UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

ANALISADOR DE ALARMES DE TRÁFEGO DE REDES ATRAVÉS DE *WAVELETS*

TRABALHO DE GRADUAÇÃO

Bruno Lopes Dalmazo

Santa Maria, RS, Brasil

2008

ANALISADOR DE ALARMES DE TRÁFEGO DE REDES ATRAVÉS DE *WAVELETS*

por

Bruno Lopes Dalmazo

Trabalho de Graduação apresentado ao Curso de Ciência da Computação
da Universidade Federal de Santa Maria (UFSM, RS), como requisito
parcial para a obtenção do grau de
Bacharel em Ciência da Computação

Orientador: Prof. Raul Ceretta Nunes

Co-orientador: Mestrando Tiago Perlin

Trabalho de Graduação N. 269

Santa Maria, RS, Brasil

2008

Universidade Federal de Santa Maria
Centro de Tecnologia
Curso de Ciência da Computação

A Comissão Examinadora, abaixo assinada,
aprova o Trabalho de Graduação

ANALISADOR DE ALARMES DE TRÁFEGO DE REDES ATRAVÉS DE *WAVELETS*

elaborado por
Bruno Lopes Dalmazo

como requisito parcial para obtenção do grau de
Bacharel em Ciência da Computação

COMISSÃO EXAMINADORA:

Prof. Raul Ceretta Nunes
(Presidente/Orientador)

Prof[a] Roseclea Duarte Medina (UFSM)

Prof. Antonio Marcos de Oliveira Candia (UFSM)

Santa Maria, 15 de Dezembro de 2008.

“Hakuna matata.”
— JAMBO BWANA

AGRADECIMENTOS

Em primeiro lugar, agradeço a minha família e a todos aqueles que acreditaram que era possível. Agradeço ao meu Orientador, Raul Ceretta Nunes pela amizade, apoio e ensinamento, não somente durante o trabalho final, mas também ao longo de toda a graduação. Agradeço aos meus colegas de aula por todas as explicações, ajudas, festas e momentos felizes. Agradeço aos demais professores do curso que sempre estiveram dispostos e presentes, contribuindo para minha formação acadêmica e pessoal, especialmente à professora Alice de Jesus Kozakevicius pela orientação nesta reta final. Agradeço a FAPERGS por acreditar neste trabalho e pelo apoio financeiro na forma de bolsa de iniciação científica. Agradeço aos colegas de trabalho, Sebastian Feld e Tiago Perlin, pelo companheirismo e grande contribuição, ajudando a tornar este trabalho de graduação possível.

RESUMO

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

ANALISADOR DE ALARMES DE TRÁFEGO DE REDES ATRAVÉS DE *WAVELETS*
Autor: Bruno Lopes Dalmazo
Orientador: Prof. Raul Ceretta Nunes
Co-orientador: Mestrando Tiago Perlin
Local e data da defesa: Santa Maria, 15 de Dezembro de 2008.

Nos tempos atuais, com uma grande elevação do uso dos mais diversos serviços utilizando a Internet, houve uma grande evolução nos métodos de conseguir, de forma ilegal, benefícios dos computadores em rede. Desta forma, tornaram-se necessários a pesquisa e o desenvolvimento de sistemas de barreiras ao acesso indevido. Mesmo configurando métodos para inibir a ação de usuários maliciosos, a todo instante estes acham brechas nos sistemas que permitem de alguma forma prejudicar a segurança dos dados. Pode-se dizer que segurança é considerada todo o tipo de proteção aos riscos relativos às ameaças internas ou externas que podem resultar em acessos não autorizados a alguma informação.

Este trabalho explora mecanismos para identificar possíveis ataques à disponibilidade dos sistemas, procurando minimizar o número de alarmes falsos. Para isso, foi necessário estabelecer quais ataques são abrangidos; quais as características principais de cada ataque; e como este ataque pode ser detectado. Para a detecção de um ataque, utilizou-se um detector de intrusão baseado em Séries Temporais, porém, o detector apresenta um alto nível de falsos alarmes. Para a correção dos falsos alarmes, utilizou-se um sistema de filtragem dos alarmes baseado em *wavelets*, tornando, assim, os resultados mais confiáveis.

Palavras-chave: Detecção de intrusões; gerência de redes de computadores; segurança; séries temporais; *wavelets*.

ABSTRACT

Trabalho de Graduação
Curso de Ciência da Computação
Universidade Federal de Santa Maria

ANALYSIS OF NETWORK TRAFFIC AND ALARMS WITH WAVELETS

Author: Bruno Lopes Dalmazo
Advisor: Prof. Raul Ceretta Nunes
Coadvisor: Mestrando Tiago Perlin

In current times, there is a large rise of various services using the internet. And in the same proportion there is an increase of developed methods to achieve benefits of a computer network in an illegal way. Thus, it is necessary to research and develop systems that block improper access. Even setting methods to inhibit the action of malicious users, at any instant they find new vulnerabilities in systems which allow, in some way, compromising the of data security. Security is considered to be every degree of protection from risks relating to internal or external threats that can result in unauthorized access of some information. The key point to the success of a system is to keep the probability low for inaccessible information because of not planned interruptions.

Thus, this work is basically about identifing possible attacks to the availability of systems, and afterwards generate results with a low number of false positives. For this, it was necessary to investigate which attacks are covered; what are the main characteristics of each attack; and how this attacks can be detected. To detect an attack, we used an intrusion detection system based on Time Series, however, this IDS presents a high level of false alarms. To minimize the false alarms, we have used a filtering system for the alarms based on wavelets. As result we have get an improvement on IDS reliability.

Keywords: Intrusion detection; computer networks management; security; times series; wavelets.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACK	Acknowledgment
ARP	Address Resolution Protocol
DHCP	Dynamic Host Configuration Protocol
DoS	Denial of Service
DDoS	Distributed Denial of Service
FTP	File Transfer Protocol
GPL	General Public License
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
MAC	Media Access Control
RST	Reset
SMTP	Simple Mail Transfer Protocol
SYN	Synchronization
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

SUMÁRIO

1 INTRODUÇÃO

Devido ao grande crescimento da Internet, as redes de computadores ficam mais expostas, aumentando potencialmente o risco de usos inadequados ou indevidos das mesmas. O aumento da complexidade das redes, causado pela sua expansão, dificulta a detecção de anomalias que podem interferir no seu funcionamento normal. A pesquisa na área de Detecção de Intrusão tem por objetivo a busca de métodos para tratar alguns destes problemas.

Um IDS (*Intrusion Detection System*) usa informações coletadas em uma rede para identificar possíveis usos inadequados da mesma, ele realiza uma análise desses dados com o objetivo de identificar uma invasão que esteja ocorrendo ou que já tenha acontecido. Dada uma seqüência de tráfego de rede relacionada com variáveis de dados de um intervalo fixo, pode-se gerar uma função que descreve o comportamento desta rede, esta função pode ser usada para gerar alarmes correspondentes a eventos anômalos na rede (THOTTAN; JI, 2003). Após uma detecção de invasão, o IDS deve gerar uma resposta ao evento, que pode ser uma intervenção automatizada no sistema ou gerar um alerta para intervenção humana (NORTHCUTT, 1999).

Este trabalho foi motivado pela parceria entre a Universidade Federal de Santa Maria (UFSM) e a *Fachhochschule Gelsenkirchen* (FHGe). A FHGe produziu um *software* para coleta de dados e armazenamento em contadores chamado *Internet Analysis System* - IAS (POHLMANN; PROEST, 2006). Este *software* armazena em contadores a quantidade de pacotes de conexão trafegados na rede de forma contínua, separando-os em contadores individuais a cada 300 segundos. Este intervalo de tempo para a coleta dos contadores pode ser configurável conforme a necessidade do administrador da rede.

O IAS da FHGe originalmente utiliza um sistema de detecção de intrusão baseado em redes neurais. Na UFSM foi desenvolvido um trabalho que utiliza séries temporais

para detecção de intrusões (LUNARDI et al., 2008), porém, este apresenta um número muito grande de falsos positivos. A idéia de usar *wavelets* para a filtragem e análise de sinais apresenta uma nova proposta que na maioria das vezes é menos custosa a nível computacional (HUANG; THAREJA; SHIN, 2006).

Wavelets é uma técnica matemática que, dentre muitas aplicações, também é usada para tratamento e filtragem de sinais (STRANG, 1993). O presente trabalho objetiva estudar os métodos de análise em um IDS baseado na detecção de anomalias e realizar a filtragem dos alarmes gerados pelo DIBSeT (LUNARDI et al., 2008) através de *wavelets*, com o objetivo de tornar mais eficiente a geração de alertas na detecção de anomalias.

Este trabalho está organizado da seguinte maneira: No segundo capítulo é apresentada uma revisão bibliográfica referente aos principais ataques a redes de computadores, dando um enfoque especial aos ataques mais difíceis de serem tratados por métodos tradicionais. Também neste mesmo capítulo, é mostrado como se classifica os tipos de detecção de ataques e descrita as características das bases de dados estudadas neste trabalho. O terceiro capítulo faz uma revisão bibliográfica sobre séries temporais e *wavelets*. O capítulo quatro, apresenta uma proposta de implementação para detecção de ataques com a solução encontrada a partir da evolução do DIBSeT e agregação de filtros *wavelets*. O quinto capítulo relata o desenvolvimento do DIBSeT-W, mostrando comparações e resultados. Finalmente, o capítulo seis apresenta as conclusões e as principais questões que ainda estão em aberto no desenvolvimento do DIBSeT-W.

1.1 Recursos utilizados

Para realizar este trabalho foram utilizadas as instalações do CRS/INPE MCT (Centro Regional Sul de Pesquisas Espaciais do Instituto Nacional de Pesquisas Espaciais), que possui parceria com UFSM. O prédio sede do CRS/INPE MCT possui uma rede estruturada que segue o padrão EIA/TIA 568a (ELETRONIC INDUSTRIES ASSOCIATION AND TELECOMMUNICATIONS INDUSTRY ASSOCIATION, 2008).

Foram utilizados computadores do GMICRO para realizar a implementação do DIBSeT-W. A captura dos dados foi feita no Centro de Processamento de Dados da Universidade Federal de Santa Maria, através do IAS. Neste trabalho também foi usada uma base de dados do DARPA (Defense Advanced Research Projects Agency, 2008), muito utilizada em trabalhos científicos, pois todos os ataques contidos nesta base de dados são conhecidos e

documentados.

1.2 Situação atual dos detectores de intrusões

Hoje em dia existem vários sistemas de detecção de intrusões, porém não existe um único que possua todas as características que agradem a todos os públicos. Um exemplo destes sistemas é o SNORT (SNORT - http://www.snort.com.br, 2008), um dos mais conhecidos detectores de intrusões, que apresenta bom funcionamento somente em redes de pequeno e médio porte (MARTINS et al., 2002).

Para a análise dos dados em um IDS, os métodos mais conhecidos são os métodos baseados em assinaturas e aqueles baseados em anomalias (NORTHCUTT, 1999). Quanto à eficiência, ambos os métodos tem limitações. Os métodos baseados em assinaturas exigem um conhecimento prévio a respeito da forma como cada ataque a uma rede ocorre, ou seja, sua assinatura, por isso, são menos eficientes na identificação de ataques que se utilizam de técnicas ainda não conhecidas. Já os métodos baseados na detecção de anomalias podem gerar um excessivo número de falsos positivos, inviabilizando a intervenção automatizada ou acarretando a geração de muitos falsos alertas para a intervenção humana (NORTHCUTT, 1999).

As principais pesquisas atuais concentram-se na busca por assinaturas de ataques, já que a detecção por anomalias de tráfego conhecida atualmente gera muitos falsos positivos. Atualmente a diversidade de ataques é muito grande e o número de diferentes tipos de ataques continua crescendo, principalmente ataques distribuídos (KOMPELLA; SINGH; VARGHESE, 2007). Assim, torna-se necessário um estudo constante de cada ataque que está sendo usado. Com o intuito de criar um sistema de detecção de intrusão baseado em anomalias considerando o histórico dos dados obtidos, tornando-o assim mais eficaz, este trabalho tem por objetivo utilizar séries temporais em detecção de intrusões, e refinar seus resultados através de *wavelets*.

2 ATAQUES E DEFINIÇÃO DE INTRUSÃO

Hoje em dia elementos de rede como os *bridges* ou *switches* são componentes de redes complexos, com milhares de linhas de código. Mesmo assim podem ser vulneráveis a ataques, permitindo até a execução remota de código no seu controlador. Os sistemas de detecção de intrusão baseados em rede capturam todos os pacotes que passam pela rede detectando qualquer anomalia presente nestes pacotes. Mas, o modo como essas anomalias são detectadas depende fundamentalmente do modo como os pacotes são analisados. Neste capítulo são apresentados alguns ataques às redes de computadores que podem gerar estas anomalias. O capítulo contém as principais formas de detecção de intrusão usadas atualmente, e no final, também apresenta as bases de dados usadas para os testes e validações do DIBSeT-W.

2.1 Tipos de ataques

Os tipos de ataques escolhidos para ser estudados neste trabalho foram selecionados devidos as suas características e efeitos colaterais anômalos.

2.1.1 *SYN Attack*

O ataque *SYN Flood* é um ataque do tipo (*Denial of Service*) (DoS) (KARGL; MAIER; WEBER, 2001) que inunda (*flood*) a vítima com pacotes TCP SYN (CERT - TCP SYN Flooding and IP Spoofing, 2008). Este ataque tira vantagem do comportamento do *three way handshake* efetuado pelo protocolo TCP no processo de uma conexão.

Quando um cliente tenta começar uma conexão TCP com um servidor, o cliente e o servidor trocam uma série de mensagens, que normalmente são assim:

- O cliente requisita uma conexão enviando um SYN (*synchronize*) ao servidor;
- O servidor confirma esta requisição mandando um SYN-ACK (*synchronize and*

acknowledgment) de volta ao cliente;

- O cliente por sua vez responde com um ACK (*acknowledgment*), e a conexão está estabelecida. Isto é o chamado aperto de mão em três etapas (*Three-Way Handshake*).

Um cliente malicioso pode não mandar esta última mensagem ACK. O servidor irá esperar por isso por um tempo, já que um simples congestionamento de rede pode ser a causa do ACK faltante. Esta chamada conexão semi-aberta irá ocupar recursos no servidor, ou seja, é armazenada em um *buffer* até que um pacote ACK seja recebido ou termine o tempo de espera (*timeout*) e seja descartado. Pode ser possível ocupar todos os recursos da máquina, com pacotes SYN. Quando um atacante envia, num curto intervalo de tempo, um grande volume de pacotes SYN para um servidor e não responde com um ACK as respostas SYN-ACK recebidas, o *buffer* do servidor torna-se congestionado e este não consegue processar novos pedidos de conexão (legítima ou não) que acabam sendo negados, a figura 2.1 ilustra o ataque.

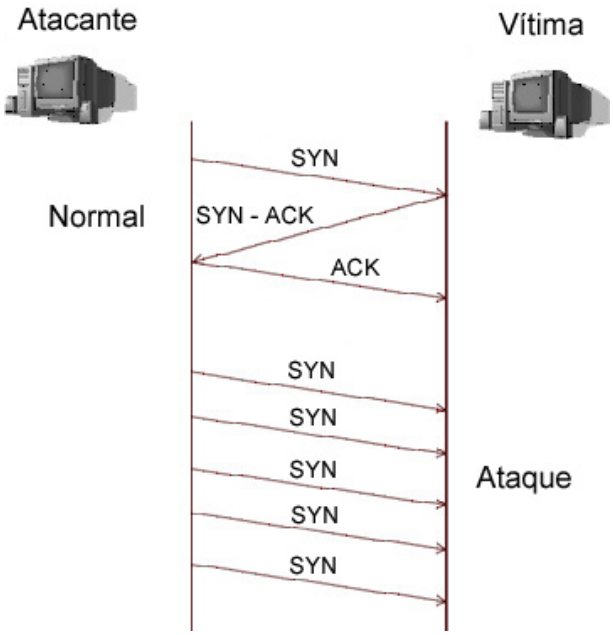

Figura 2.1: *SYN Attack*

Um ataque *SYN Flood*, normalmente, é feito com os IPs forjados ou clonados (*spoofing*), para que o atacante não receba os ACKs de suas falsas solicitações e dificulte a sua identificação. Quando o atacante utiliza um IP clonado, a máquina clonada receberá os pacotes SYN/ACK como resposta do servidor, desta forma tanto a máquina que recebeu

o SYN quanto a que receber o SYN/ACK ficarão ocupadas e não poderão responder a novas conexões.

2.1.1.1 Identificação

O SYN ataque possui uma característica muito marcante que é o envio massivo de pacotes TCP SYN, e em decorrência disto, a conexão não é continuada. Portanto, existem picos de pacotes SYN, com baixo número de pacotes ACK, RST ou FIN. As duas principais formas de detectar um ataque TCP SYN são através da detecção de picos de pacotes SYN no tráfego da rede e através da relação de pacotes SYN com pacotes ACK, RST e FIN na rede.

2.1.1.2 Prevenção

Não se resolve negação de serviço por *SYN flood* limitando o número de conexões por minuto (como usar o módulo *limit* ou *recent* do *iptables*), pois as conexões excedentes seriam descartadas pelo *firewall*, sendo que desta forma o próprio *firewall* tiraria o serviço do ar. Neste caso, a tarefa do atacante se torna mais fácil ainda (NOGUEIRA, 2005).

Algumas contramedidas para este ataque são os *SYN Cookies* (CERT - TCP SYN Flooding and IP Spoofing, 2008). Apenas máquinas Sun e Linux usam *SYN Cookies*. O uso de *SYN Cookies* permite que o servidor aloque recursos do sistema apenas na última fase do *handshake* do protocolo TCP, evitando assim a reserva de recursos para conexões semi-abertas.

Porém mesmo com a utilização de *SYN Cookies*, o ataque poderá causar prejuízos à vítima quando o atacante possuir uma banda muito maior que a vítima ou utilizar várias máquinas.

2.1.2 *PortScan*

Os *port scanners* são programas que percorrem as principais portas e serviços do sistema em busca de respostas (NOGUEIRA, 2005). Um exemplo similar seria uma pessoa percorrendo uma rua e indo de porta em porta das casas, verificando se algum dono deixou alguma das portas abertas. Existem inúmeros tipos de *scanners*, e eles são de grande ajuda tanto para *hackers*, como para administradores de sistemas. Os mais populares são de domínio público (GPL), porém também existem *scanners* comerciais disponíveis (na grande maioria para a plataforma *Microsoft*). Para o administrador, conhecer as fraquezas

do sistema é algo fundamental, pois cedo ou tarde alguém pode vir a bater sua porta.

Periodicamente são lançados *scanners* que detectam as vulnerabilidades mais recentes. Cabe ao administrador providenciar a correção antes que alguém consiga tirar proveito de alguma vulnerabilidade do sistema. Observe que um scanner somente detecta o problema, raros são aqueles que automaticamente se aproveitam da falha para obter algum nível de acesso.

2.1.2.1 Identificação

Como a utilização de *port scanner* por invasores de sistemas (*crackers*) cresce a cada dia, diferentes técnicas de *portscanning* foram desenvolvidas para tentar driblar a proteção criada pelos gerentes de segurança de redes para interceptar a técnica. Os tipos de *scanners* são (NOGUEIRA, 2005):

- *TCP Connect Scan*: este tipo de *scanner* se conecta à porta e executa o *handshakes* básico (SYN/ACK, ACK). Ele é facilmente detectável;

- *TCP SYN Scan*: conhecido como *half-open scanning*, devido à conexão parcial TCP durante a operação. Desta forma, evita que o *log* da operação fique no sistema. Normalmente, o programa envia um pacote SYN para a porta-alvo, se for recebido um SYN/ACK, significa que a porta está ativa naquele momento;

- *UDP Scan*: trata-se de um dos processos mais lentos de *scanning*, pois depende de fatores de utilização da rede e de recursos de sistema. O *scanner* envia um pacote UDP para a porta alvo, se a resposta for um *ICMP port unreachable*, a porta encontra-se fechada, caso contrário, o *scanner* deduz que a porta está aberta;

- *TCP Null Scan*: neste caso, o *scanner* desativa todos os *flags* e aguarda do alvo um RST para identificar todas as portas fechadas. Baseado na RFC 793 (RFC 793, 1981);

- *TCP FIN Scan*: o *scanner* envia pacotes FIN para a porta-alvo e espera o retorno de um RST para as portas fechadas. Baseado em RFC 793;

- *TCP XMAS TREE Scan*: neste caso, o *scanner* envia pacotes FIN, URG e *Push* para a porta-alvo e espera o retorno de um RST para as portas fechadas. Baseado também na RFC 793.

