12. **TCP Connection Anomalous Entropy** (tcp-con-anomalous-entropy): Mede as médias e as distribuições de entropias anômalas de todas as conexões TCP dentro de uma janela de tempo. Esta característica foi inicialmente proposta in Staniford [37]. A equação para calcular a entropia anônima da conexão é dada na equação 7.1.

13. **TCP Connection Half Opened Ratio** (tcp-com-half-opened-ratio): Mede as médias e as distribuições de ratio entre conexões TCP half-opened e todas as conexões TCP dentro de uma janela de tempo.

14. **TCP Connection Duration** (tcp-com-duration): Mede as médias e as distribuições de conexões TCP durante uma janela de tempo.

A.3 Características UDP

1. **UDP Packet Lengh** (udp-pkt-len): Mede as médias e as distribuições de pacotes UPD dentro de uma janela de tempo.

2. **UDP Packet Rate** (udp-pkt-rate): Mede as médias e as distribuições de pacotes UDP dentro de uma janela de tempo.

3. **UDP Byte Rate** (udp-byt-rate): Mede as médias e distribuições de pacotes UDP dentro de uma janela de tempo.

4. **UDP Packets from Different Sources** (udp-diff-src): Mede as distribuições de pacotes UDP de endereços IP diferentes.

5. **UDP Packet to Different Destination** (udp-diff-dst): Mede as distribuições de pacotes UDP destinados a endereços IP diferentes.

A.4 Características ICMP

1. **ICMP Packet Lenght** (icmp-pkt-len): Mede as médias e as distribuições de pacotes ICMP dentro de uma janela de tempo.

2. **ICMP Packet Rate** (icmp-pkt-rate): Mede as médias e as distribuições de pacotes ICMP dentro de uma janela de tempo.

3. **ICMP Packets from Different Sources** (icmp-diff-src): Mede as distribuições de pacotes ICMP originados de diferentes endereços IP dentro de uma janela de tempo.

4. **ICMP Packets from Different Destinations** (icmp-diff-dst): Mede as distribuições de pacotes ICMP destinados a diferentes endereços IP dentro de uma janela de tempo.

5. **ICMP Anomalous Echo Replies** (icmp-anomalous-echo-reply): Mede as médias e as distribuições de echo replies ICMP anômalos, que são pacotes echo reply sem prévios pacotes de echo request, dentro de uma janela de tempo.

6. **ICMP DUR Packet Rate** (icmp-dur-pkt-rate): Mede as médias de distribuições de pacotes ICMP DUR (destination-Unreachable) dentro de uma janela de tempo.

Apêndice II Simulação de Detecção de Intrusão com OPNET

Resumo: A detecção de intrusão e a segurança em redes de computadores são preocupações constantes no domínio da Tecnologia da Informação. Os invasores têm utilizado diversas técnicas com o objetivo de comprometer as operações normais dos sistemas em produção. A maioria dos recursos de defesa atualmente utilizados pelas empresas, como firewalls e sistemas de detecção de intrusão, é eficiente em detectar tentativas de intrusão conhecidas, analisando suas assinaturas de ataques, não obtendo os mesmos resultados para novos ataques. O tráfego gerado em simulações OPNET permite analisar e desenvolver novas estratégias de filtragem de pacotes e detecção de intrusão. Este projeto tem o objetivo de simular o tráfego de intrusão, gerando arquivos de dados baseados em arquivos TCPDUMP reais, que contenham pacotes de intrusão, além de avaliar a eficiência das simulações de desempenho em redes de computadores para a elaboração de algoritmos de detecção de intrusão. As estratégias adotadas permitem detectar tendências de invasões sem informações prévias sobre assinaturas de ataques.

Palavras-chaves: detecção de intrusão em redes, simulação de intrusão, ataques DoS, TCPDUMP.

I. INTRODUÇÃO

As invasões em sistemas de computadores têm ocorrido com freqüência. Os analistas de segurança têm como uma de suas principais preocupações a detecção de intrusão, implantando, por exemplo, sistemas do tipo IDS (*Intrusion Detection System*), avaliando o comprometimento entre fatores de riscos e custos. A disponibilidade de acesso aos serviços de rede e à Internet dentro das empresas é cada vez mais ampla, trazendo inúmeros benefícios aos negócios, clientes, parceiros e funcionários, mas com comprometimento da segurança e aumento de vulnerabilidades. Este projeto tem o objetivo de realizar estudos sobre intrusões e seus efeitos em um ambiente simulado de redes de computadores, apresentando resultados experimentais de desempenho e eficiência da simulação de redes sobre ataques do tipo DoS (Denial of Service), utilizando o *software* OPNET.