2.1.2.2 Prevenção

Antes de querer proteger um sistema de ataques de *port scan* é necessário ter em mente as seguintes afirmações:

1. Todo o sistema que disponibiliza um serviço para usuários provenientes da Internet deverá ter uma porta aberta em seu computador;

2. A maioria dos programas utilizados para servir usuários provenientes da Internet ou até mesmo da rede local possui portas padrão;

3. 95% dos programas de *portscaning* disponíveis na Internet procuram por portas padrão;

4. Toda porta pode ser fechada e suas conexões canceladas ou rejeitadas.

Por exemplo, para se evitar qualquer um dos *portscan TCP connection*, é necessário filtrar as conexões externas da rede, ou seja, rejeitar conexões da Internet para as portas ou serviços específicos. Mas para que isso seja possível, é preciso que se utilize uma ferramenta chamada IPFW (IPFIREWALL). O IPFW é um filtro de pacotes nativo do *kernel* de alguns sistemas operacionais - entre eles o Linux e o FreeBSD (NOGUEIRA, 2005) - sistemas abertos bem conhecidos. Um administrador pode elevar o nível de proteção de sua rede interna, caso o IPFW seja instalado no *gateway* da mesma.

Até mesmo um usuário comum preocupado com a sua segurança pessoal pode utilizá-lo para proteger seu próprio PC, rodando Linux, de conexões indesejáveis. Por ser parte de pacotes de *software* de livre distribuição, nenhum dos dois terá algum ônus financeiro por utilizar este filtro. Para sistemas operacionais da família *Microsoft* é necessário instalar algum aplicativo *firewall*, porém muitos deles são pagos.

2.1.3 DoS

Os ataques DoS (*Denial of Service*) são muito conhecidos no âmbito da comunidade de segurança de redes (MIRKOVIC; REIHER, 2004). Esses ataques acontecem através do envio indiscriminado de requisições a um computador-alvo e visam causar a indisponibilidade dos serviços oferecidos por ele (BOLZ; ROMMEY; ROGERS, 2004). Ao longo dos últimos anos, uma categoria de ataques de rede tem se tornado bastante conhecida: a intrusão distribuída. Neste novo enfoque, os ataques não são baseados no uso de um único

computador para iniciar um ataque, no lugar são utilizados centenas ou até milhares de computadores desprotegidos e ligados na Internet para lançar coordenadamente o ataque (CERT - Denial-of-Service Attack via ping, 2008).

A tecnologia distribuída não é completamente nova, no entanto, vem amadurecendo e se sofisticando de tal forma que até mesmo vândalos curiosos e sem muito conhecimento técnico podem causar sérios danos. A este respeito o CAIS (Centro de Atendimento a Incidentes de Segurança, 2008) tem sido testemunha do crescente desenvolvimento e uso de ferramentas de ataque distribuídas (CERT - Denial-of-Service Tools, 2008), em várias categorias: *sniffers*, *scanners*, DoS. Seguindo na mesma linha de raciocínio, os ataques *Distributed Denial of Service*, nada mais são do que o resultado de se conjugar os dois conceitos: negação de serviço e intrusão distribuída (CERT - Denial-of-Service Vulnerabilities in TCP/IP Stacks, 2008).

Os ataques DDoS podem ser definidos como ataques DoS diferentes partindo de várias origens, disparados simultânea e coordenadamente sobre um ou mais alvos. De uma maneira simples, ataques DoS em larga escala. Os primeiros ataques DDoS documentados surgiram em agosto de 1999, no entanto, essa categoria se firmou como ameaça na Internet na semana de 7 a 11 de fevereiro de 2000 (CERT - Denial-of-Service Vulnerabilities in TCP/IP Stacks, 2008), quando *crackers* deixaram inoperantes por algumas horas sites como o *Yahoo*, o *EBay*, *Amazon* e CNN. Uma semana depois, teve-se notícia de ataques DDoS contra sites brasileiros, tais como: UOL, *Globo On Line* e IG, causando com isso uma certa apreensão generalizada.

2.1.3.1 Identificação

Em um *Denial of Service* comum, uma máquina ataca a outra explorando falhas de *software* e de protocolos. Já no DDoS, o atacante utiliza várias máquinas para atacar uma máquina alvo. O objetivo do ataque é esgotar algum recurso da máquina-alvo (CERT - Denial-of-Service Developments, 2008).

Um *hacker* pode desenvolver um sistema para explorar alguma vulnerabilidade dos protocolos TCP/IP. Este sistema é composto por dois programas: Mestre e Zumbi.

Para um atacante dar início ao ataque, é necessário que um número grande de computadores façam parte do ataque. Uma das formas encontradas para se ter tantas máquinas, foi inserir programas de ataque DDoS em vírus ou em *softwares* maliciosos. Nestes com-

putadores atacados fica instalado o programa Zumbi.

Quando o *cracker* já tem uma quantidade suficiente de computadores comprometidos com o programa Zumbi, ele dispara o ataque. Para isto ele utiliza o programa Mestre, o qual é capaz de se comunicar de forma direta (e normalmente criptografada) com os Zumbis. Através deste programa Mestre, o *cracker* identifica o alvo do ataque, e como será feito o ataque. Veja na figura 2.2.

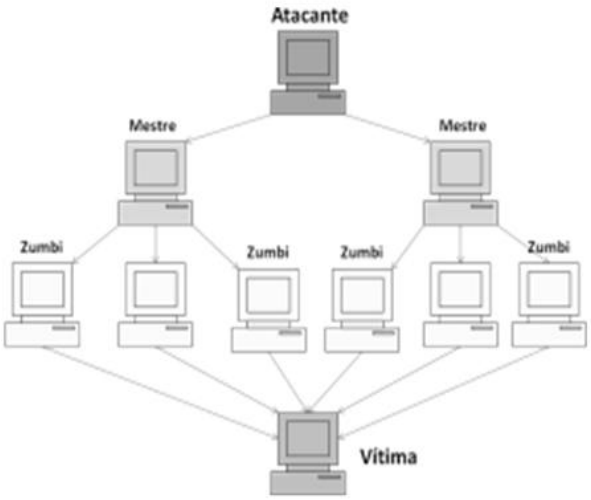

Figura 2.2: *Distributed Denial of Service*

Uma vez recebido o comando de ataque, todos os computadores Zumbis começam simultaneamente a atacar o computador-alvo.

Normalmente os programas DDoS utilizam *IP Spoofing* (falsificação de IPs). Desta forma a equipe responsável pela administração do sistema que está sendo atacado não tem como saber quais os IPs fazem parte do ataque, pois os IPs são falsos. Isto dificulta as tentativas de filtragem. A única saída é tentar descobrir alguma característica nos pacotes recebidos que possa permitir separar os pacotes de ataque dos pacotes de acesso legítimo. Isto não é fácil, e exige que a equipe responsável pela segurança esteja bem treinada e preparada para este tipo de ataque.

2.1.3.2 Prevenção

Um *Connection Flooder* é um programa que tenta abrir o máximo de conexões com uma máquina-alvo a fim de esgotar todo o limite disponível de conexões da máquina visando impedir que outros usuários possam usufruir do serviço (NOGUEIRA, 2005).

O sucesso desta técnica se faz presente quando é conseguido esgotar o número má-

ximo de conexões permitidas pelo servidor. Uma ótima maneira de se bloquear um *connection flooder* é alterar a configuração padrão do servidor. Dessa forma o número de conexões simultâneas poderá ser limitado. Também é aconselhável fechar todas as portas dos serviços que não estão sendo usados pelo servidor (PENG; LECKIE; RAMAMOHANARAO, 2007). Também é possível limitar o número de conexões de um único cliente.

2.1.4 *Brute Force*

Brute Force consiste basicamente em uma técnica que, através do método de tentativa e erro, procurar obter sucesso em uma determinada sessão de autenticação de usuário, ou seja, um *Brute Force* é geralmente usado para se obter nomes de usuário e senhas de determinados programas ou serviços (TANENBAUM, 2003).

O fato de se implementar este conceito com o uso de programas para automatizarem a sua implementação é o que torna o *Brute Force* uma técnica. O *Brute Force* pode vir a ser uma técnica muito eficiente para se obter acesso a um sistema remoto, pois ainda existem milhares de falhas nas implementações de programa em relação a sua criação de senhas (NOGUEIRA, 2005). Porém, vale lembrar que esta técnica pode ser muito demorada e cansativa, pois podem ser testadas milhões de senhas até que se encontre uma válida, e outro motivo que leva a uma grande lentidão é o fato de que alguns serviços possuem um limite de tentativas de senhas, podendo descartar novas conexões por certo período de tempo fazendo com que o atacante precise esperar para poder fazer novas tentativas.

Para aumentar a chance de acertos, geralmente é usada uma lista de palavras, que se refere a um conjunto de palavras, que geralmente são usadas através do programa *Brute Force* como "palpites"para as tentativas de obtenção de usuários ou senhas. Existem servidores que não aceitam senhas fáceis, por exemplo, "12345", logo, uma boa lista de palavras para esse servidor poderia ser uma que misture letras e números ou que não possuísse menos de seis caracteres aleatórios do tipo "d8U10p". Para a criação de uma lista de palavras, basta escrever linha por linha várias palavras em um arquivo de texto, vale lembrar que quanto mais palavras a lista compreender, mais chances são as de acertar a senha correta.

2.1.4.1 Identificação

Existem algumas implementações para o *Brute Force* que variam de acordo com a necessidade e disponibilidade de sua utilização (Centro de Atendimento a Incidentes de

Segurança, 2008). São elas:

- Manual: neste caso o atacante tenta o *Brute Force* sem a utilização de um *software* específico. Ele, por sua vez, efetua suas tentativas diretamente no decorrer do uso de programa pelo seu interpretador de comandos. Por exemplo, um usuário utilizando esta técnica para a obtenção de uma senha válida para o usuário ROOT em serviço de FTP.

- Local: esta representa a implementação do *Brute Force* para um usuário que já possui uma senha de acesso local válida no sistema. Esta implementação consistem em obter novas senhas para outros usuários do sistema explorando serviços que estão ativos na máquina. Pode-se tentar descobrir uma senha para um usuário dono de algum serviço com acesso privilegiado, como por exemplo o TELNET.

- Remoto: neste caso o *Brute Force* é utilizado para a obtenção de usuários e senhas válidos em máquinas remotas (disponíveis *on-line*). Esta é, sem dúvida, a implementação mais utilizada pelos invasores de sistemas que visam acesso em grandes servidores. Serviços como MySQL, POP, SSH, RSH, TELNET são os mais visados, pois uma vez que se consiga uma senha válida, fica extremamente mais fácil para o invasor obter acesso total no sistema utilizando alguma outra técnica.

2.1.4.2 Prevenção

Se proteger de um ataque do tipo *Brute Force* é relativamente complicado. Inúmeras são as providências que podem ser tomadas, mas ainda assim continua sendo possível que um atacante em potencial consiga obter sucesso em uma única tentativa. O fator mais importante na prevenção de técnicas de *Brute Force* está na própria senha, pois, por exemplo, um administrador de sistema pode configurar todos os serviços da máquina para descartar a execução após um número *X* de tentativas de *login* inválidos e mesmo assim, por sorte, em uma única tentativa, o atacante pode digitar a senha correta e obter acesso ao sistema (TANENBAUM, 2003).

Outro exemplo seria em relação ao próprio administrador. Por exemplo, se ele configurasse todo o sistema para não permitir mais que 3 tentativas de *login* por sessão em um serviço e utilizasse as melhores chaves de encriptação para as senhas, nada disso teria resultado algum se a sua senha fosse "1234", pois qualquer usuário, mesmo estando sobre

todas as proteções, poderia em uma primeira e única tentativa, garantir o acesso como se fosse o Administrador do sistema.

Portanto, para que possa dificultar o máximo possível um ataque de *Brute Force*, é necessário que se tenha uma ótima política de criação de senhas.

2.2 Formas de detecção de intrusão

A detecção de intrusão é uma área de pesquisa em expansão na segurança em redes de computadores. Com o grande crescimento da interconexão de computadores em todo o mundo, é verificado um conseqüente aumento nos tipos e no número de ataques a esses sistemas, gerando uma complexidade muito elevada para a capacidade dos tradicionais mecanismos de prevenção (BEJTLICH, 2004). Para a maioria das aplicações atuais, é praticamente inviável a simples utilização de mecanismos que diminuam a probabilidade de eventuais ataques. Um ataque, em casos extremos, pode causar a interrupção total de um serviço ou deixá-lo extremamente lento (GOODALL, 2006). O processo de auditoria e posterior restauração manual, normalmente, é lento e oneroso. Isso justifica o estudo e desenvolvimento de mecanismos mais eficientes que a simples prevenção.

Intrusão é uma ação, ou conjunto delas, que tem por objetivo comprometer a confidencialidade, integridade e disponibilidade de um sistema. De uma forma mais abrangente uma intrusão é qualquer violação da política de segurança de um sistema (TANENBAUM, 2003). Um IDS é um processo ou dispositivo que analisa as atividades do sistema e da rede e tem como objetivo detectar ações anômalas, impróprias e incorretas, sendo um componente de defesa fundamental em uma organização (NAKAMURA; GEUS, 2002). Além disso, estes sistemas também podem detectar ataques provenientes de portas legítimas que passam pelo *firewall*, abalando a segurança interna. O IDS funciona como uma espécie de alarme contra invasões, tendo como base de suas detecções assinaturas conhecidas ou desvios de comportamento.

A maneira como um IDS detecta anomalias pode variar de sistema para sistema, mas o objetivo final de qualquer IDS é identificar uma intrusão antes que esta danifique algum recurso. Um IDS protege um sistema de ataques, também pode monitorar as atividades da rede, realizar auditoria nas configurações da rede e do sistema para detectar vulnerabilidades, analisar integridade de dados e muito mais, dependendo do método de detecção escolhido (DWYER, 2003).

Um IDS irá, normalmente, notificar um especialista humano sempre que detectar alguma atividade considerada suspeita ou fora dos padrões normais. Detecção de intrusão em sistemas computacionais é uma tecnologia relativamente nova. IDSs mais recentes podem, opcionalmente, atuar automaticamente ao detectar uma anomalia. Sem a detecção de intrusão, muitos ataques podem acontecer sem que sejam percebidos. Assim como não é possível obter informações sobre ataques que ocorrem de forma bem-sucedida, não é possível obter informações suficientes para poder prevenir um novo ataque. A maioria dos IDSs possuem três componentes fundamentais (NORTHCUTT, 1999):

- Fonte de Informações: são as diferentes fontes de informações sobre eventos, usadas para determinar quando um ataque ocorre. As fontes de informações podem ser um *host*, uma rede, ou também podem ser consideradas apenas um segmento da rede.

- Análise: é a parte do sistema de detecção de intrusão que verifica os eventos derivados da fonte de informações, determinando quando estes eventos indicam que uma intrusão está ocorrendo ou já ocorreu. Os métodos de análise são as detecções baseadas em assinaturas e as detecções baseadas em anomalias.

- Resposta: é o conjunto de ações que o SDI faz quando detecta uma intrusão. Estas ações são tipicamente agrupadas em medidas ativas e passivas, onde medidas ativas envolvem a intervenção automatizada em parte do sistema e medidas passivas abrangem a geração de alertas e relatórios para interpretação e intervenção humana.

Um IDS possui algumas características desejáveis (LAUREANO, 2005):

- Deve rodar continuamente sem interação humana e deve ser seguro o suficiente de forma a permitir sua operação em *background*, mas deve ser de fácil compreensão e operação;

- Deve ser tolerante a falhas, de forma a não ser afetada por uma falha do sistema, ou seja, sua base de conhecimento não deve ser perdida quando o sistema for reinicializado;

- Deve resistir às tentativas de mudança (subversão) de sua base, ou seja, deve monitorar a si próprio de forma a garantir sua segurança;

- Deve ter o mínimo de impacto no funcionamento do sistema;
- Deve detectar mudanças no funcionamento normal;
- Deve ser de fácil configuração, cada sistema possui padrões diferentes e a ferramenta de IDS deve ser adaptada de forma fácil aos diversos padrões;
- Deve cobrir as mudanças do sistema durante o tempo, como no caso de uma nova aplicação que comece a fazer parte do sistema;
- E deve ser difícil de ser enganado.

Os IDSs podem ser classificados em diferentes categorias conforme sua arquitetura e abordagem (NORTHCUTT, 1999):

- Arquitetura:
  - Baseada em *host*: analisa cada máquina da rede;
  - Baseada em rede: analisa toda a rede.
- Abordagem:
  - Baseada em assinatura: analisa padrões de comportamentos;
  - Baseada em anomalias: analisa se o comportamento segue um padrão normal.

2.2.1 Arquitetura

Os detectores de intrusão podem ter diferentes categorias quanto à sua arquitetura. Eles podem analisar cada máquina da rede isoladamente, ou analisar a rede como um todo.

2.2.1.1 Detecção de intrusão baseada em host

Sistemas HIDS (*Host-based Intrusion Detection System*) que monitoram e procuram por intrusões em um único *host*. São sistemas baseados em agente, ou seja, exigem a instalação de agentes em todos os sistemas monitorados. Isso faz com que se tornem mais completos e intrusivos, além de serem mais trabalhosos para instalar e administrar. Consistem, tipicamente, de sistemas especialistas que monitoram chamadas de funções do sistema operacional, acesso a arquivos considerados críticos, uso de recursos como

processador, disco e memória procurando por padrões que determinem um ataque ou intrusão ou por desvios significativos em relação ao perfil de uso considerado normal e regular. A característica principal e que distingue esta classe é que a visão do IDS está restrita apenas ao *host* no qual ele está instalado.

Os sistemas baseados em *host* são bastante diversificados em relação à quantidade de recursos que oferecem. A maioria desses mecanismos faz o monitoramento de arquivos de registro de eventos do sistema referente a atividades básicas como tentativas de *login* sem sucesso, violações de acesso e criação de novas contas de usuário. Outros produtos fazem o monitoramento de mensagens do sistema operacional para certos eventos considerados hostis. Existem, também, os sistemas baseados em *host* mais avançados, que podem capturar instalações de código malicioso e, inclusive, terminar a execução de processos ilegais (NAKAMURA; GEUS, 2002). Basicamente são funções de um HIDS (NAKAMURA; GEUS, 2002):

- Monitorar acessos e alterações em arquivos importantes do sistema;
- Controlar o uso da Unidade Central de Processamento (UCP);
- Analisar modificações nos privilégios de acesso dos usuários;
- Controlar programas em execução;
- Realizar verificações da integridade dos arquivos do sistema;
- Detectar ataques de força bruta através da análise de arquivos de registro de eventos do sistema.

Os sistemas de detecção de intrusão baseados em *host* possuem como pontos positivos (NAKAMURA; GEUS, 2002):

- Verificação do sucesso ou a falha de um ataque, com base nos registros do sistema;
- Monitoramento detalhado das atividades específicas do sistema, como: acesso a arquivos, modificação em permissões de arquivos, *logon* e *logoff* do usuário e funções do administrador;
- Detecção de ataques que ocorrem fisicamente no servidor;

- Ataques que utilizam criptografia podem passar despercebidos pela rede, mas podem ser descobertos pelo HIDS;
- É independente da topologia de rede;
- Gera poucos falsos positivos, ou seja, os administradores recebem poucos alarmes falsos de ataques;
- Não necessita de *hardware* adicional.

Os pontos negativos que devem ser considerados no HIDS são (NAKAMURA; GEUS, 2002):

- Dificuldade de gerenciá-lo e configurá-lo em todos os computadores que devem ser monitorados;
- É dependente do sistema operacional, isto é, um HIDS que funciona no Linux é totalmente diferente de outro que opera no Windows;
- Caso o HIDS seja invadido, as informações podem ser perdidas;
- Necessita de espaço de armazenamento adicional para os registros do sistema;
- Por terem como base os registros do sistema, podem não ser tão eficientes em Sistemas Operacionais que geram poucas informações de auditoria, como o Microsoft Windows XP;
- Apresenta uma baixa de desempenho no computador que está sendo monitorado;
- Não é capaz de emitir alertas em tempo real;
- Há uma carga extra da CPU do *host* para a execução das tarefas do HIDS.

2.2.1.2 Detecção de intrusão baseada em rede

Um NIDS (*Network Intrusion Detection System*) consiste em um conjunto de sensores que são distribuídos em vários pontos da rede para monitorar o tráfego através da análise dos pacotes da rede. Desta forma, um IDS instalado pode monitorar uma grande rede e não interferir no seu desempenho. Os NIDS são considerados *sniffers* de alto nível, capturando e analisando os pacotes que passam pela rede de forma passiva, sem que os

outros sistemas percebam isso (NORTHCUTT, 1999). Para isso configuram a placa de rede utilizada para monitoração para funcionar em modo promíscuo. Uma placa de rede em modo promíscuo captura todos os pacotes que chegam independentemente de serem ou não endereçados a ela.

Os NIDS são muito eficientes contra ataques de varredura de portas, falsificação de IP (*spoofing*) ou ataques de inundação (*flooding*) como o ataque do tipo *SYN flood*. Por outro lado, podem ter dificuldades em analisar todos os pacotes numa rede grande ou sobrecarregada em períodos de tráfego intenso. Outro ponto importante é que os NIDS não podem analisar informações criptografadas (NORTHCUTT, 1999).

Embora os IDSs baseados em rede apresentem boa eficiência em relação à detecção de ataques, eles trazem algumas questões que devem ser consideradas. Os NIDSs têm como pontos positivos (NAKAMURA; GEUS, 2002):

- Um único IDS pode fornecer monitoramento para múltiplas plataformas;
- Analisam pacotes;
- Monitora atividades suspeitas em portas conhecidas, como a porta TCP 80, que é utilizada pelo HTTP;
- Os ataques podem ser detectados em tempo real e o administrador pode determinar rapidamente o tipo de resposta apropriada;
- Possuem capacidade de detectar não só ataques, mas também, tentativas de ataque que não tiveram sucesso;
- Apresentam cuidados para que um *cracker* não possa apagar seus rastros, caso consiga invadir um equipamento;
- Um *cracker* terá dificuldades em saber que existe um NIDS monitorando suas atividades;
- Não causam impacto no desempenho da rede.

Os pontos negativos que podem ser encontrados em NIDS são (NAKAMURA; GEUS, 2002):

- Incapacidade de monitorar grandes redes com alto tráfego;

- Dificuldade de compreensão de protocolos de aplicação específicos;
- Não são capazes de monitorar tráfego cifrado;
- Possuem dificuldade de utilização em redes segmentadas.

2.2.2 Abordagem

Os detectores de intrusão podem ter diferentes abordagens. Eles podem analisar as atividades à procura de assinaturas, ou analisar as atividades em busca de anomalias no tráfego normal da rede.

2.2.2.1 Detecção de intrusão baseada em assinaturas

Este tipo de IDS analisa as atividades do sistema procurando por eventos que correspondam a padrões pré-definidos de ataques e outras atividades maliciosas. Estes padrões são conhecidos como assinaturas e geralmente cada assinatura corresponde a um ataque.

Uma assinatura pode ser utilizada para detectar um ou múltiplos tipos de ataques. Assinaturas podem estar presentes em várias partes de um pacote, e são conhecidas como uma combinação de dados que caracterizam o ataque. Dois ou mais protocolos conjugados e em situação anômala identificam o ataque, por exemplo: alto número de pacotes TCP SYN e um baixo número de pacotes TCP ACK.

Uma desvantagem desta técnica de detecção é que ela pode detectar somente ataques conhecidos, ou seja, que estão incluídos no conjunto de assinaturas que o IDS possui, necessitando-se assim de constante atualização deste conjunto (NORTHCUTT, 1999). Este é o principal fator motivador para pesquisa e desenvolvimento de sistemas de detecção de intrusos baseado em anomalias.

2.2.2.2 Detecção de intrusão baseada em anomalias

Este tipo de análise parte do princípio que os ataques são ações diferentes das atividades normais de sistemas. Detectores baseados em anomalias constroem um perfil que representa o comportamento normal de um usuário, *host* e conexão de rede. Estes detectores monitoram a rede e usam várias medidas para determinar quando os dados monitorados estão fora do normal, ou seja, desviando do perfil. Como ela detecta comportamentos não usuais, acaba detectando sintomas de ataques sem um conhecimento prévio deles, além de produzir informações que podem ser usadas na definição de assinaturas de

detectores baseados em assinaturas. Uma desvantagem é a geração de um grande número de alarmes falsos devido ao comportamento imprevisível de usuários e do próprio sistema (NORTHCUTT, 1999).

Os IDSs surgiram para auxiliar a monitoração de computadores e também na monitoração de redes de computadores, como uma forma preventiva de ataques. Eles complementam a segurança de uma rede, fornecendo informações de grande valia para a análise de máquinas e redes invadidas.

Entretanto os IDSs ainda são difíceis de configurar e operar. Geralmente não podem ser eficientemente usados por pessoas inexperientes, já que o processo de ajuste e posicionamento de um IDS não é trivial (DERI; SUIN; MASELLI, 2003). É necessário realizar uma análise do tráfego da rede e, a partir dessa análise, ajustar os tipos de informações que deverão ser monitoradas e em quais pontos da rede deve ser feita essa monitoração. Os IDSs, de modo geral, não são perfeitos e muitos erros de detecção podem ocorrer. Podem ser classificados como: falso positivo, falso negativo e erros de subversão.

1. Falso positivo ocorre quando a ferramenta classifica uma ação como uma possível intrusão, quando na verdade trata-se de uma ação legítima;

2. Falso negativo ocorre quando uma intrusão real acontece, mas a ferramenta permite que ela passe como se fosse uma ação legítima;

3. Subversão ocorre quando o intruso modifica a operação da ferramenta.

2.3 Base de dados de tráfego de redes

Para possibilitar os testes com o DIBSeT-W e comparação com os resultados obtidos por trabalhos relacionados, foi necessário utilizar uma base de dados conhecida. De nada adiantaria testar o IDS com os contadores de saída/entrada de uma rede qualquer. Como poderíamos ter a certeza de que o IDS está funcionando corretamente se não temos plena convicção do que acontece com o tráfego desta rede? Para solucionar este problema, foi usada a base de dados do DARPA (Defense Advanced Research Projects Agency, 2008). Depois de todos os testes e ajustes com essa base confiável, podemos passar para a próxima fase do trabalho e analisar os resultados do DIBSeT-W com os dados capturados da rede que passa no *backbone* da Universidade Federal de Santa Maria, através do PROBE do IAS (POHLMANN; PROEST, 2006).

2.3.1 DARPA

A ARPA (*Advanced Research Project Agency*), órgão do Departamento de Defesa dos EUA, foi criada no final da década dos 50, no auge da guerra fria, com o objetivo de desenvolver projetos que garantissem ao país a condição de superpotência tecnológica. Em 1965, a Arpa iniciou os estudos para a criação de uma rede de telecomunicações, descentralizada, baseada em computadores e, que não pudesse ser destruída por nenhum ataque localizado. Esta rede, a Arpanet, inicialmente interligou instituições militares. E em meados dos anos 70 incorporou um consórcio formado pelas Universidades da Califórnia (UCLA), Santa Barbara (UCSB), o *Stanford Research Institute* (SRI) e a Universidade do Utah, e deu origem a Internet.

A partir da década dos 70, passou a se chamar DARPA (*Defense Advanced Research Projects Agency*). Como parte da DARPA, o AFRL (*Air Force Research Laboratory*) trabalha na pesquisa e avaliação de detecção de intrusão desde o ano de 1998. E com esse objetivo, o *System Technology Group of Massachusetts Institute of Technology* (MIT), *Lincoln Laboratory* e *Air Force Research Laboratory* (AFRL/SNHS) criaram uma base de dados (Defense Advanced Research Projects Agency, 2008) para estudos de caso de vários tipos de invasões que uma rede ou computador estão sujeitos, todos os ataques registrados nessa base de dados já se encontram documentados para estudos.

Todo o sistema de detecção de intrusão (IDS) é autorizado a utilizar a gama completa de dados do DARPA/AFRL, com esses dados já documentados e seus ataques todos conhecidos, desta forma, os IDSs podem iniciar uma série de testes para verificar sua eficiência. Além disso, testes *off-line* podem medir atributos de um sistema IDS que são difíceis de conhecer em uma avaliação em tempo real, tais como a latência de detecção e desempenho sob estresse.

Com a disponibilidade desta base de dados, e todo o estudo já existente sobre ela, é possível testar se o IDS está com uma boa detecção de anomalias, ou seja, detectando o maior número de anomalias com um baixo índice de falsos positivos.

2.3.2 IAS

PROBES são sondas que passivamente acessam o tráfego de rede coletando informações e armazenando em banco de dados (POHLMANN; PROEST, 2006). Para o desenvolvimento do trabalho, uma outra alternativa foi coletar os contadores da quanti-

dade de pacotes de conexão trafegados na rede usado um sistema de análise de Internet - IAS (POHLMANN; PROEST, 2006). Esse PROBE foi disponibilizado pela universidade alemã *Fachhochschule Gelsenkirchen* (FHGe), graças a uma parceria com a UFSM, fortalecendo os laços com a FHGe.

O PROBE instalado no *backbone* principal da rede da Universidade Federal de Santa Maria faz a coleta e análise dos pacotes que trafegam na rede da Universidade. Esse sistema possui um banco de dados no qual armazena as informações coletadas, esse banco de dados chama-se *db_ias*. A intenção do estudo dessa base de dados é identificar e usar os contadores de alguns protocolos de rede, a fim de encontrar possíveis anomalias no tráfego da rede.

2.4 Resumo do capítulo

Este capítulo apresentou uma revisão bibliográfica sobre os principais métodos utilizados pelos *crackers* para invadir ou perturbar o funcionamento normal de uma rede de computadores. Depois da leitura pode-se dizer que os ataques do tipo DoS ainda são uma grande ameaça aos sistemas computacionais atuais, por não existir ainda métodos totalmente eficazes de tratar tais ataques. Métodos do tipo *scan*, também ainda representam riscos, por revelar ao atacante importantes informações sobre a rede.

Embora não haja métodos preventivos eficientes contra ataques *DoS* e *Port Scan*, tais métodos possuem características marcantes, as quais se encaixam no perfil desejado para o trabalho, pois ambas alteram o tráfego normal da rede gerando picos. Deste modo, é permitido que sejam identificados com relativa facilidade no momento em que estejam acontecendo, permitindo uma intervenção precisa. Isto justifica o estudo de formas de ataques e detecção de intrusão.

Vimos também neste capítulo que os IDSs são ferramentas de segurança para redes de computadores que tem por objetivo identificar intrusões ou tentativas de intrusões em uma rede ou em um *host*, analisando os seus estados através da coleta de informações. A maioria dos IDSs atualmente utilizam métodos baseados em assinaturas de ataques. A principal desvantagem de um IDS baseado em assinaturas é a sua dificuldade em reconhecer ataques novos ou que ainda não estejam na sua base de dados. Detectores de Intrusão baseados em anomalias surgiram como forma contornar esta limitação dos IDSs baseados em assinaturas.

No final do capítulo, apresentamos o histórico das duas bases de dados utilizadas neste trabalho, a primeira, se faz necessária para que pudesse ser feito os testes e ajustes com uma base confiável, que tivesse todos os seus ataques já documentados. A segunda se faz necessária para que o nosso trabalho de detecção de intrusão através de séries temporais e filtro de alarmes através de *wavelets* possa ser comparado com o trabalho que a universidade alemã *Fachhochschule Gelsenkirchen* (FHGe) vem desenvolvendo na detecção de intrusão através de redes neurais. Mais detalhes sobre os ataques presentes e coletas dos dados serão abordados no capítulo 4.

A Detecção de Anomalias é uma área de estudo recente, e como tal, ainda apresenta problemas relacionados ao nível elevado de falsos positivos. Também não se possui a indicação de métodos mais eficientes ou mais completos para a detecção de anomalias em redes de computadores. Por causa de tais problemas, muitas vezes um IDS pode ser desacreditado ou exigir uma intervenção muito grande por parte do administrador.

Durante o decorrer do trabalho, será mostrado um estudo mais aprofundado quanto aos métodos empregados na detecção de anomalias em redes de computadores, realizando um comparativo entre os métodos existentes, analisando seus pontos fortes e suas falhas. Isto será útil para podermos escolher conscientemente métodos de detecção de anomalias mais eficientes, ou técnicas para a filtragem desses falsos positivos. O objetivo final é a construção de um Detector de Intrusão de Redes de Computadores baseado em Anomalias.

3 SÉRIES TEMPORAIS E *WAVELETS*

Neste capítulo será apresentada a base teórica matemática na qual se fundamenta o trabalho de detecção de intrusões baseada em séries temporais com filtragem dos alarmes através de *wavelets*.

3.1 Séries Temporais

Uma Série Temporal pode ser definida como uma seqüência de dados, capturados no tempo, que possuem uma forte relação com o seu passado (WHEELWRIGHT; MAKRIDAKIS, 1985). Desde muito tempo, análises de séries temporais são utilizadas em diversas áreas, como bolsa de valores, eletrocardiogramas, previsões do tempo (KIM et al., 2006), dentre outros. Esta análise permite obter previsões e elaborar cenários úteis na tomada de decisões. A idéia de utilização para detecção de intrusões é uma abordagem recente, sendo assim requer muitos cuidados na escolha dos modelos a serem utilizados e na implementação.

3.1.1 Características

Pelo fato das Séries Temporais trabalharem com dados de determinado instante de tempo, que dependem de dados de momentos anteriores, faz-se necessário o uso de técnicas específicas que levem em conta a ordem temporal dos fatos (BOWERMAN; O'CONNELL, 1993). A seleção de uma técnica adequada muitas vezes é dificultada devido à necessidade de ser efetuada uma análise do que se deseja como resultado e de como os dados capturados se comportam. Deste modo, existem diversas técnicas, mas para que se obtenha o resultado esperado é fundamental um estudo prévio de cada caso (EHLERS, 2005).

3.1.2 Modelo ARIMA

O modelo Autoregressivo (AR) é dado pela seguinte equação 3.1:

$$\chi_t = \phi_1 \chi_{t-1} + \phi_2 \chi_{t-2} + \ldots + \phi_p \chi_{t-p} + \epsilon_t \quad (3.1)$$

Onde o χ_t corresponde à observação da série temporal no tempo t; ϕ_p corresponde ao parâmetro do modelo AR de ordem p e ϵ_t representa o erro de eventos aleatórios que não podem ser explicados pelo modelo.

Caso as observações da série temporal possam ser representadas pela equação (3.1), e a ordem do modelo puder ser determinada e os parâmetros estimados, é possível prever o valor futuro da série em análise (WHEELWRIGHT; MAKRIDAKIS, 1985). Já o modelo de Médias Móveis (MA) fica definido conforme equação 3.2:

$$\chi_t = \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \ldots - \theta q \epsilon_{t-q} \quad (3.2)$$

onde ϵ_t representa o erro de eventos aleatórios que não podem ser explicados pelo modelo e θ_q corresponde ao parâmetro do modelo MA de ordem q.

A equação (3.2) é parecida à equação (3.1), exceto pelo fato de que o valor previsto para a observação depende dos valores dos erros observados em cada período passado, ao invés das observações propriamente ditas (WHEELWRIGHT; MAKRIDAKIS, 1985). Wheelwrigth e Makridakis especificaram o modelo misto Autoregressivo e de Médias Móveis (ARMA) através da equação (3.3), como sendo a combinação dos modelos AR e MA.

$$\chi_t = \phi_1 \chi_{t-1} + \phi_2 \chi_{t-2} + \ldots + \phi_p \chi_{1-p} + \epsilon_t - \theta_1 \epsilon_{t-1} - \theta_2 \epsilon_{t-2} - \ldots - \theta_q \epsilon_{t-q} \quad (3.3)$$

Analisando a equação (3.3) é possível verificar que os modelos ARMA relacionam os valores futuros com as observações passadas, assim como também com os erros passados apurados entre os valores reais e os previstos. Deste modo, foi escolhido para trabalhar com detecção de intrusões o modelo Autoregressivo de Médias Móveis Integrado - *Autoregressive Integrated Moving Average* (ARIMA) - para produzir a série temporal dos dados capturados (LUNARDI et al., 2008). Este tipo de modelo é usado para trabalhar com dados aleatórios de uma série estacionária ou não estacionária (EHLERS, 2005). Séries estacionárias representam processos com a variância e covariância que ficam em torno de uma média, isto é, os dados se comportam de forma mais equilibrada. Já séries

não estacionárias representam dados com que não possuem uma aproximação de valores entre as amostras, podendo variar abruptamente. Séries sazonais referem-se a existência de periodicidade dos dados, isto é, tendem a possuir repetições de comportamento durante o tempo (TRAN; REED, 2001).

Com o uso de Séries Temporais, mais especificamente com o modelo ARIMA, foi estabelecido um padrão de comportamento do tráfego de rede, e este padrão é avaliado a cada amostra para verificar se a série está se comportando como esperado (NUNES, 2003). Caso a série temporal se comporte de forma diferente do esperado, pode indicar o início do acontecimento de um ataque. Esta abordagem foi implementada e apresentou problemas de falsos positivos (DALMAZO et al., 2008). Como uma solução a este problema, este trabalho implementa uma análise dos alarmes gerados pela série temporal através de *wavelets*, desta forma pode-se diminuir os falsos positivos e gerar resultados mais confiáveis.

3.2 *Wavelets*

Wavelet é uma função matemática capaz de decompor uma função no domínio do tempo em diferentes escalas, de modo que seja possível uma análise da função nos domínios da freqüência e tempo (STRANG, 1993). A *Transformada de Wavelet* é a representação de uma função por meio de *Wavelets*. A análise através desta técnica é feita pela aplicação sucessiva da *Transformada de Wavelet*, representando a decomposição do sinal original em diversos componentes localizados no tempo e na freqüência. Cada *wavelet* possui melhor ou pior localização nos domínios da freqüência e do tempo, por isso a análise pode ser feita com diferentes *wavelets*, de acordo com o resultado desejado. Algumas aplicações de *wavelets* são: a identificação de picos em sinais cardíacos (KOZAKEVICIUS et al., 2005), compressão de imagens (USEVITCH, 2001), reconhecimento da fala (XU et al., 2007) e filtragem de sinal (WANG, 2008).

3.2.1 Aplicações

No trabalho sobre identificação de picos em sinais cardíacos é aplicada a *Transformada de Wavelet* para a análise de sinais cardíacos. Ela é utilizada em conjunto com a estratégia de *threshold* SURE (*Stein's Unbiased Risk Estimator*) para a filtragem do sinal, eliminando ruídos causados por interferências ou problemas na coleta. A *Transformada*

Wavelet de Haar foi utilizada para identificar a localização dos picos QRS no sinal cardíaco.