II. A DETECÇÃO DE INTRUSÃO EM REDES

Os sistemas de detecção de intrusão são utilizados para auxiliar sistemas de computadores a se prepararem contra ataques e reagirem aos mesmos. Este objetivo é atingido com a coleta de

informações de varias fontes nos sistemas de uma rede, com a análise destas informações na procura por sintomas de problemas de segurança [1,2]. Em alguns casos, os sistemas de detecção de intrusão permitem ao usuário especificar respostas em tempo real para as violações. A seguir, algumas das funções que um sistema de detecção de intrusão deve fornecer:

- Monitoração e análise das atividades do sistema e dos usuários.
- Auditoria das configurações do sistema e vulnerabilidades.
- Reconhecimento de padrões de atividades que indiquem ataques conhecidos.
- Análise estatística de padrões de atividades anormais.
- Gerenciamento de dados de auditoria do sistema operacional, com reconhecimento de atividades dos usuários que indiquem violações de políticas.

Alguns sistemas possuem outras características adicionais, que podem incluir:

- Instalação automática de atualizações ou correções dos fornecedores.
- Instalação e operação de servidores com a função de enganar intrusos com o objetivo de obter informações sobre os atacantes.

A combinação destas características permite aos administradores de sistemas utilizarem com mais facilidade as ferramentas de monitoração, auditoria e avaliação de seus sistemas e redes. Estas auditorias e avaliações contínuas são práticas necessárias no gerenciamento da segurança. Muitas medidas de segurança empregadas não são 100% seguras. Os sistemas de detecção de intrusão podem auxiliar analistas de segurança a identificar se um ataque está sendo realizado, em tempo real ou imediatamente após o ataque. Um dos desafios dos administradores de segurança é encontrar a melhor maneira de identificar intrusões de rede e como avaliar a eficácia de um sistema de detecção de intrusão.

III. A SIMULAÇÃO DA INTRUSÃO EM REDE

Os algoritmos de detecção de intrusão são normalmente eficientes na detecção de assinaturas conhecidas de ataques, mas não na detecção de novos ataques. Estudar e testar um novo algoritmo de detecção de intrusão contra uma variedade de atividades intrusivas (reais ou simuladas) sob um tráfego de rede real é um problema interessante e complexo. Este estudo pode ser executado em ambiente real ou simulado [3]. A seguir, um breve resumo destes dois

métodos de simulação de intrusão, suas vantagens e desvantagens, e a descrição do método proposto neste projeto.

Simulação em ambientes reais

Neste método muitos usuários reais produzem tráfego significativo utilizando uma variedade de serviços de rede, como correio, *telnet*, etc. Este tráfego é coletado e gravado, e atividades intrusivas podem ser emuladas executando scripts do tipo *exploit*. A vantagem deste método é que o tráfego é suficientemente real, o que elimina a necessidade de analisar e simular os padrões de atividades normais dos usuários. Entretanto, os seguintes inconvenientes são observados:

- O ambiente de testes pode ser exposto a ataques da Internet, além dos ataques simulados.
- A imprecisão dos resultados pode aumentar se o tráfego possuir dados intrusivos inesperados originados de fontes externas.
- As operações normais dos sistemas podem ser interrompidas pelos ataques simulados.

Simulação em ambientes experimentais

Muitos pesquisadores executam testes e estudos em ambientes experimentais ou simulados, devido ao alto risco da execução de testes em um ambiente real [3]. Neste método o tráfego real pode ser gerado de três modos distintos:

- Utilizar humanos para manualmente executar toda e qualquer operação que possa ocorrer em um ambiente real;
- Utilizar scripts de simulação para gerar dados de acordo com as distribuições estatísticas do tráfego em um ambiente real;
- Coletar dados de um ambiente real e replicar em um ambiente simulado.

As desvantagens de estudar intrusão em um ambiente simulado incluem:

- A sobrecarga de manualmente produzir tráfego envolvendo muitos usuários pode ser elevada.
- O comportamento de usuários reais é difícil de modelar, e não existem distribuições estatísticas padronizadas de tráfego.

Simulação de intrusão proposta

O *software* OPNET é utilizado para construir o ambiente de rede simulado. As fontes de tráfego de rede utilizadas são os arquivos TCPDUMP do MIT/Lincoln Lab, que possuem tráfego de intrusão simulando vários ataques de rede [4]. Como alternativa podem ser utilizadas ferramentas como NMAP [5] para gerar ataques enquanto executando um *sniffer* de rede como o Ethereal [6], para capturar o tráfego de rede, com as atividades de ataque e coleta do *sniffer* ocorrendo em um ambiente controlado de laboratório. Os arquivos Ethereal que incluem os dados de ataque e dados normais de usuários são utilizados neste projeto de simulação de intrusão. Tanto para os arquivos TCPDUMP como para os arquivos Ethereal é utilizada uma ferramenta de pré-processamento para extrair as informações de tráfego que são necessárias para a simulação OPNET. Esta ferramenta de pré-processamento analisa os cabeçalhos, informações de *flag*s e distribuição de tempo (o pacote total não é incluído na simulação, embora isto seja possível). O *software* OPNET possui um módulo ACE (*Application Characterization Environment*) que é utilizado na importação dos dados dos pacotes para a simulação, suportando formatos de pacotes de várias fontes, incluindo arquivos TCPDUMP [7]. Este método foi escolhido por sua flexibilidade na seleção de partes dos pacotes de dados e dividindo grandes arquivos de dados em partes mais gerenciáveis para a simulação.

IV. MÉTODO DE SIMULAÇÃO COM OPNET

O *software* OPNET foi escolhido para esta proposta por apresentar inúmeros benefícios. O OPNET possui uma interface gráfica para o desenho da topologia, que permite simulação de redes, além de possuir um módulo de exibição do desempenho da coleção de dados. Outra vantagem da utilização do OPNET é a consagrada confiabilidade na validação dos resultados produzidos. O OPNET permite análises de medidas de desempenho e a eficácia nas técnicas de detecção de intrusão. Um dos objetivos deste projeto é estudar técnicas que possam acelerar as simulações com OPNET para grandes arquivos de dados suspeitos de ataques de intrusão.

A. Geração dos dados de intrusão

Inicialmente é utilizado o ataque Dosnuke como um exemplo para ilustrar o processo de simulação de intrusão. Dosnuke é um tipo de ataque de negação de serviço que envia dados *Out-of-Band* (MSG_OOB) para a porta 139 (NetBIOS) de um sistema Windows, fazendo com que o PC vítima trave em tela azul. O ataque pode ser detectado através de pesquisa nos

dados gerados pelo *sniffer*, localizando-se um *handshake* NetBIOS seguido de pacotes NetBIOS com um *flag* do tipo "*urg*" ativo.

B. Formatação dos pacotes de dados

Foi definido o formato de pacotes na simulação OPNET. O pacote corresponde ao cabeçalho IP, que inclui os endereços IP, números de portas, *flags*, e outros campos.

C. Pré-processamento dos arquivos-fonte de tráfego

Antes de construir o modelo de rede simulado utilizando-se do OPNET, foi necessário pré-processar os arquivos TCPDUMP ou Ethereal) para extrair informações pertinentes, que incluem:

- O pacote "*inter-arrival times*", que é salvo como uma lista de valores de tipo duplo.
- A duração de tempo, que é a diferença de tempo entre o primeiro pacote e o último pacote da fonte de tráfego.
- Uma lista de endereços IP distintos na fonte de tráfego.

D. Construção do modelo de rede

A Figura 1 mostra o modelo OPNET para a simulação do ataque Dosnuke. São 10 nós virtuais de PCs agrupados em duas colunas: PC0 a PC4 no lado esquerdo e PC5 a PC9 no lado direito. O nó superior na coluna central é o "gerador", quer prepara os pacotes extraídos da fonte de tráfego. Uma vez que o pacote esteja pronto, ele é enviado ao nó PC de destino através do hub (nó inferior da coluna central). Não existe atraso entre o gerador e os nós dos PCs finais, portando o fluxo de tráfego é consistente com a fonte de tráfego capturada.