Em (HUANG; THAREJA; SHIN, 2006) é apresentado um estudo comparativo entre várias bases *wavelets* na detecção de anomalias em um IDS baseado em rede (NIDS).

As métricas utilizadas para a avaliação do trabalho de Chin-Tser Huang foram o desvio percentual e a entropia. O desvio percentual representa a variação percentual padrão dos coeficientes em relação à média. Pode-se considerar a melhor base aquela que possuir os maiores valores para o desvio percentual no início e no final de uma anomalia, contrastando com o tráfego normal e tornando a identificação possível. A entropia mede a desordem de um sistema. Não foi realizado um estudo com outros valores para a taxa de amostragem e o tamanho da janela de detecção.

Em (GAO et al., 2006) é apresentado um sistema para a detecção de anomalias baseado em *Wavelet Packet*. O sistema utiliza a *Transformada Wavelet Packet* no tráfego de rede a ser analisado e realiza a detecção de anomalias através de um algoritmo estatístico e um esquema de *thresholds* duplos. A *Transformada Wavelet Packet* proporciona um método mais preciso para a análise de sinal porque divide as bandas de freqüências em vários níveis e decompõe a banda das freqüências altas, o que não é realizado na *Transformada Wavelet*. Na *Transformada Wavelet* o sinal é dividido em uma banda de freqüências baixas e uma de freqüências altas, a banda de freqüências baixas por sua vez é dividida novamente em freqüências baixas e altas e assim sucessivamente até que se chegue a um único valor e vários níveis de coeficientes, que representam as freqüências existentes naquele nível. Na *Transformada Wavelet Packet* ambas as bandas de freqüências altas e baixas são processadas representando uma divisão de bandas de freqüências mais finas.

Da mesma forma que a *Transformada Wavelet de Haar* foi utilizada para identificar a localização dos picos QRS no sinal cardíaco, pretende-se adaptar os algoritmos baseados em *wavelets* utilizados por (KOZAKEVICIUS et al., 2005) na detecção de picos em sinais cardíacos para a detecção de picos e concentração de alertas gerados pelo DIBSeT (LUNARDI et al., 2008). Então, foi desenvolvido um módulo destinado a filtragem e melhor análise dos alarmes gerados pelo DIBSeT, que utiliza *wavelets* e técnicas estatísticas para alcançar os objetivos propostos. No próximo capítulo será apresentado como construir um segundo nível de análise na detecção de anomalias de rede para o DIBSeT integrado na saída do método baseado em Séries Temporais, como forma de evitar o ex-

cesso de alarmes e diminuir o número de falsas detecções.

3.3 Resumo do capítulo

Este capítulo apresentou a teoria matemática que está envolvida na detecção de intrusão baseada em Séries Temporais. Com esta análise é possível obter estimativas futuras e elaborar cenários úteis na tomada de decisões. Foram mostradas algumas áreas onde esta técnica é utilizada, como bolsa de valores, eletrocardiogramas e previsões do tempo.

Vimos também que *wavelet* é uma função matemática capaz de decompor uma função no domínio do tempo em diferentes escalas. Essa função é muito utilizada para compressão de imagens, identificação de picos em sinais cardíacos, reconhecimento da fala e filtragem de sinais. Este trabalho utiliza *wavelets* para filtragem dos níveis de alarmes gerados pelo detector de intrusão baseado em Séries Temporais.

4 PROPOSTA DO DIBSET-W

Este capítulo mostra como as bases de dados descritas no capítulo 2 foram usadas, o funcionamento do DIBSeT 1.0 elaborado por Lunardi (LUNARDI et al., 2008) e todas as demais versões criadas exclusivamente para o desenvolvimento deste trabalho. No final do capítulo é apresentado o algoritmo para o uso de *wavelets* na detecção de anomalias no tráfego da rede.

4.1 Histórico do DIBSeT

Esta seção descreve todas as fases do processo de melhoria do DIBSeT. Estas melhorias foram necessárias para que se pudesse comparar os resultados obtidos neste trabalho.

4.1.1 DIBSeT versão 1.0

O NTOP (DERI; SUIN; MASELLI, 2003), igualmente como o IAS, também é um programa para captura de tráfego de rede e armazenamento dos dados em contadores. É um *software* gratuito, disponível no site de seus mantedores. No DIBSeT, o NTOP foi utilizado seguindo uma arquitetura de detecção de intrusão baseada em rede, porém, o computador analisado estava em uma rede isolada, deste modo os contadores de pacotes correspondiam a apenas um computador.

A primeira versão do Detector de Intrusão Baseado em Séries Temporais implementado por Lunardi (LUNARDI et al., 2008), foi implementada tendo somente o NTOP como fonte de dados junto com classes implementadas pelo orientador deste trabalho durante sua Tese de Doutorado, como mostra na figura 4.1. Muitas destas funções e bibliotecas se utilizavam do programa RPS (DINDA, 2005). O DIBSeT, a cada 15 segundos, executa predições através das séries temporais somando uma margem de erro (estática), que pode ser positiva ou negativa. Também foi criada por Lunardi uma classe contendo

níveis de alarmes. Os níveis de alarmes, inicialmente, iam de -5 até +5, e eram atribuídos exponencialmente conforme a grandeza da diferença entre a predição e o valor real fornecido pelos contadores extraídos do NTOP. Todos os valores são armazenados para que seja possível a realimentação da série e a análise *off-line* dos dados. Com exceção do DIBSeT 1.0, todas as demais versões que serão apresentadas a seguir foram elaboradas e implementadas exclusivamente para este trabalho.

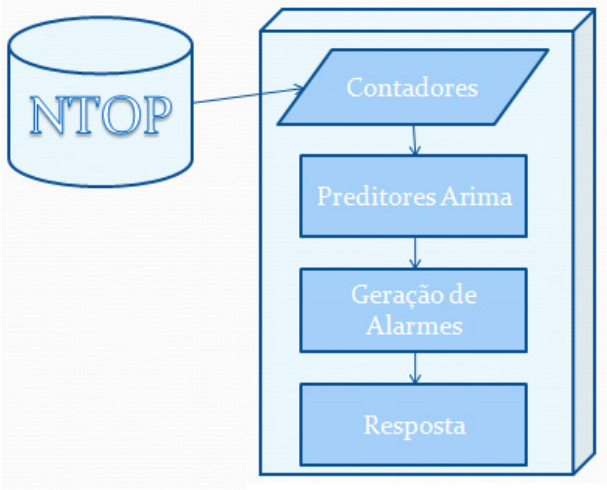

Figura 4.1: DIBSeT versão 1.0

4.1.2 DIBSeT versão 1.1

O primeiro passo para dar seguimento ao trabalho foi tornar o DIBSeT adaptável a qualquer outra base de dados, de modo que o programa possa ser alimentado por uma fonte genérica de dados. Além disso, essa mudança permite que o programa adquira a capacidade de trabalhar *off-line*. Esta característica é importante para que o sistema baseado em Séries Temporais possa ser "calibrado", gerando um número menor de falsos positivos no início de seu funcionamento. Com o DIBSeT funcionando de forma *off-line* torna-se possível a validação do algoritmo através de comparações com outros trabalhos publicados. A partir desta melhoria fica aceitável o uso da base de dados do DARPA, possibilitando a comparação com os demais IDSs.

Com o detector de intrusão baseado em séries temporais sendo alimentado por contadores provenientes de qualquer base de dados, notou-se a necessidade do aperfeiçoamento das gerações de alarmes. Os alarmes eram gerados de modo estático e proporcional à quantidade trafegada no computador no qual foi instalado o NTOP, ou seja, usando um tráfego algumas vezes maior foi percebido que os alarmes gerados não acompanhavam a

grandeza dos novos contadores. A solução encontrada foi a criação de uma nova classe de alarmes que aumentasse seus níveis proporcionalmente ao maior volume de tráfego dos novos contadores. A nova classe possui 10 níveis positivos e 10 níveis negativos, cada nível é calculado pela relação entre a predição e o valor real. Quanto maior é essa relação, maior é o nível do alarme. Esta alteração pode ser representada em alto nível pela figura 4.2. Com alarmes gerados de forma linear fica mais fácil a compreensão do que está acontecendo com o sistema.

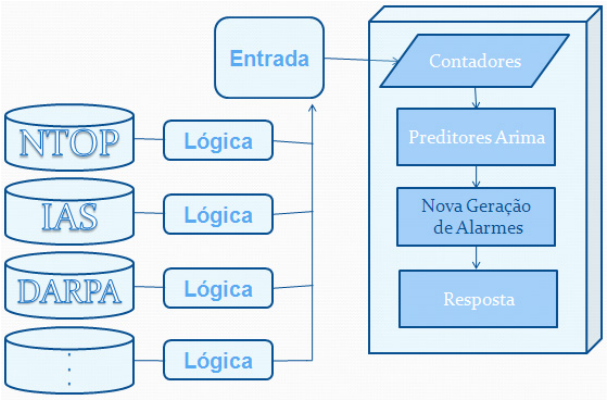

Figura 4.2: DIBSeT versão 1.1

4.1.3 DIBSeT versão 1.2

O DIBSeT possui margens de segurança em sua implementação. Essas margens têm a finalidade de dar um grau de liberdade para o cálculo dos alarmes, visto que o valor real do tráfego pode sofrer variações aleatórias não previstas pelo modelo ARIMA. Estas variações são compreendidas entre a predição mais um limite de margem e a predição menos um limite de margem, esses extremos são o limite superior e inferior, respectivamente. Todos os dados encontrados fora deste limite são considerados anomalias.

Depois de alguns testes, foi percebido que o DIBSeT tinha uma grande dificuldade em estabelecer corretamente os níveis de alarmes de acordo com o grau do ataque. Desta forma notou-se que o grande problema era encontrar uma margem adequada a ser estabelecida para cada protocolo estudado. Até então, o DIBSeT utilizava uma margem de segurança fixa. Nesta versão, foi substituída a classe geradora das margens fixas por outra classe que atualiza a margem a cada nova predição, tornando o limite inferior e superior mais confiável. Esta nova classe foi adaptada do trabalho desenvolvido na Tese de Doutorado do orientador deste trabalho (NUNES, 2003).

4.1.4 DIBSeT versão 1.3

Na versão 1.1 foram realizados testes com o DIBSeT original e a nova classe de alarmes. Na versão 1.2 voltou-se a utilizar os alarmes originais e então foram realizados testes somente com a nova classe de margem dinâmica. Na versão 1.3 do DIBSeT foram testadas em conjunto as melhorias da versão 1.1 e da versão 1.2, tornando possível as comparações e a percepção do quanto o sistema ficou mais eficiente. O próximo capítulo mostra com mais detalhes os ganhos na detecção dos ataques e a diminuição dos alarmes falsos.

4.2 DIBSeT-W

Como foi visto no capítulo anterior, uma Série Temporal é definida como um conjunto de amostragens de uma determinada variável, ordenadas no tempo, normalmente em intervalos eqüidistantes (WHEELWRIGHT; MAKRIDAKIS, 1985). Matematicamente pode ser representada como uma função discreta de Z:

$$Z(t) = \{Z_1, Z_2, Z_3, ...\} \quad (4.1)$$

Na Análise de Sinais uma Série Temporal pode ser representada como a soma de uma função determinística dependente do tempo $f(t)$ e um processo estocástico, r_t, ou ruído branco (gaussiano).

$$Z_t = f(t) + r_t \quad (4.2)$$

Onde Z_t representa o valor observado da variável Z no instante t. Assume-se que $f(t)$ independe de r_t e r_t independe de t.

A análise baseada em Séries Temporais busca encontrar o modelo, ou o polinômio, que melhor represente a função f(t) e estime os seus parâmetros componentes. Supondo que a função $f(t)$ seja difícil de ser estimada ou não viável do ponto de vista computacional, tem-se um erro na análise causado pelo erro na estimativa do modelo e pelo ruído que restou no modelo.

O preditor ARIMA do DIBSeT utiliza uma janela de análise finita e deslizante, na sua versão original ela foi definida com o tamanho de 200 (NUNES, 2003). Como a janela de análise é truncada pode-se dizer que dependências de longa duração no tráfego, maiores que a janela de análise, não possam ser modeladas corretamente pelo sistema

e que conseqüentemente o ruído gaussiano também não seja devidamente considerado. Considerando a existência de alguns erros na estimativa do modelo ARIMA pelo DIBSeT pode-se assumir a geração de alguns falsos alarmes.

A proposta de utilização de *wavelets* em um segundo estágio de um Detector de Anomalias baseado em Séries Temporais, o qual chama-se DIBSeT-W, vem da tentativa de se reduzir o número de falsos positivos.

Para a análise de alarmes baseada em *wavelets* pode-se adotar a seguinte definição:

$$A[t] = I[t] + r_t \text{ (4.3)}$$

Onde A é o sinal amostrado, neste caso os alarmes gerados pelo módulo baseado em Séries Temporais do DIBSeT, $I[t]$ é o valor correto do alarme que esta sendo procurado e r_t um ruído branco gaussiano residual.

A *Transformada Wavelet* decompõe um sinal dado em coeficientes *wavelets* e uma série de dilatações e translações de uma *wavelet* mãe Ψ (MALLAT, 1989).

$$Y = \sum_{i=-\infty}^{+\infty} C_{J,i}\varphi_{j,i} + \sum_{j=J}^{1}\sum_{i=-\infty}^{+\infty} d_{j,i}\Psi_{j,i} \text{ (4.4)}$$

Onde $C_{J,i}$ é a representação do sinal no nível mais grosseiro, $d_{j,i}$ são os detalhes, φ é a função escala e Ψ a função *wavelet*.

A decomposição de um sinal em aproximação a um conjunto de detalhes pode ser feita pela convolução do sinal dado na resolução anterior com filtros G (passa alta) e H (passa baixa) (MALLAT, 1989).

$$C_{2,j} = \sum_{k=-\infty}^{+\infty} h(2n-k)C_{2,j+1}Y \text{ (4.5)}$$

$$D_{2,j} = \sum_{k=-\infty}^{+\infty} g(2n-k)C_{2,j+1}Y \text{ (4.6)}$$

A filtragem de um dado sinal usando *wavelet* consiste no cálculo da *Transformada Wavelet Direta* seguido do corte dos valores nos detalhes considerados ruído e cálculo da *Transformada Wavelet Inversa* (DONOHO; JOHNSTONE, 1995).

$$\widetilde{l} = W^{-1}\overline{W}Y \text{ (4.7)}$$

Onde $\widetilde{l}$ é a estimativa da função I, W^{-1} é a *Transformada Wavelet Inversa* e $\overline{W}$ é a *Transformada Wavelet Direta* com corte do ruído.

Para a função de corte dos coeficientes pode-se utilizar o *Hard Thresholding* (DONOHO; JOHNSTONE, 1995) que consiste no corte dos valores menores que um determinado valor de corte t.

$$H(c) = \begin{cases} c, & |c| > t \\ 0, & |c| \leq t \end{cases} \quad (4.8)$$

Onde t representa o valor do *threshold* de corte. Este valor pode ser encontrado utilizando-se a estratégia de *threshold universal* por nível (DONOHO; JOHNSTONE, 1995) definada como:

$$t = \sqrt{2logn\delta^2} \quad (4.9)$$

Onde δ^2 é estimativa da variância do ruído.

O módulo *wavelet* do DIBSeT-W realiza a filtragem dos alarmes gerados pelo módulo anterior baseado no modelo ARIMA, utilizando a filtragem baseada em *wavelet* como definido por (DONOHO; JOHNSTONE, 1995), usando *hard thresholding* e *threshold universal*. A função *wavelet* escolhida foi a *Haar*, por ser computacionalmente mais simples e o valor da variância do ruído foi definido manualmente, nos testes realizados o valor 1.0 foi considerado ótimo. Foi considerada uma janela de análise deslizante de 256 valores. Desta forma realiza-se um corte dos valores dos coeficientes *wavelet* no nível mais grosseiro como forma de evidenciar os detalhes, onde estão as variações. O algoritmo completo pode ser descrito da seguinte forma:

```
1   INÍCIO
2      Enquanto existir novo_valor faça
3      {
4         Adiciona novo_valor na janela deslizante;
5         Atualiza a janela de análise deslizante;
6         Transformada Wavelet Direta do vetor de análise;
7         Cálculo do valor do threshold;
8         Filtragem do ruído por hard thresholding;
9         Corte do nível mais grosseiro do sinal;
10        Transformada Wavelet Inversa do vetor de análise;
11        Saída do novo nível de alarme correspondente;
12     }
13  FIM
```

O problema encontrado no DIBSeT foi o grande número de falsos positivos (DALMAZO et al., 2008). Desta forma foi implementado um sistema de refinamento dos alarmes gerados através de *wavelets*. Através dessa filtragem, obteve-se bons resultados em comparação com as outras versões do DIBSeT, tais resultados serão apresentados no próximo capítulo. A linguagem usada para a implementação do DIBSeT-W foi o Java (JAVA Technology, 2008). Vários foram os motivos que levaram à escolha desta linguagem de programação, entre eles destacam-se: grande portabilidade; possui uma extensa biblioteca de rotinas que facilita a cooperação com protocolos TCP/IP, HTTP e FTP; maior segurança em execuções remotas; possui vasta documentação disponível; e para permitir um melhor reaproveitamento de código, pois o DIBSeT versão 1.0 foi criado em Java. Uma arquitetura em alto nível do detector de intrusão baseado em séries temporais e *wavelets* pode ser mostrada na figura 4.3.

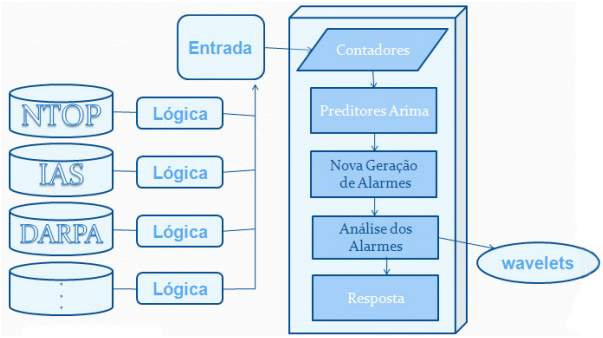

Figura 4.3: Arquitetura DIBSeT-W

4.3 Alarmes-W

O DIBSeT-W não gera alarmes próprios. Como a filtragem através de *wavelets* é executada sobre os alarmes originários das séries temporais e armazenado na janela deslizante, ele somente fará cálculos relativos aos alarmes de entrada, ou seja, após a *Transformada Wavelet Direta* são estabelecidos o *threshold* e a variância do ruído. Depois da filtragem através do *hard thresholding*, variações normais dos níveis dos alarmes da rede são subtraídos do sinal original. Novamente o sinal é reconstruído através da *Transformada Wavelet Inversa*. O valor retornado pela *Transformada Wavelet Inversa* corresponde ao nível do alarme sem as oscilações consideradas normais, neste caso, somente as anomalias.

4.4 Resumo do capítulo

Neste capítulo foi mostrado como as bases de dados descritas no segundo capítulo foram usadas para prover os dados necessários aos testes. Também foi apresentada a primeira versão do DIBSeT (LUNARDI et al., 2008) e todos os demais processos envolvidos ao longo de suas melhorias. Este capítulo mostrou como o DIBSeT-W trabalha na geração de seus alarmes. E o algoritmo usado neste trabalho para fazer a filtragem dos alarmes baseado em *wavelets* foi proposto e uma arquitetura em alto nível do DIBSeT-W foi apresentada.

5 DESENVOLVIMENTO

Este capítulo mostra as comparações entre os resultados obtidos pelo DIBSeT-W e as demais versões do DIBSeT.

5.1 Dados usados

Nesta seção será descrito como foram utilizadas as duas bases de dados apresentadas no capítulo 2.