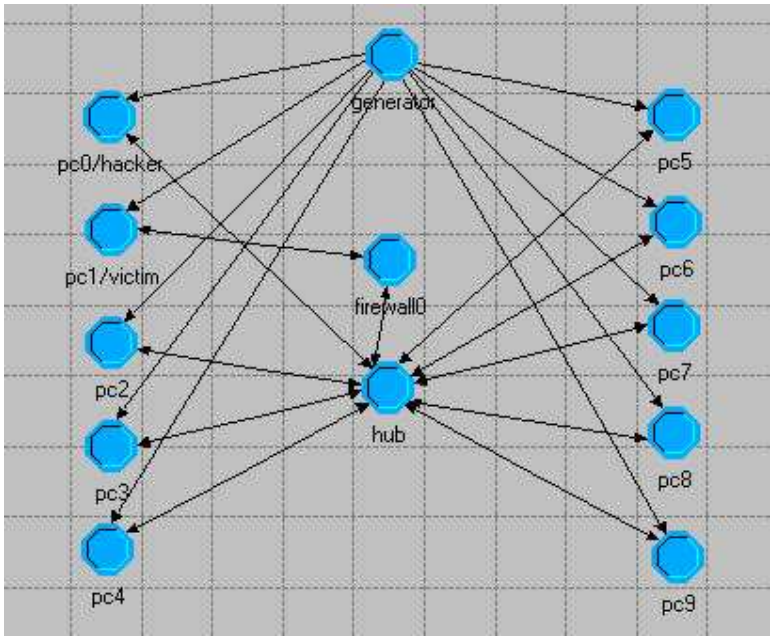

Figura 1. - Modelo de rede simulando uma intrusão Dosnuke.

O número de PCs virtuais é conseqüência do pré processamento do arquivo-fonte de tráfego. Como existem 10 endereços IP distintos na fonte, o modelo utiliza 10 nós PCs conectados uns aos outros através de um hub. O nó 0 (o nó superior na coluna do lado direito) é o "hacker", e o nó 1 (abaixo do nó 0 na figura) é a "vítima" do ataque Dosnuke. Existe um nó "firewall" entre a vítima e o hub que é utilizado para capturar pacotes de dados suspeitos enviados e recebidos pela vítima usando a assinatura de um ataque Dosnuke. O domínio-nó do "gerador" é exibido na Figura 2. Existe um módulo gerador (pk_generator) configurado para utilizar um arquivo script de "inter-arrival times" para gerar os pacotes.

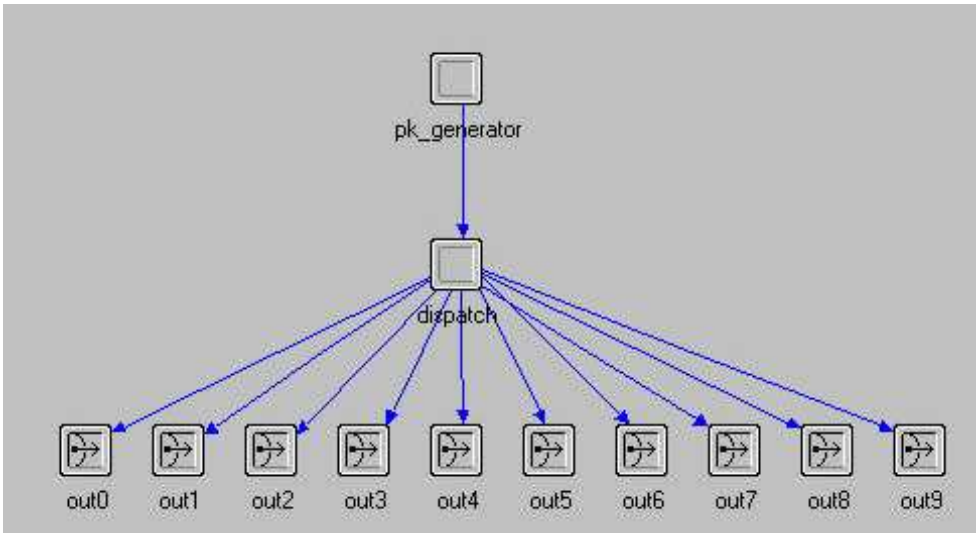

Figura 2. - Estrutura nó do gerador de pacotes.