5.1.1 Utilização do DARPA

5.1.1.1 Descrição da base de dados

A base de dados é composta por 5 semanas, contendo 5 dias cada semana. A base contém cerca de 9 Gb de dados, e os dados são compostos por:

- Saída da rede gerada pelo *tcpdump*;
- Entrada da rede gerada pelo *tcpdump*;
- Auditoria do Solaris BSM;
- Auditoria NT;
- *Dumps* dos diretórios selecionados;
- Sistema de arquivos.

Para nosso caso de estudo, foram usados somente dados de saída e dados de entrada da rede do *Lincoln Laboratory* e *Air Force Research Laboratory*, capturados pelo programa *tcpdump*, que é um dos mais conhecidos, se não o mais conhecido, *sniffer* para sistemas GNU/Linux (TCPDUMP - http://www.tcpdump.org, 2008). Das 5 semanas disponibilizadas, as 3 primeiras são chamadas de dados de treinamento e contém seus ataques

documentados, as semanas 4 e 5 são chamadas de dados de teste e seus ataques não encontram-se documentados.

Para o trabalho de análise do tráfego de rede e alarmes através do DIBSeT-W, foram usadas as semanas 1, 2 e 3. Os principais motivos que nos levaram a usar esta base de dados foram:

- Conhecimento prévio e documentação de todos os ataques compreendidos nesse espaço de tempo;
- Medir a eficácia de um IDS na presença de um comportamento intrusivo na atividade entre um computador e a rede;
- Medir a eficácia dos mecanismos de resposta;
- A possibilidade de comparação com outros métodos de detecção já conhecidos.

Os tipos de ataques abordados pela base de dados do DARPA são:

- DoS (*Denial of service*);
- Acesso de uma máquina remota sem autorização;
- Transmissão de privilégios sem autorização entre usuário ROOT a um usuário comum;
- Comportamento anômalo de usuários.

O tempo médio de duração de cada ataque na base de dados do DARPA é de aproximadamente 10 segundos, a figura 5.1 mostra um ataque de *Denial of Service* ocorrido no terceiro dia da segunda semana de treinamento. Este ataque possui cerca de 8 segundos de duração, iniciando no segundo 14536 e sendo finalizado no segundo 14544.

O foco deste trabalho é abordar ataques que geram variações anormais no volume padrão do tráfego de rede.

5.1.1.2 Avaliação

Abaixo segue uma breve descrição de configuração e condução dos testes, para que possamos obter uma noção de como são impostas as condições que o IDSs foi testado.

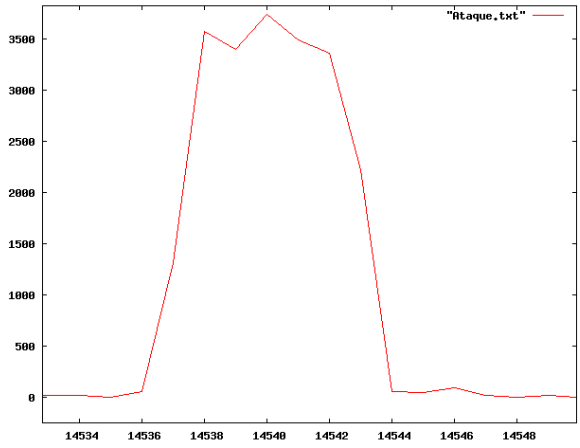

Figura 5.1: Duração de um ataque em segundos

- Ambiente: Cada avaliação foi executada em um ambiente controlado. Os testes foram isolados de qualquer outra rede. Os dados não estão absolutamente ordenados, fato natural, visto que não é possível assegurar que pacote por pacote fique registrado com exatidão a cada tempo, uma vez que essas pequenas variações em uma rede complexa são obrigadas a produzir alguma variação temporal, nem que seja mínima, no final dos eventos. Casualmente, essas variações podem produzir um número de falsos alarmes no decorrer da execução, mas as técnicas de medição deverão ser desenvolvidas com a tentativa de minimizar os possíveis efeitos deste problema de ordenação.

- Topologia da rede: A rede do DARPA/AFRL possui múltiplos subdomínios localizados atrás do *firewall*. Os testes foram executados com uma grande variedade de plataformas:

  - Sun Solaris 5.5/SunOs 4.x
  - IBM AIX 2.5
  - HP HP-UX
  - PC NT/Linux

Para todos os sistemas citados acima há tentativas de ataques documentados.

- Tráfego normal: O tráfego normal foi gerado por *scripts* que inicializam vários serviços, e são similares aos usados na avaliação dos dados de todos dos dias coletados durante as semanas de treinamento. Um vasto leque de serviços são representados,

tais como conexões TELNET, FTP, SMTP, HTTP, TCP, e UDP, e foram inicializados e encerrados a partir de cada ponto da rede.

- Tráfego intrusivo: Comportamentos intrusivos foram colocados, conforme as classes de ataques descritas anteriormente. Todos os ataques são conhecidos na literatura, e muitos destes ataques são apresentados múltiplas vezes com uma variação no grau de sua intensidade. Os ataques foram executados tanto de fontes internas, como também de fontes externas.

5.1.1.3 Dados de treinamento

Os dados originais do DARPA/AFRL estão no formato de saída do *tcpdump*, desta forma, a ferramenta *tcpstat* (TCPSTAT - http://www.frenchfries.net/paul/tcpstat/, 2008) foi usada para gerar os contadores no formato de *logs*, necessários para a detecção das anomalias. Depois de descompactado o arquivo, tem-se uma pasta, chamada wX, onde o X representa o número da semana. Dentro da pasta wX, existem 5 subpastas, numeradas de 1 até 5, cada uma representando um dia da semana. Neste trabalho foram usados pacotes contendo o tráfego total e também, individualmente, os protocolos TCP e ICMP. Por exemplo, a geração dos contadores TCP com a duração de 60 segundos, a partir do *tcpstat* foi gerada conforme o esquema abaixo:

tcpstat -r outside.tcpdump -o "\%T\n"60 > w1-d1-out-60-tcp.data

Após este comando, nós temos todos os arquivos de *log* TCP individuais, referentes aos dias de todas as semanas. Mas é preciso concatenar todos os dias de uma semana em um único arquivo, e para isso, foi usada a seguinte linha de comando:

cat w1/d1/w1-d1-in-60-tcp.data w1/d2/w1-d2-in-60-tcp.data w1/d3/w1-d3-in-60-tcp.data w1/d4/w1-d4-in-60-tcp.data w1/d5/w1-d5-in-60-tcp.data > w1/fullw1-in-60-tcp.data

Um exemplo de como fica o gráfico do protocolo TCP durante a primeira semana completa esta estampado na figura 5.2.

Para gerar um único arquivo, contendo todos os dados das semanas de treinamento, foram concatenadas as 3 primeiras semanas da seguinte forma:

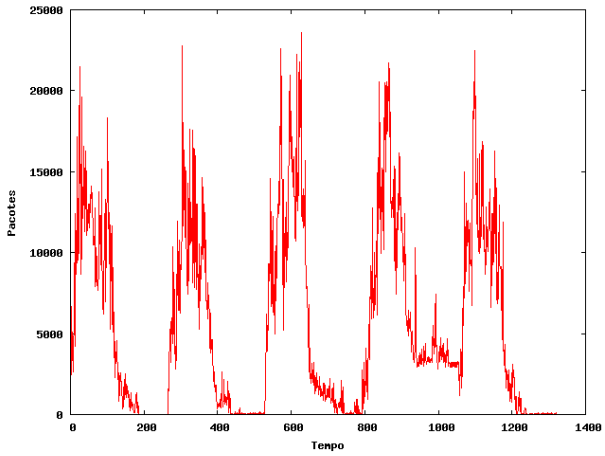

Figura 5.2: Uma semana de dados DARPA

cat w1/fullw1-out-60-tcp.data w3/fullw3-out-60-tcp.data w2/fullw2-out-60-tcp.data > fullweeks-out-60-tcp.data

O gráfico gerado a partir do arquivo *fullweeks-out-60-tcp.data* fica como mostra a figura 5.3.

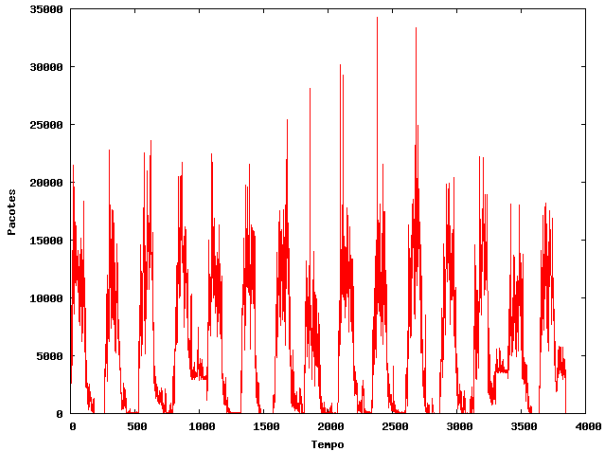

Figura 5.3: Três semanas de dados DARPA

Dados das semanas de treinamento

A simulação normal do tráfego de rede coletou dados durante 22 horas por dia. O programa *tcpslice* foi usado para examinar a saída dos dados do *tcpdump* e extrair os tempos do primeiro e último pacote. Durante a primeira semana de treinamento a simulação dos testes não teve nenhum imprevisto. Durante a segunda semana de treinamento a simulação dos testes teve que ser suspensa no segundo dia às 03h por motivo de manutenção. Na coleta da terceira semana de treinamento a simulação dos testes precisou ser interrompida

por causa de manutenções imprevista na rede. As informações detalhadas com os dias e tempos de coletas podem ser vistas nas tabelas 5.1, 5.2 e 5.3.

Tabela 5.1: *Primeira semana*

Tempo do primeiro pacote			Tempo do último pacote		
Segunda	1 Mar	08:00:02	Terça	2 Mar	06:00:02
Terça	2 Mar	08:00:02	Quarta	3 Mar	06:00:01
Quarta	3 Mar	08:00:03	Quinta	4 Mar	06:00:01
Quinta	4 Mar	08:00:03	Sexta	5 Mar	06:00:02
Sexta	5 Mar	08:00:02	Sábado	6 Mar	06:00:02

Tabela 5.2: *Segunda semana*

Tempo do primeiro pacote			Tempo do último pacote		
Segunda	8 Mar	08:00:01	Terça	9 Mar	06:00:49
Terça	9 Mar	08:00:01	Quarta	10 Mar	02:59:59
Quarta	10 Mar	08:00:03	Quinta	11 Mar	06:00:01
Quinta	11 Mar	08:00:03	Sexta	12 Mar	06:00:00
Sexta	12 Mar	08:00:02	Sábado	13 Mar	06:00:00

Tabela 5.3: *Terceira semana*

Tempo do primeiro pacote			Tempo do último pacote		
Segunda	15 Mar	08:00:02	Terça	16 Mar	06:00:02
Terça	16 Mar	08:00:01	Quarta	17 Mar	06:00:01
Quarta	17 Mar	08:00:03	Quinta	18 Mar	06:00:01
Quinta	18 Mar	08:00:02	Sexta	19 Mar	04:11:44
Sexta	19 Mar	08:00:03	Sábado	20 Mar	01:02:46

Lista de ataques ocorridos na segunda semana de treinamento

A primeira e a terceira semana da fase de treinamento possuem um tráfego normal de rede, ou seja, não possuem nenhum tipo de ataque registrado neste período. A tabela 5.4 possui uma lista com os ataques que ocorreram na segunda semana da fase de treinamento. Esta tabela possui o identificador do ataque, a data, tempo inicial, endereço de origem do ataque, e nome do ataque.

A seguir apresenta-se a descrição de alguns dos ataques documentados no DARPA que podem ser identificados pelo DIBSeT-W.

Tabela 5.4: Lista com ataques DARPA

ID	$Data$	$Tempo$	$Origem$	$Nome$
1	08/03/1999	08:01:01	hume.eyrie.af.mil	NTinfoscan
2	08/03/1999	08:50:15	zeno.eyrie.af.mil	smurf
3	08/03/1999	09:39:16	marx.eyrie.af.mil	back
4	08/03/1999	12:09:18	pascal.eyrie.af.mil	httptunnel
5	08/03/1999	15:57:15	pascal.eyrie.af.mil	land
6	08/03/1999	17:27:13	marx.eyrie.af.mil	secret
7	08/03/1999	19:09:17	pascal.eyrie.af.mil	ps attack
8	09/03/1999	08:44:17	marx.eyrie.af.mil	portsweep
9	09/03/1999	09:43:51	pascal.eyrie.af.mil	eject
10	09/03/1999	10:06:43	marx.eyrie.af.mil	back
11	09/03/1999	10:54:19	zeno.eyrie.af.mil	smurf
12	09/03/1999	11:49:13	pascal.eyrie.af.mil	secret
13	09/03/1999	14:25:16	pascal.eyrie.af.mil	mailbomb
14	09/03/1999	13:05:10	172.016.112.001-114.254	ipsweep
15	09/03/1999	16:11:15	marx.eyrie.af.mil	dict
16	09/03/1999	18:06:17	pascal.eyrie.af.mil	httptunnel
17	10/03/1999	12:02:13	marx.eyrie.af.mil	satan
18	10/03/1999	13:44:18	pascal.eyrie.af.mil	mailbomb
19	10/03/1999	15:25:18	marx.eyrie.af.mil	perl (Failed)
20	10/03/1999	20:17:10	172.016.112.001-114.254	ipsweep
21	10/03/1999	23:23:00	pascal.eyrie.af.mil	eject (console)
22	10/03/1999	23:56:14	hume.eyrie.af.mil	crashiis
23	11/0311999	08:04:17	hume.eyrie.af.mil	crashiis
24	11/03/1999	09:33:17	marx.eyrie.af.mil	satan
25	11/03/1999	10:50:11	marx.eyrie.af.mil	portsweep
26	11/03/1999	11:04:16	pigeon.eyrie.af.mil	neptune
27	11/03/1999	12:57:13	marx.eyrie.af.mil	secret
28	11/03/1999	14:25:17	marx.eyrie.af.mil	perl
29	11/03/1999	15:47:15	pascal.eyrie.af.mil	land
30	11/03/1999	16:36:10	172.016.112.001-254	ipsweep
31	11/03/1999	19:16:18	pascal.eyrie.af.mil	ftp-write
32	12/03/1999	08:07:17	marx.eyrie.af.mil	phf
33	12/03/1999	08:10:40	marx.eyrie.af.mil	perl (console)
34	12/03/1999	08:16:46	pascal.eyrie.af.mil	ps (console)
35	12/03/1999	09:18:15	duck.eyrie.af.mil	pod
36	12/03/1999	11:20:15	marx.eyrie.af.mil	neptune

- *Back*: Ataque de negação de serviço contra o servidor Apache quando um cliente solicita uma URL que contenha muitas barras invertidas.

- *Dict*: Adivinhar senhas de um usuário válido com variantes do nome da conta, ao longo de uma conexão por SSH ou TELNET.

- *Land*: Negação de serviço, quando um *host* remoto envia vários pacotes UDP com a mesma origem e destino.
- *MailBomb*: Ataque de Negação de Serviço onde temos um grande envio de mensagens para entregar, com o intuito de travar ou limitar o funcionamento normal de um servidor.
- *Neptune*: Ataque *SYN Flood* para negação de um serviço em uma ou mais portas.
- *Smurf*: Negação de serviço através de *flood* em resposta a uma requisição ICMP.

5.1.2 Utilização do IAS

Para que se pudessem usar os dados do PROBE (POHLMANN; PROEST, 2006) no trabalho, foi necessário:

- Instalação e configuração do IAS;
- Estudo do PROBE disponibilizado pela parceira alemã;
- Identificação dos contadores do banco de dados necessários para a identificação das possíveis anomalias;
- Elaboração de algoritmos para o problema em questão;
- Implementação de uma API em Java (JAVA Technology, 2008) que integrasse o banco de dados com as classes de Séries Temporais (WHEELWRIGHT; MAKRIDAKIS, 1985);

5.1.2.1 Descrição da base de dados

Ataques que geram anomalias ocorrem através de atividades que superam limites estabelecidos. O problema deste tipo de detecção é o grande número de falsos positivos, isto é, classificação de comportamentos normais como ataques. Por outro lado, esta forma de detecção consegue captar tentativas de intrusões, mesmo que ainda não sejam conhecidas pela comunidade científica.

Observe que a partir dos dados coletados é possível deduzir características de um possível ataque. Outra característica importante do IAS é que as sondas de coleta de dados

apenas identificam os tipos de pacotes transferidos, abstendo-se de extrair qualquer informação relevante do *payload* do pacote. Este procedimento garante a proteção dos dados e a confidencialidade da informação. Portanto, o que basicamente é feito é a análise de picos de protocolos suspeitos que trafegam na rede, ou seja, os protocolos que sofrem alterações na sua quantidade quando a rede está sendo atacada. Para possibilitar a avaliação estatística do comportamento de tráfego, visando a detecção de anomalias, o IAS possibilita a amostragem dos contadores em intervalos regulares de tempo. Com o acesso aos contadores de pacotes pode-se extrair o contador de qualquer protocolo. A base de dados está disposta de forma contínua, e a cada 300 segundos ela salva um contador individual para cada protocolo. Um gráfico do protocolo TCP trafegado na rede pode ser observado na figura 5.4.

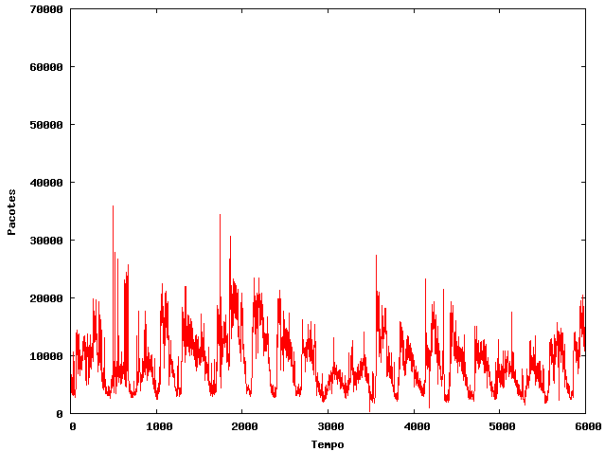

Figura 5.4: *Base Line IAS*

Desta forma, é possível estimar o "comportamento normal" do tráfego e realizar análises *off-line* ou em tempo real com intuito de detectar possíveis anomalias.

Para este trabalho, partiu-se a procura de anomalias em protocolos conhecidos, com grande exploração em artigos, o que foi de grande ajuda no auxílio a respeito dos resultados obtidos. Desta forma podem-se saber quais contadores são analisados e como deve ser o comportamento destes em casos de ataques.

5.2 Prova de conceito

Para fins de validação do algoritmo proposto neste trabalho, foram utilizados dados do terceiro dia da segunda semana da base de dados do DARPA. Estes dados foram extraídos na forma de contadores de pacotes a cada 60 segundos. Dentro deste espaço de tempo do

terceiro dia, foram consideradas 500 amostras contínuas. Nesta amostra está registrado um ataque de *Denial of Service*, o qual altera o tráfego normal do protocolo TCP. E também está registrado na mesma janela, um *Smurf Attack*, o qual altera os contadores dos protocolos ICMP. A razão de pegar uma janela de apenas 500 valores se dá ao fato de ter-se mais controle sobre os dados e anomalias estudadas, podendo gerar resultados mais consistentes.

5.2.1 Testes com DIBSeT versão 1.0

Os gráficos a seguir mostram dados de saída do DIBSeT versão 1.0 gerados a partir de dados TCP e ICMP.

5.2.1.1 Pacotes TCP

A figura 5.5 mostra o tráfego normal da rede com a predição e as margens inferior e superior fixas. A escala de tempo se refere ao tamanho da amostra em intervalos de 60 segundos, ou seja, a janela possui pouco mais que 8 horas. Note que tudo o que estiver entre as margens não são considerados anomalias.