O arquivo script é o resultado do pré-processamento da fonte de tráfego. A figura 3 mostra o painel de atributos do pk_generator.

Attribute	Value
name	pk_generator
process model	simple_source_1
icon name	processor
Packet Format	_virtual_pk
Packet Interarrival Time	scripted (TimeStampSequence)
Packet Size	constant (1024)
Start Time	0.0
Stop Time	Infinity

Figura 3. - Painel de atributos do "pk_generator".

Dentro do módulo "dispatch"do nó gerador na Figura 2, o arquivo-fonte de tráfego é analisado e o próximo pacote de dados é extraído. Sempre que um pacote chega do "*pk_generator*", seus campos são preenchidos de acordo com os valores correspondentes do pacote de dados do trafego fonte, como destino, flags, etc. Então o pacote é enviado para o nó PC correspondente ao endereço IP fonte. Assim, o *packet arrival time* e seu conteúdo irão combinar com a informação como na fonte de tráfego original. A Figura 4 exibe o domínio do processo para todos os PC virtuais, que suporta o fluxo de pacotes de entrada e saída.

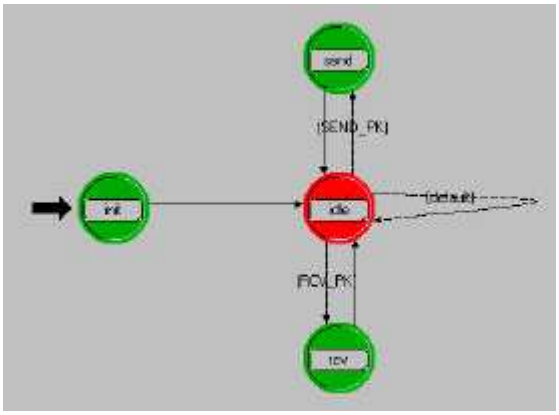

Figura 4. - Estrutura do PC virtual no domínio do processo.

Um firewall também é configurado entre o hub e a vitima do ataque dosnuke. O firewall usa uma simples detecção baseada em assinaturas que procura por pacotes enviados para a porta 139 (NetBIOS) do PC vítima, com o flag "*urg*" configurado no cabeçalho do pacote. A ferramenta de pré-processamento pode ser reutilizada para simular outros tipos de ataques de

intrusão. Para demonstrar, foi também simulado o ataque *ProcessTable DoS* utilizando-se os arquivos TCPDUMP do MIT/Lincoln Lab. Foi necessário configurar uma rede no OPNET com 20 nós PC porque existem 20 endereços IP distintos envolvidos na fonte de tráfego. A lógica de detecção do nó do firewall também foi modificada utilizando-se uma nova assinatura de ataque, e adicionando-se as medidas estatísticas correspondentes à simulação OPNET.

V. Análise dos Resultados da Simulação

A. O ataque Dosnuke

A fonte de dados do tráfego para o ataque Dosnuke foi obtida através dos arquivos TCPDUMP do MIT/Lincoln Lab (1999/week5/Monday). Este conjunto de dados inclui os 5 minutos iniciais de dados, e apenas um tipo de ataque. Para as simulações, no pré-processamento dos arquivos-fonte, foram extraídos menos de 3 minutos de dados, contendo um total de 367 pacotes TCP. Existiam 10 pacotes capturados pelo nó firewall devido ao ataque Dosnuke, 9 dos quais foram enviados do nó atacante para a vítima e um enviado da vítima, de volta para o atacante. Foram configuradas diversas medições estatísticas no OPNET para estudar o desempenho da simulação da intrusão. Por exemplo, a Figura 5 exibe a distribuição de endereços IP dos pacotes de dados durante toda a simulação, onde os endereços IP correspondem aos nós de PCs de 0 a 9 no eixo y, onde ficam claros os padrões de acessos consecutivos aos mesmos endereços IP durante vários intervalos curtos da simulação, embora estes acessos sejam irrelevantes para o ataque *Dosnuke*.

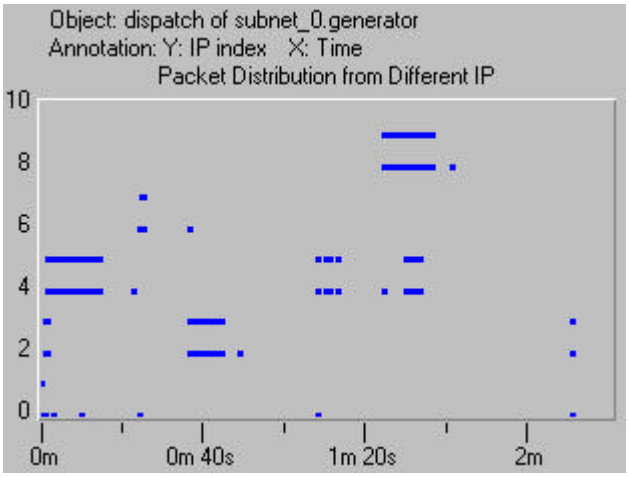

Figura 5. - Distribuição de endereços IP dos pacotes de dados.

A figura 6 exibe as taxas de pacotes de dados capturados pelo *firewall*. As ocorrências dos pacotes e os tempos de suas chegadas são claramente visíveis na figura – existe pouco tráfego rápido de chegada no início, seguido por mais 3 posteriormente. Esta figura demonstra claramente as ocorrências do ataque *Dosnuke* e sua captura pelo *firewall*.

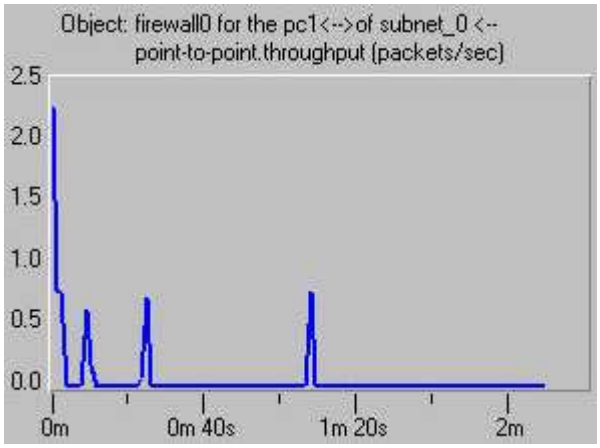

Figura 6. - Tráfego de chegada no firewall.

Foram coletadas informações estatísticas de todo o tráfego de rede, exibidas na Figura 7, embora esta medida de desempenho também aparente ser irrelevante para o ataque *Dosnuke*.

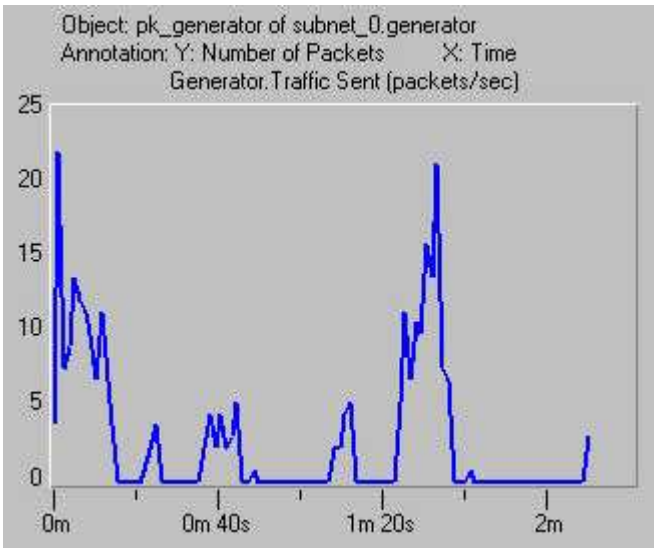

Figura 7. - O tráfego total de rede durante a simulação.