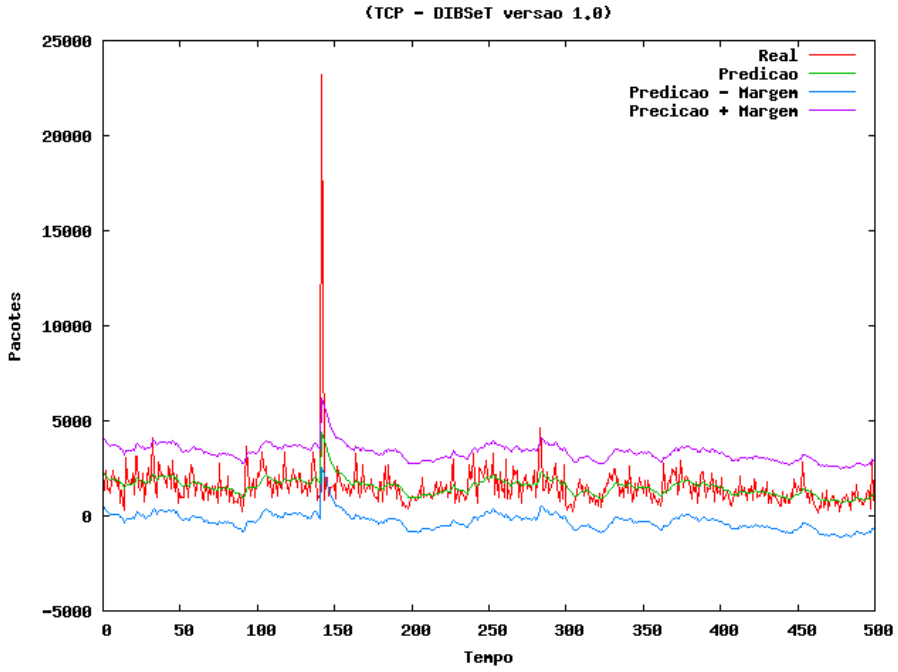

Figura 5.5: DIBSeT versão 1.0 com detalhes TCP

Os gráficos com detalhes sobre a margem fixa 5.6 (a), sobre o tráfego real 5.6 (b) e sobre os alarmes 5.6 (c) podem ser encontrados a seguir. Nota-se que houve um número exagerado de alarmes gerado pela verão 1.0 do DIBSeT. Este fato mostra o quanto é fragil

a geração original dos alarmes.

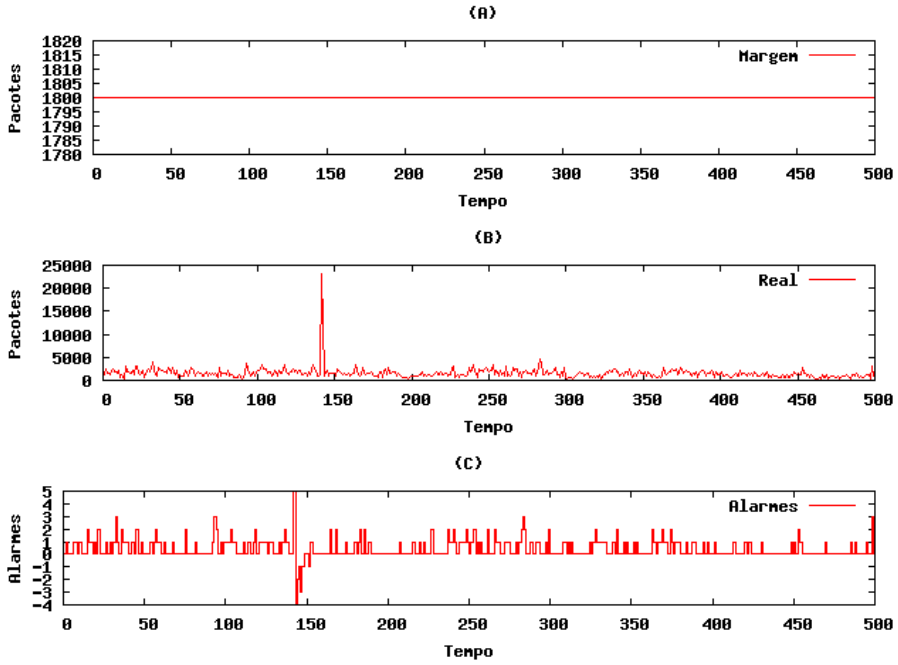

Figura 5.6: DIBSeT versão 1.0 Alarmes TCP

5.2.1.2 Pacotes ICMP

A figura 5.7 mostra o tráfego normal da rede com a predição e as margens inferior e superior fixas em 15 pacotes a cada minuto.

Os gráficos com detalhes sobre a margem fixa 5.8 (a), sobre o tráfego real 5.8 (b) e sobre os alarmes 5.8 (c), indicam que com a análise isolada sobre o tráfego ICMP o IDS melhorou um pouco, porém ainda é gerado muitos alarmes. A tabela 5.5 mostra um resumo indicando a quantidade de ataques registrados na base do DARPA e os ataques identificados pelo DIBSeT versão 1.0. Nota-se que a quantidade de alarmes gerado é muitas vezes maior que o número de ataques registrado.

Tabela 5.5: Quadro do DIBSeT versão 1.0

	TCP	**ICMP**
Alarmes	178	37
Ataques	1	1

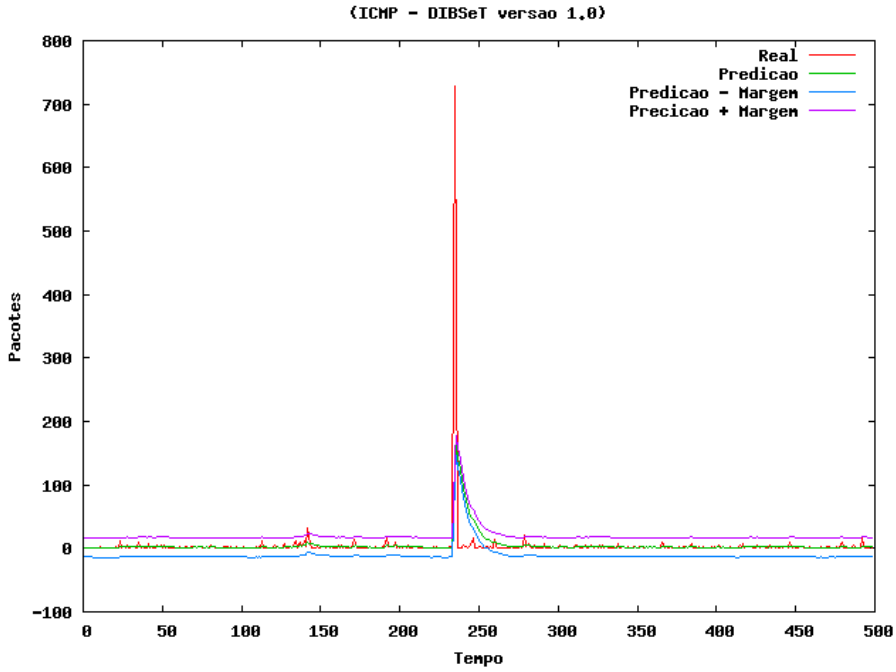

Figura 5.7: DIBSeT versão 1.0 com detalhes ICMP

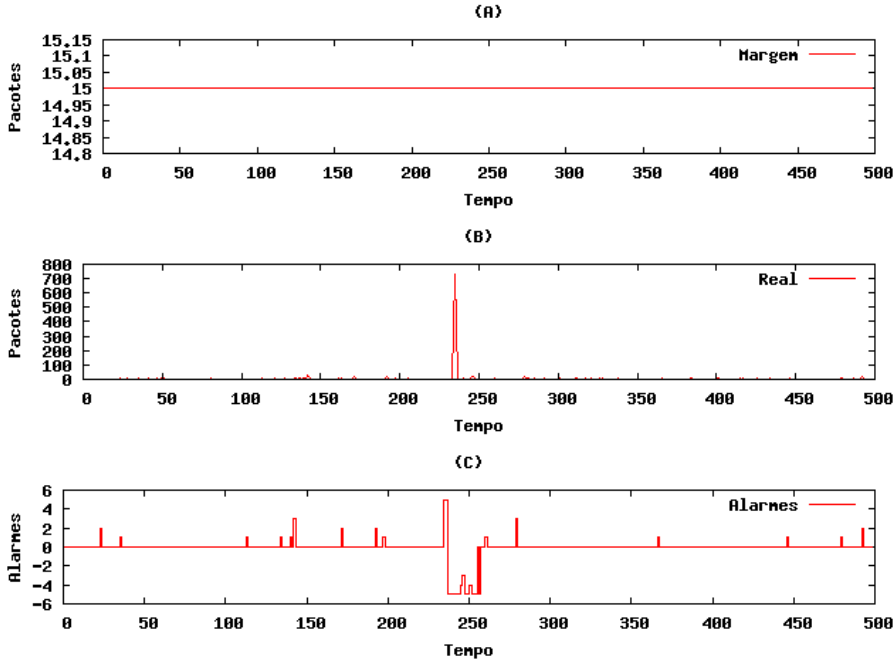

Figura 5.8: DIBSeT versão 1.0 Alarmes ICMP

5.2.2 Testes com DIBSeT versão 1.1

Os gráficos a seguir mostram dados de saída do DIBSeT versão 1.1 gerados a partir de dados TCP e ICMP.

5.2.2.1 *Pacotes TCP*

A figura 5.9 mostra o tráfego normal da rede com a predição, as margens inferior e superior fixa em 1800 pacotes por minuto. A escala de tempo com 500 amostras representa cerca de 8 horas de monitoramento contínuo do tráfego da rede. Todos os picos que estão além das margens são identificados como uma anomalia.

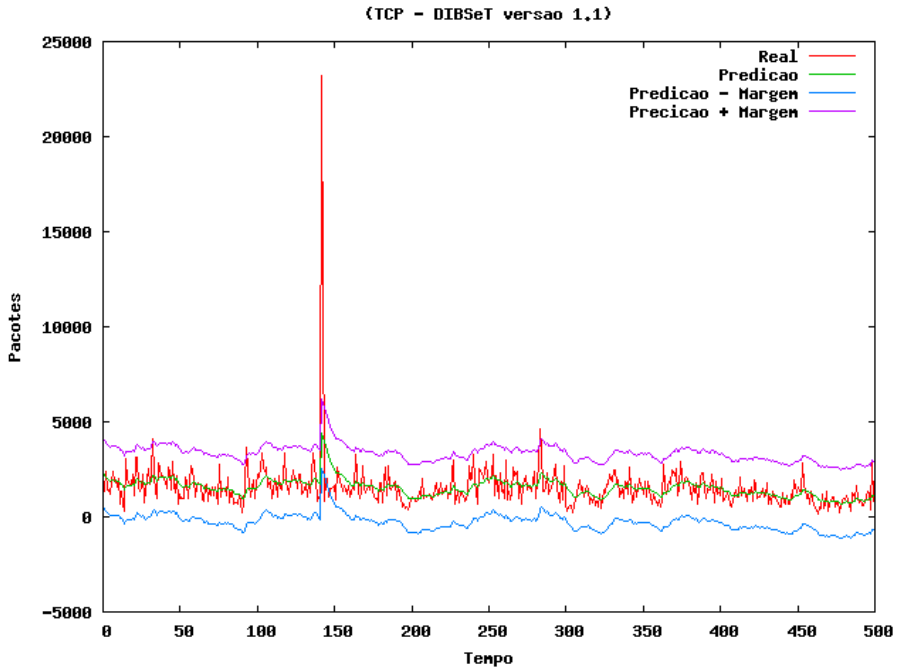

Figura 5.9: DIBSeT versão 1.1 com detalhes TCP

Os gráficos a seguir tratam com detalhes a respeito da margem fixa em 1800 pacotes por minuto 5.10 (a), do tráfego real 5.10 (b) e dos alarmes 5.10 (c).

5.2.2.2 *Pacotes ICMP*

A figura 5.11 mostra o tráfego normal da rede com a predição e as margens inferior e superior fixas em 15 pacotes por minuto.

Os gráficos 5.12 (a), 5.12 (b) e 5.12 (c), mostram os detalhes sobre a margem fixa, sobre o tráfego real e sobre os alarmes, respectivamente.

Ambas as versões 1.0 e 1.2 tiveram o mesmo número de anomalias detectadas corretamente porém o número de falsos positivos diminuíram muito. O bom resultado exibido nestas comparações com os protocolos TCP e ICMP é explicado pela escolha de um valor ótimo para a margem estática, este valor foi escolhido de maneira empírica, com sua escolha efetivada depois de testes com cerca de 20 valores diferentes, variando entre 500

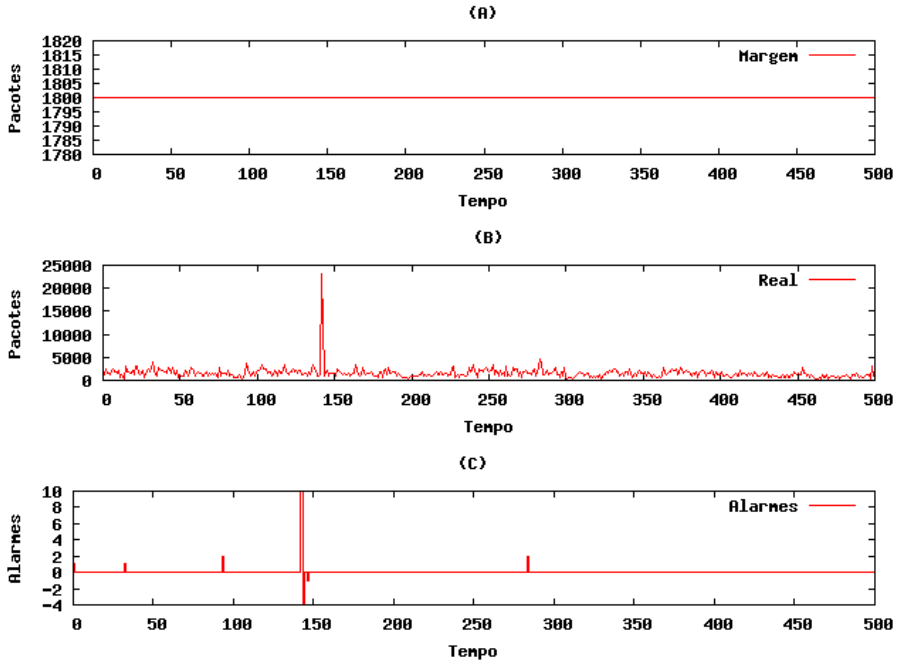

Figura 5.10: DIBSeT versão 1.1 Alarmes TCP

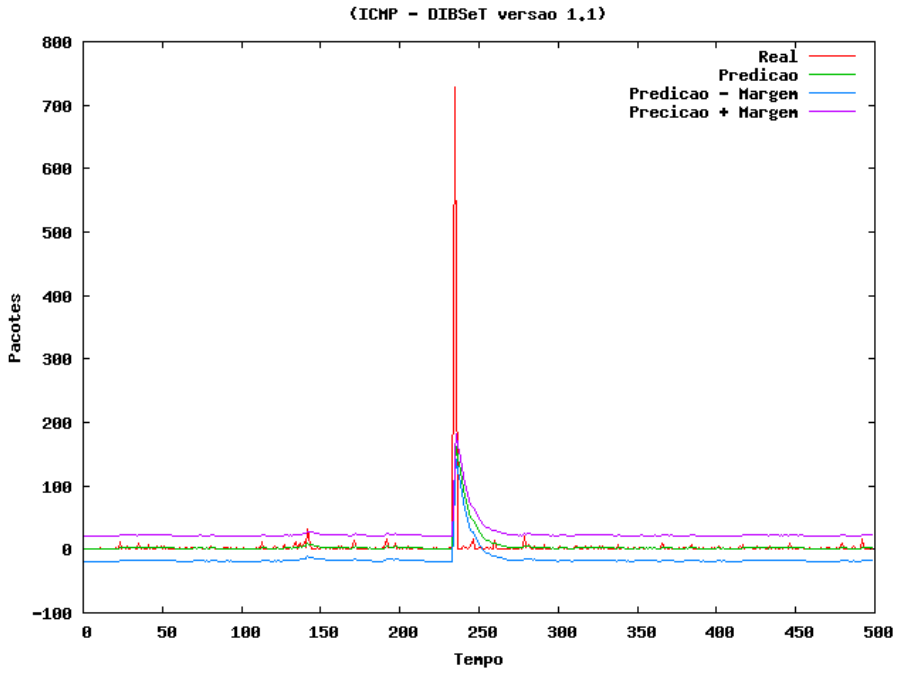

Figura 5.11: DIBSeT versão 1.1 com detalhes ICMP

e 3000 pacotes. E é importante lembrar que o valor ótimo muda de acordo com a grandeza dos valores analisados, sendo muito dispendioso encontrar um número ótimo por meios estáticos. A tabela 5.6 mostra um resumo com os ataques e alarmes detectados pelo DIBSeT versão 1.1.

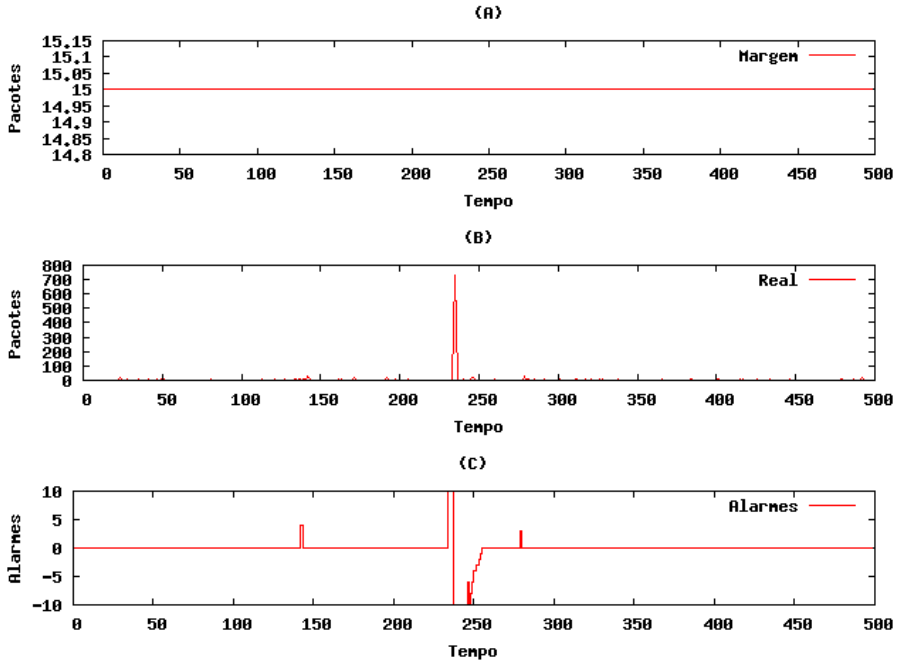

Figura 5.12: DIBSeT versão 1.1 Alarmes ICMP

Tabela 5.6: Quadro do DIBSeT versão 1.1

	TCP	**ICMP**
Alarmes	7	20
Ataques	1	1

5.2.3 Testes com DIBSeT versão 1.2

Os gráficos 5.13 e 5.15 mostram dados de saída do DIBSeT versão 1.2 gerados a partir de dados TCP e ICMP.

5.2.3.1 Pacotes TCP

A figura 5.13 mostra o tráfego normal da rede com a predição e as margens inferior e superior calculadas de modo dinâmico. A escala de tempo se refere ao tamanho da amostra em intervalos de 60 segundos, ou seja, a janela possui pouco mais que 8 horas. Pode-se visualizar neste gráfico que o intervalo de segurança ficou melhor adaptado ao tráfego real da rede.

Os gráficos com detalhes sobre a margem 5.14 (a), sobre o tráfego real 5.14 (b) e sobre os alarmes 5.14 (c) podem ser vistos a seguir.

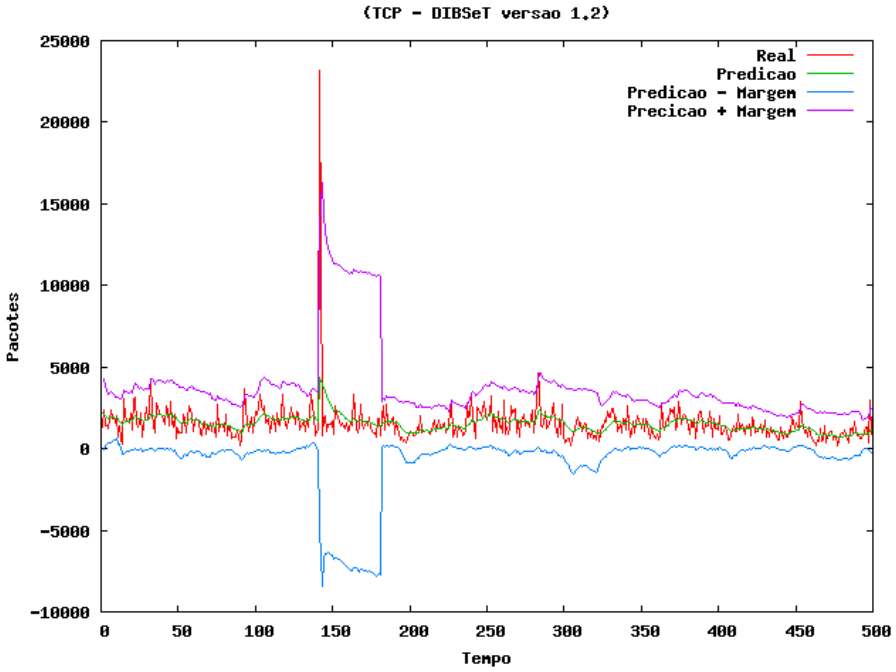

Figura 5.13: DIBSeT versão 1.2 com detalhes TCP

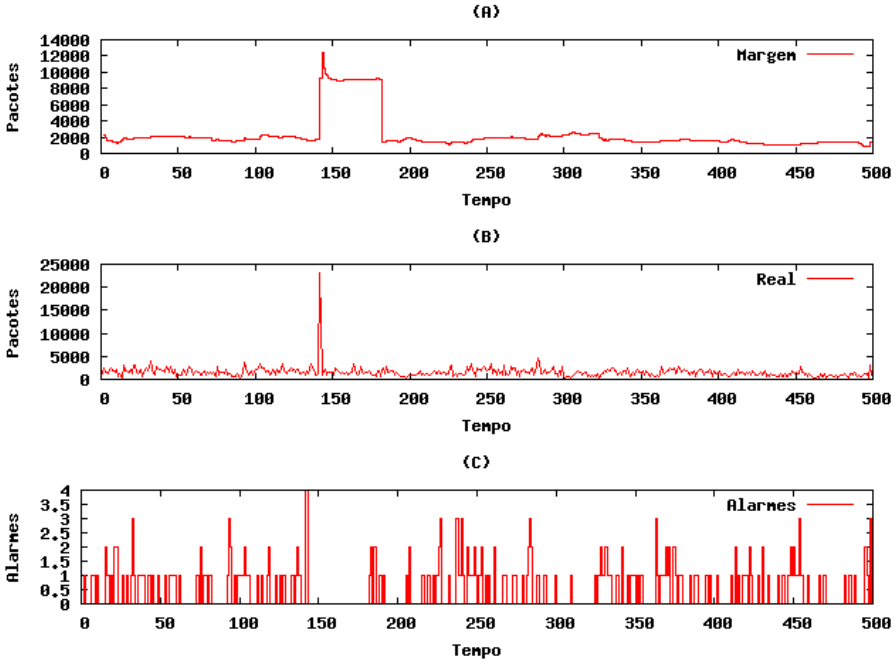

Figura 5.14: DIBSeT versão 1.2 Alarmes TCP

5.2.3.2 Pacotes ICMP

A figura 5.15 mostra o tráfego normal da rede com a predição e as margens inferior e superior dinâmicas. A escala de tempo com 500 amostras representa cerca de 8 horas de monitoramento contínuo do tráfego da rede.

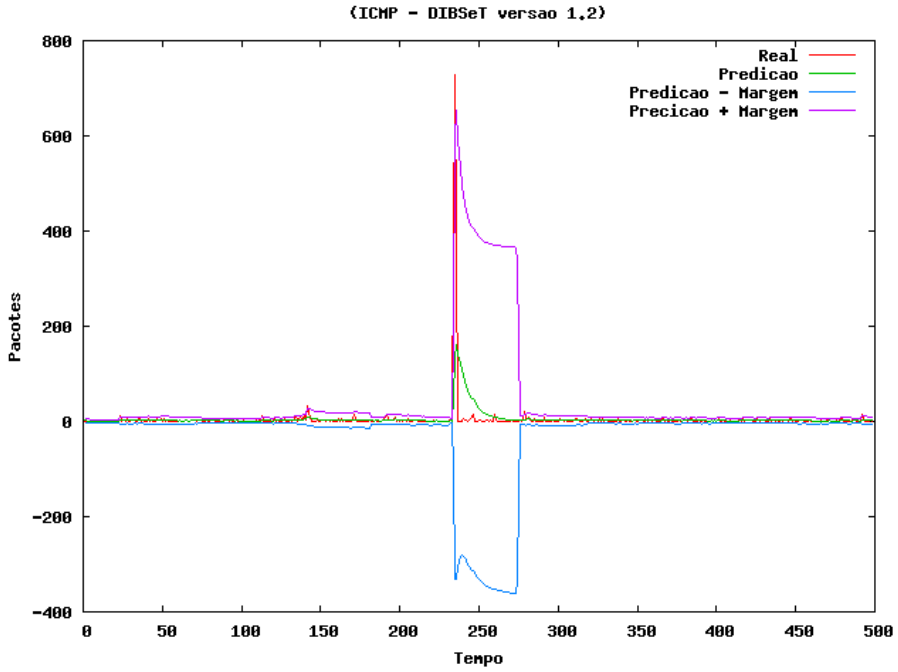

Figura 5.15: DIBSeT versão 1.2 com detalhes ICMP

Os gráficos com detalhes sobre a margem 5.16 (a), sobre o tráfego Real 5.16 (b) e sobre os alarmes 5.16 (c) podem ser vistos logo abaixo:

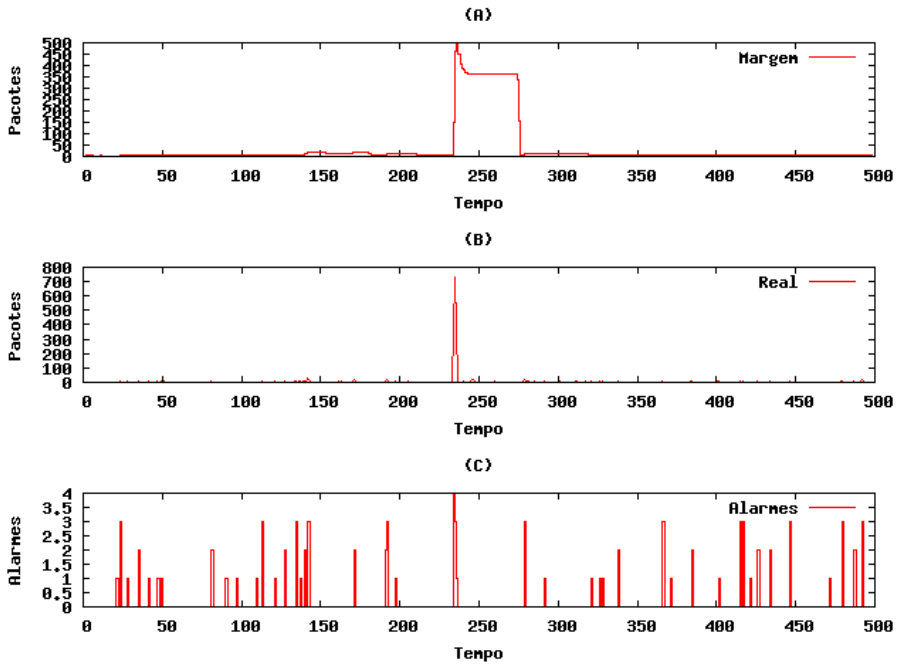

Figura 5.16: DIBSeT versão 1.2 Alarmes ICMP

A tabela 5.7 apresenta um breve resumo contendo números com os resultados obtidos através da análise das janelas dos protocolos TCP e ICMP com o DIBSeT versão 1.2.

O péssimo desempenho observado é atribuído ao uso da classe de alarmes do DIBSeT versão 1.0.

Tabela 5.7: Quadro do DIBSeT versão 1.2

	TCP	**ICMP**
Alarmes	186	45
Ataques	1	1

5.2.4 Testes com DIBSeT versão 1.3

Os gráficos 5.17 e 5.19 mostram dados de saída do DIBSeT versão 1.3 gerados a partir de dados TCP e ICMP, respectivamente.

5.2.4.1 Pacotes TCP

A figura 5.17 mostra o tráfego normal da rede com a predição e as margens inferior e superior calculadas de modo dinâmico junto com a nova classe de alarmes. A escala de tempo contém 500 amostras, cada amostra possui 60 segundos de duração, deste modo a janela de teste está compreendida em um período de aproximadamente 8 horas e 20 minutos.

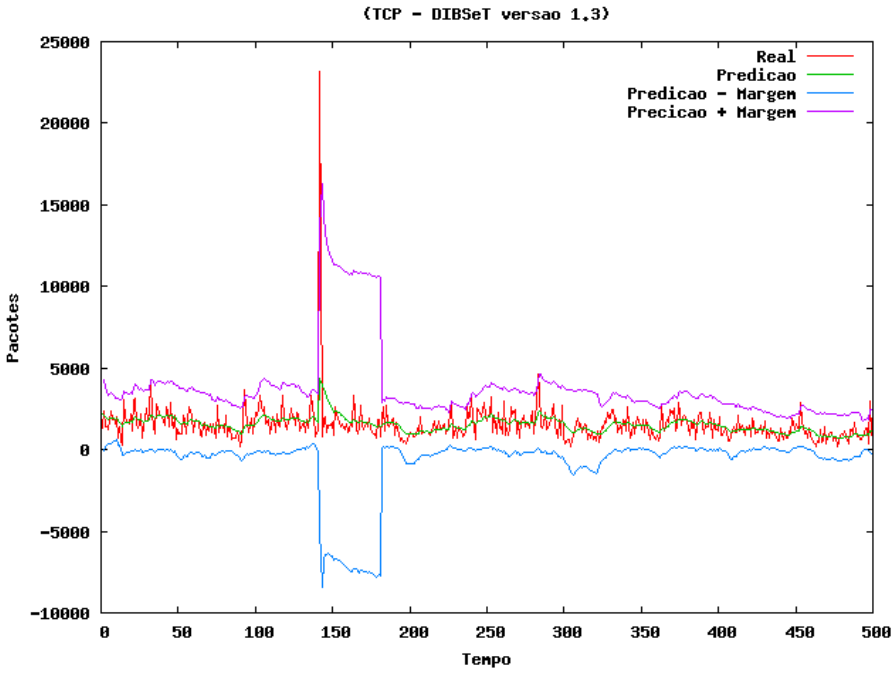

Figura 5.17: DIBSeT versão 1.3 com detalhes TCP

Os gráficos a seguir possuem detalhes sobre a margem 5.18 (a), sobre o tráfego real 5.18 (b) e sobre os alarmes 5.18 (c).

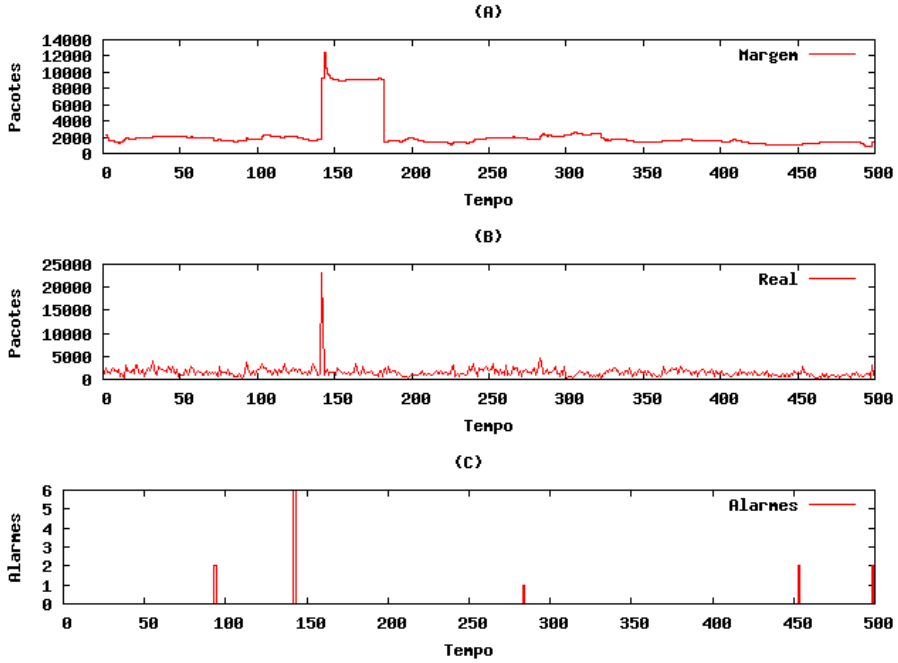

Figura 5.18: DIBSeT versão 1.3 Alarmes TCP

5.2.4.2 Pacotes ICMP

A figura 5.19 mostra o tráfego de pacotes ICMP normal da rede com a predição e as margens inferior e superior dinâmicas.

Os gráficos com detalhes sobre a margem 5.20 (a), sobre o tráfego real 5.20 (b) e sobre os alarmes 5.20 (c) podem ser vistos a seguir no texto.

A tabela 5.8 exibe as informações referentes à coleta dos alarmes e ataques informado pelo DIBSeT versão 1.3. A partir deste ponto não foi possível melhorar os resultados de detecção de anomalias, justificando a próxima etapa que é o uso de filtro dos alarmes através de *wavelets*.

Tabela 5.8: Quadro do DIBSeT versão 1.3

	TCP	**ICMP**
Alarmes	5	10
Ataques	1	1

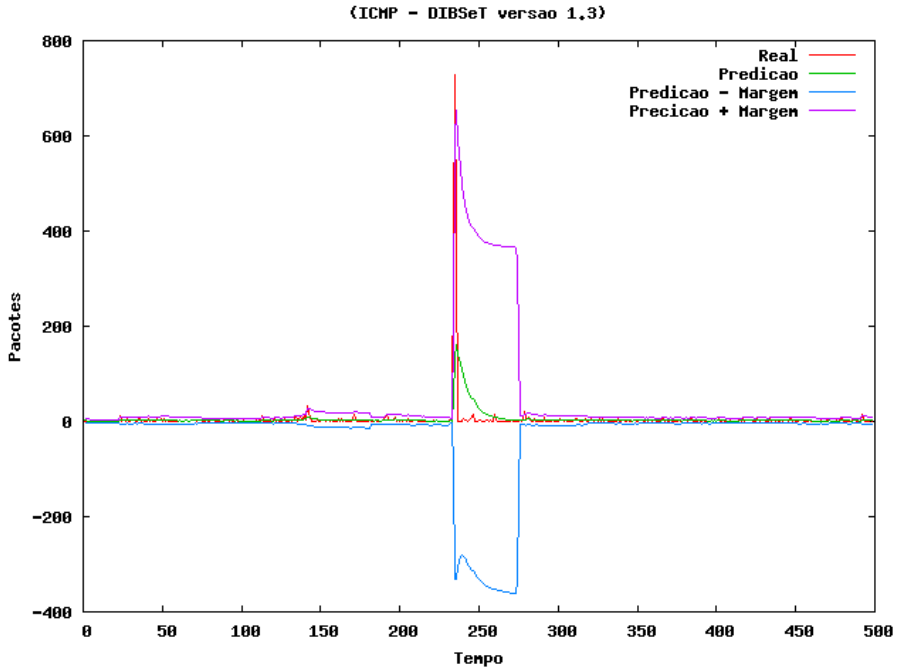

Figura 5.19: DIBSeT versão 1.3 com detalhes ICMP

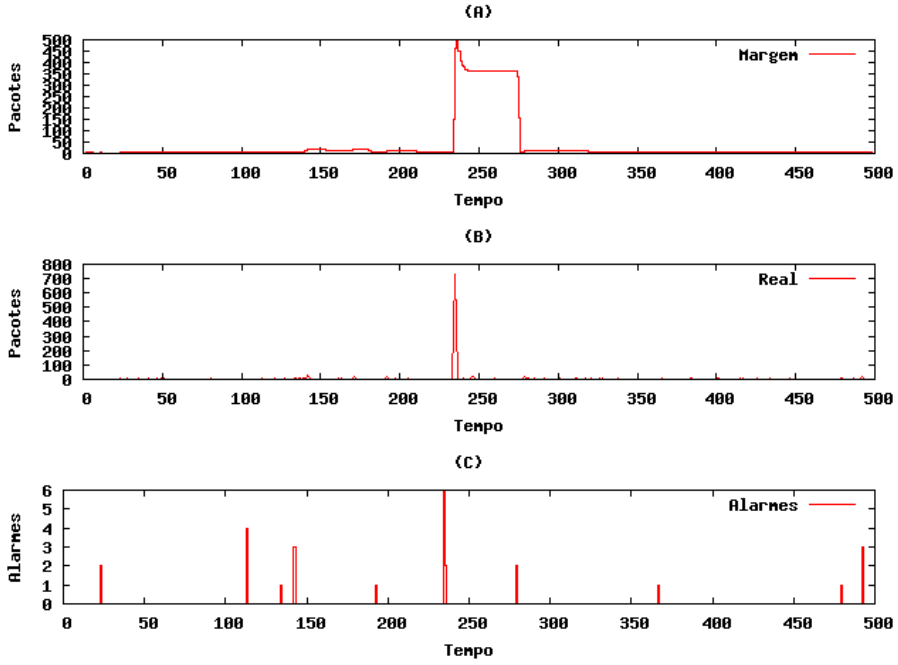

Figura 5.20: DIBSeT versão 1.3 Alarmes ICMP

5.2.5 Testes com DIBSeT-W

Os gráficos 5.21 e 5.23 a seguir no texto mostram dados de saída do DIBSeT-W gerados a partir de dados TCP e ICMP.

5.2.5.1 *Pacotes TCP*

A figura 5.21 mostra o tráfego normal da rede com a predição e as margens inferior e superior calculadas de modo dinâmico.

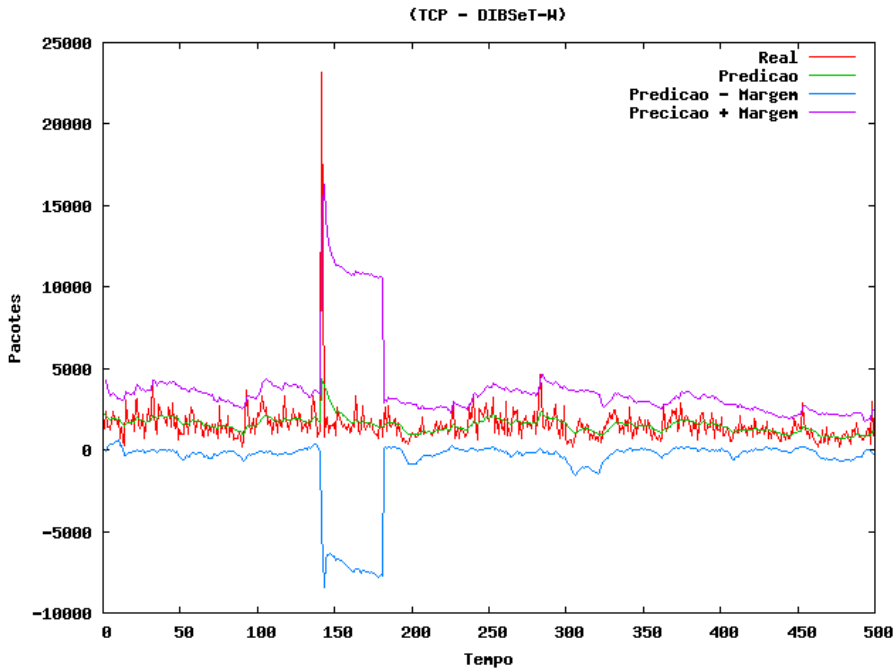

Figura 5.21: DIBSeT-W com detalhes TCP

Os gráficos 5.22 (a), 5.22 (b) e 5.22 (c), mostram, respectivamente, os detalhes sobre a margem , sobre o tráfego real e sobre os alarmes.