B. O ataque *ProcessTable*

O ataque *ProcessTable DoS* tem como objetivo preencher a tabela de processos de um sistema operacional, comprometendo-o até que o ataque termine ou que o administrador do

sistema derrube os processos atacantes [8]. Um ataque do tipo *ProcessTable* pode ser detectado pelo registro de um grande numero de conexões para uma porta em particular da vitima, durante um curto período de tempo. Para a simulação foi utilizado o arquivo TCPDUMP do MIT/Lincoln Lab que contém os pacotes *ProcessTable*, sendo extraídas as informações pertinentes com a ferramenta de pré-processamento, para então configurar a simulação no OPNET. Existe pouco menos de 2 minutos de dados com um total de 5526 pacotes de dados. Foram coletadas duas medições estatísticas no nó do *firewall* tentando detectar e identificar o ataque *ProcessTable*. A Figura 8 exibe o número de conexões de portas distintas para o PC vítima durante a simulação. Pode ser visto claramente que existem 3 saltos no gráfico, indicando rápido crescimento das conexões de portas durante 3 intervalos de tempo distintos.

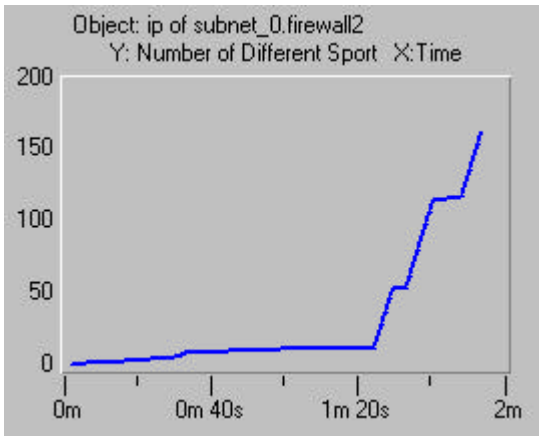

Figura 8. - Número de conexões de portas distintas para a vítima.

O ataque *ProcessTable* também pode ser direcionado para uma porta especifica da vítima. A Figura 9 exibe o tráfego de rede direcionado para a porta 25 da vitima durante a simulação. O gráfico exibe dois picos: o primeiro ocorre próximo da marca de um minuto; o segundo começa após 1 minuto e 20 segundos e permanece até o fim da simulação. Portanto, as duas figuras 8 e 9 demonstram claramente os pacotes de dados que são suspeitos de ataques *ProcessTable* (ou equivalentes).

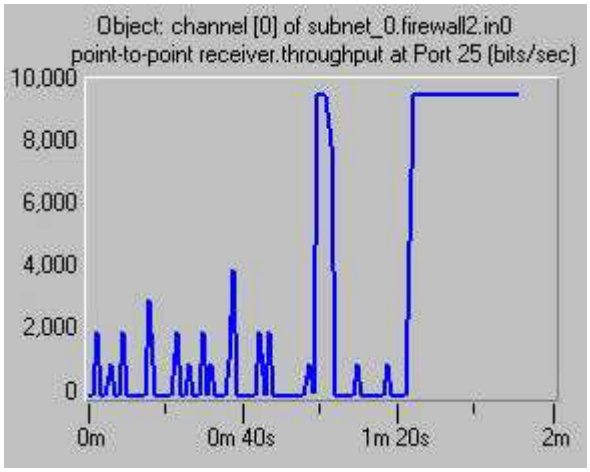

Figura 9. - Tráfego de dados para a porta 25 do PC vítima.

C. Avaliação da eficiência da simulação

Outro objetivo da simulação de intrusão com OPNET é estudar a eficiência da simulação, isto é, como aumentar a velocidade da simulação e da detecção de intrusão de tráfego de intrusão de grandes arquivos de dados. Primeiramente foi utilizado o arquivo de dados do ataque Dosnuke e foram simulados pacotes de dados e detecção de intrusão para diferentes durações de tempo. Todas as simulações foram executadas em um PC Pentium 4 , com 1.5 GHz de CPU e 256 MB de RAM. A Figura 10 exibe o tempo de simulação OPNET executando os arquivos de dados, com durações de 30 segundos até 131 segundos, em incrementos de 30 segundos. Uma vez que existem apenas umas poucas centenas de pacotes de dados (367 exatamente), todas as simulações são completamente executadas em um segundo.

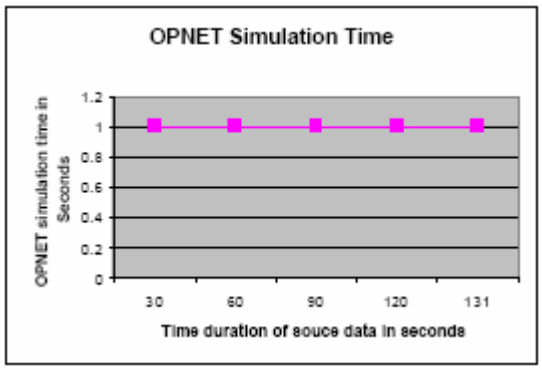

Figura 10. - Tempo de simulação OPNET para o ataque Dosnuke.

Executou-se também simulações do arquivo do ataque *ProcessTable*, com diferentes durações de tempo para medir a eficiência da simulação. A Figura 11 exibe o tempo de simulação OPNET executando os arquivos de dados, em incrementos de 30 segundos. Existe um total de

5526 pacotes de dados no arquivo inteiro (114 segundos). Observou-se que o tempo de simulação aumenta aproximadamente de modo linear à medida que a duração de tempo do arquivo fonte aumenta. Portanto, a eficiência da simulação pode se tornar um fator significativo quando se tenta detectar rapidamente intrusões que envolvam grandes arquivos de dados.

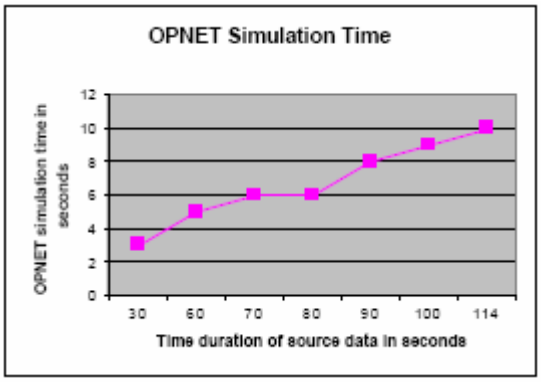

Figura 11. - Tempo de simulação OPNET para o ataque ProcessTable.

VI. CONCLUSÕES E TRABALHOS FUTUROS

Neste texto são relatados resultados experimentais da simulação de intrusão em rede utilizando-se dados TCPDUMP previamente capturados como fontes de dados. É demonstrado o uso de ferramentas de pré-processamento para facilitar a simulação de intrusão utilizando-se do software OPNET. São também demonstradas várias aplicações de simulação de intrusão utilizando-se o OPNET:

- Detecção de intrusões exibindo e identificando padrões de pacotes de dados suspeitos, empregando varias técnicas de detecção de intrusão em um *firewall*;
- Analise de desempenho da rede e a sobrecarga dos algoritmos de detecção de intrusão;
- Avaliação e a eficácia de algoritmos IDS.

Este trabalho demonstra também o problema e o desafio de tornar as simulações mais eficientes, especialmente para grandes arquivos de dados que são comuns hoje em dia. As soluções possíveis incluem a redução dos conjuntos de dados pela extração apenas das informações pertinentes e a dividisão dos conjuntos de dados em certos critérios sem degradar a eficácia do sistema de detecção de intrusão. Como trabalho futuro, planeja-se aperfeiçoar a ferramenta de pré-processamento, para que ela possa ser aplicada aos arquivos-fonte de dados para melhorar a eficiência da simulação.

REFERÊNCIAS

[1] David J. Marchette, Computer Intrusion Detection and Network Monitoring, Springer-Verlag, 2001.

[2] Stephen Northcutt, Judy Novak, Network Intrusion Detection: An Analyst's Handbook, 2nd ed., New Riders, 2000.

[3] Tao Wan, Xue Dong Yang, "IntruDetector: A Software Platform for Testing Network Intrusion Detection Algorithms ", http://www.acsac.org/2001/papers/54.pdf .

[4] DARPA Intrusion Detection Evaluation project, at http://www.ll.mit.edu/IST/ideval/data/1999/1999_data_index.html.

[5] NMAP, at http://www.insecure.org. 2004.

[6] Gerald Combs, Ethereal – Network Protocol Analyzer, http://www.ethereal.com/

[7] OPNET online documentation 8.0.C, OPNET Technologies, Inc., Washington DC.

[8] Kristopher Kendall, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems", Master's Thesis, Massachusetts Institute of Technology, 1998.