5.2.5.2 *Pacotes ICMP*

A figura 5.23 mostra o tráfego normal da rede com a predição e as margens inferior e superior dinâmicas.

Os gráficos com detalhes sobre a margem 5.24 (a), sobre o tráfego real 5.24 (b) e sobre os alarmes 5.24 (c) podem ser vistos no texto a seguir.

Na tabela 5.9 pode-se perceber a eficiência dos filtros *wavelets* na minimização dos alarmes falsos.

Tabela 5.9: Quadro do DIBSeT-W

	TCP	**ICMP**
Alarmes	1	1
Ataques	1	1

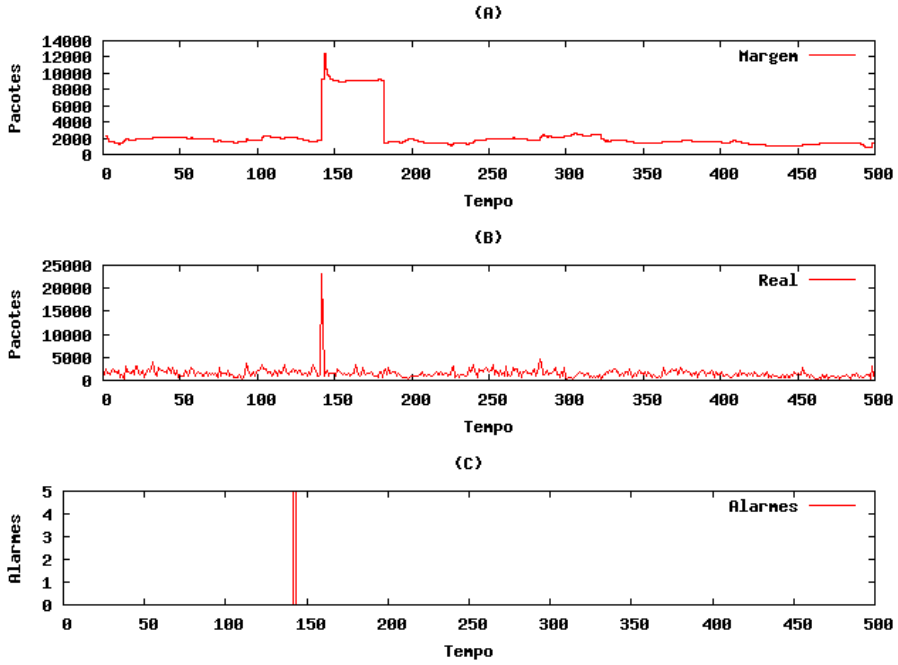

Figura 5.22: DIBSeT-W Alarmes TCP

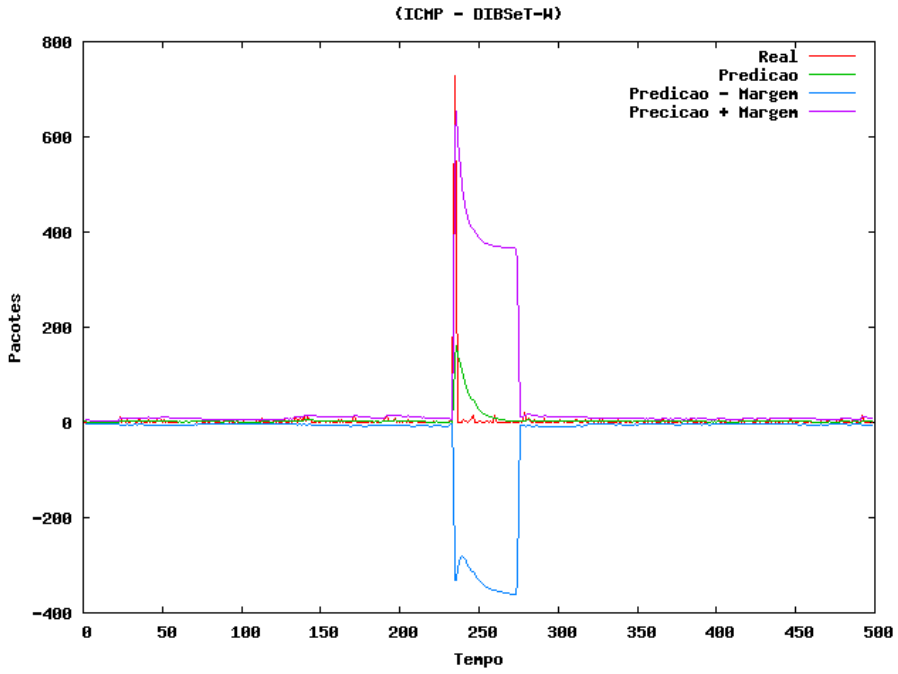

Figura 5.23: DIBSeT-W com detalhes ICMP

5.3 Experimentação

Nesta seção serão apresentados os testes com o tráfego total das três semanas de treinamento da base de dados do DARPA. E posteriormente o teste final com a base de dados do IAS. Como já descrito anteriormente na tabela 5.4, a segunda semana de treinamento

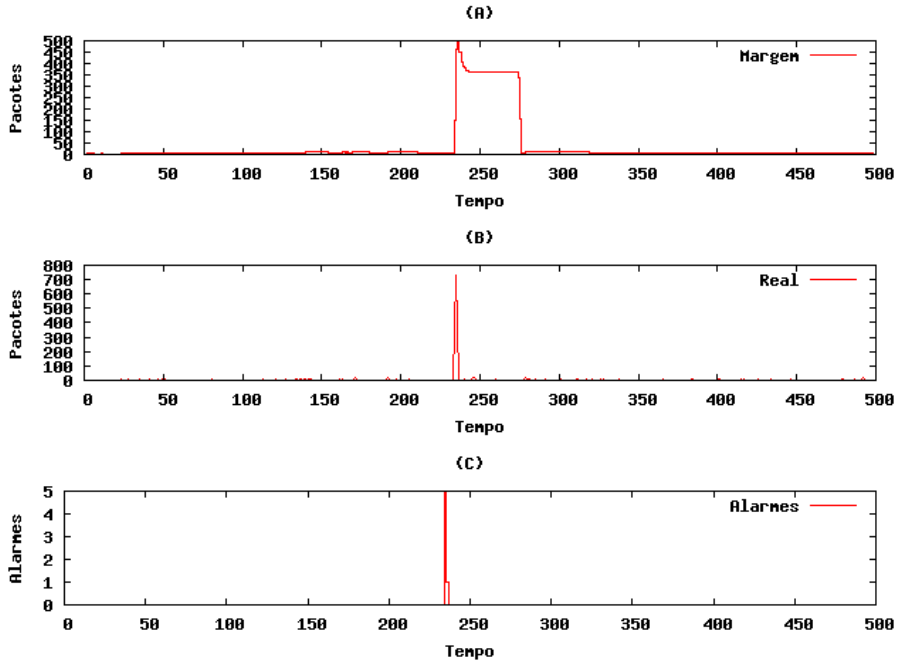

Figura 5.24: DIBSeT-W Alarmes ICMP

possui 36 ataques documentados, porém nem todos os ataques geram anomalias na rede. O DIBSeT-W pode identificar ataques que geram picos no tráfego normal da rede, os ataques enquadrados nesta categoria podem ser detectados. Todos os testes foram realizados com os contadores do tráfego total da rede, em todas as versões do DIBSeT e também no DIBSeT-W.

5.3.1 Resultados com DIBSeT versão 1.0

A figura 5.25 mostra o tráfego normal da rede com a predição e as margens inferior e superior estipuladas de forma fixa, conforme a primeira versão do DIBSeT. A fim de tornar os resultados mensuráveis, a quantidade de alarmes relatada pelo DIBSeT versão 1.0 será levada em consideração para realizar as comparações com todas as demais versões.

Os gráficos detalhados sobre o tráfego real 5.26 (a) e sobre os alarmes gerados 5.26 (b) podem ser vistos logo mais abaixo no texto.

5.3.2 Resultados com DIBSeT versão 1.1

A figura 5.27 mostra o tráfego normal da rede com a predição e as margens inferior e superior estipuladas de forma fixa, conforme a segunda versão do DIBSeT.

Os gráficos 5.28 (a) e 5.28 (b), mostram em detalhados o tráfego real e os alarmes gerados.

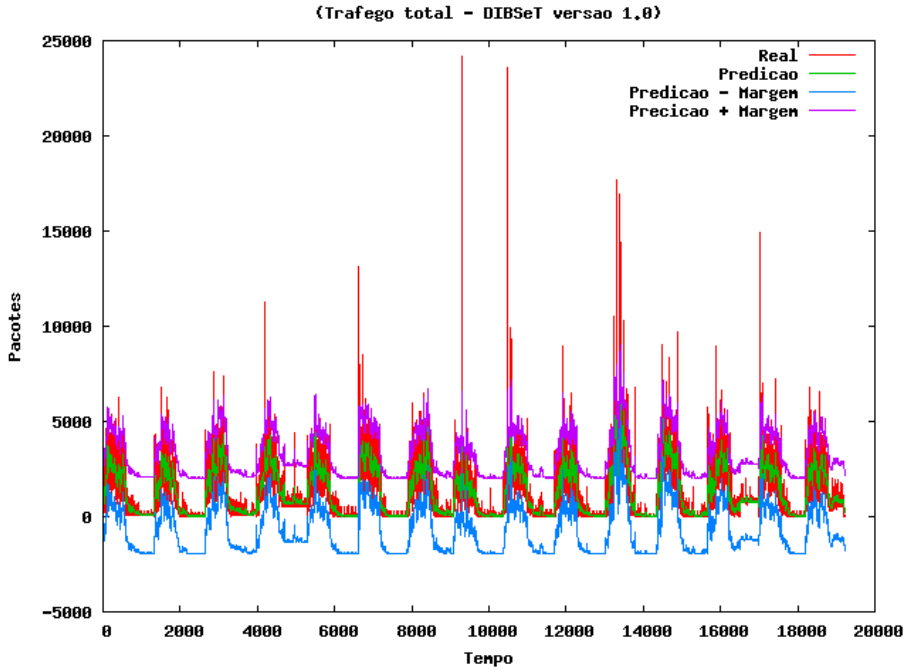

Figura 5.25: DIBSeT versão 1.0 com detalhes de todo o tráfego

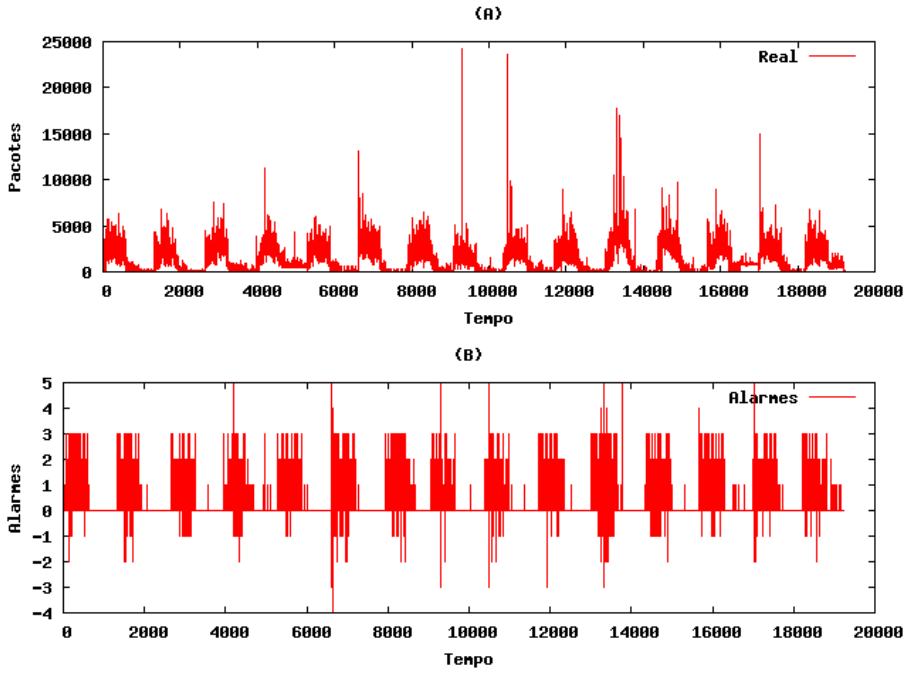

Figura 5.26: DIBSeT versão 1.0 com Alarmes do tráfego total

Na versão 1.1, o número de anomalias detectadas corretamente não melhorou com relação à versão 1.0, porém os falsos positivos diminuíram.

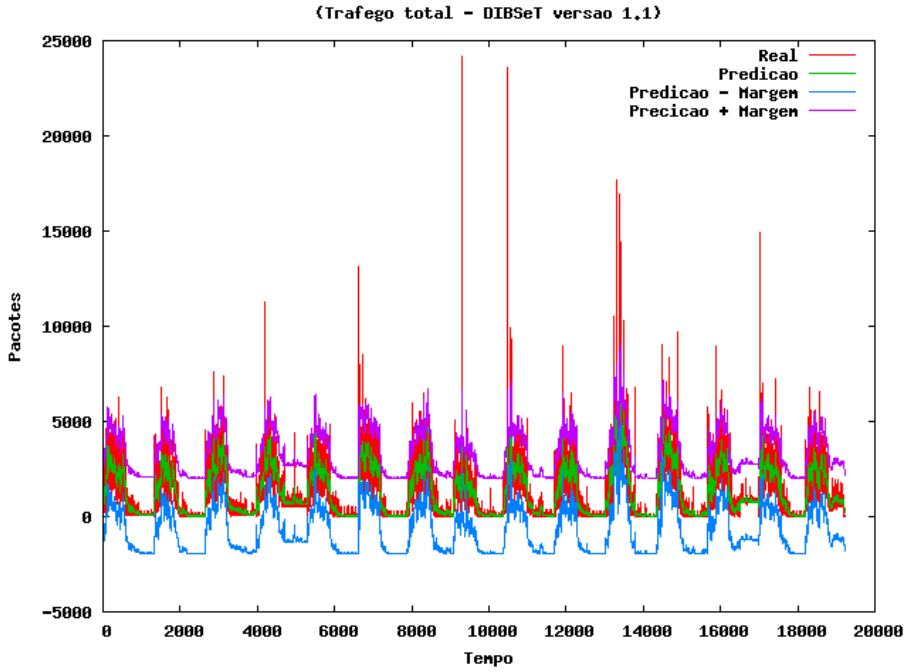

Figura 5.27: DIBSeT versão 1.1 com detalhes de todo o tráfego

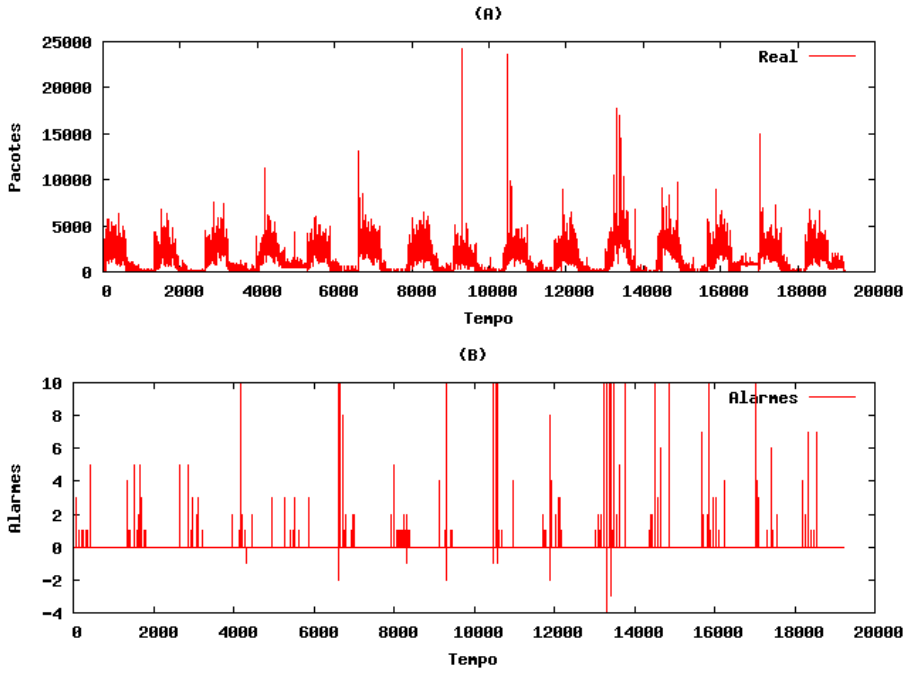

Figura 5.28: DIBSeT versão 1.1 com Alarmes do tráfego total

5.3.3 Resultados com DIBSeT versão 1.2

A figura 5.29 mostra o tráfego normal da rede com a predição e as margens inferior e superior estipuladas de forma dinâmica, conforme o DIBSeT versão 1.2. Após a mudança da classe de margem estática para classe de margem dinâmica, não houve mudança no

número de ataques detectados, mas os falsos positivos diminuíram em relação ao DIBSeT versão 1.0.

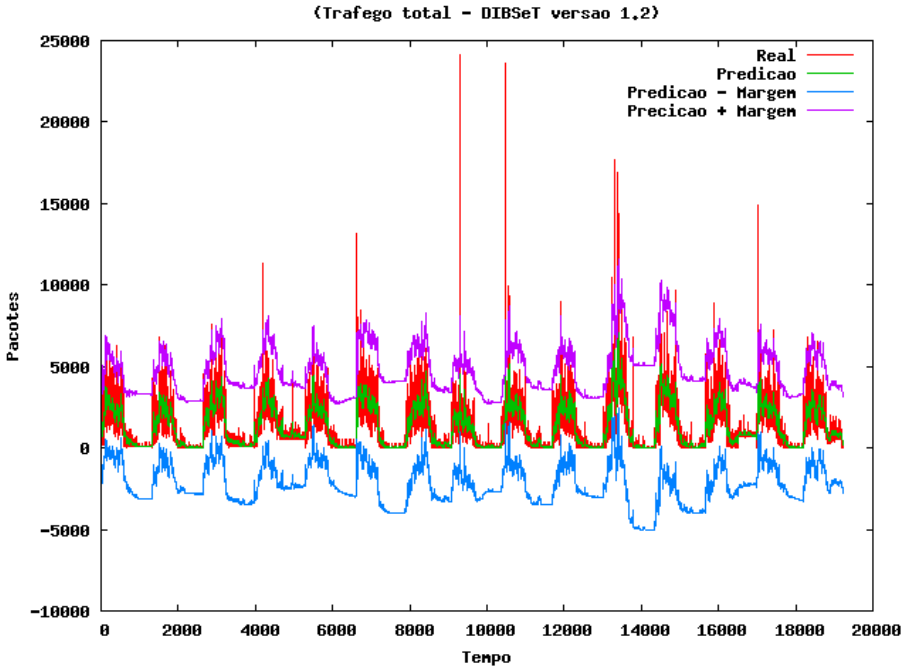

Figura 5.29: DIBSeT versão 1.2 com detalhes de todo o tráfego

Os gráficos detalhados sobre o tráfego real 5.30 (a) e sobre os alarmes gerados 5.30 (b) podem ser vistos um pouco mais abaixo no texto.

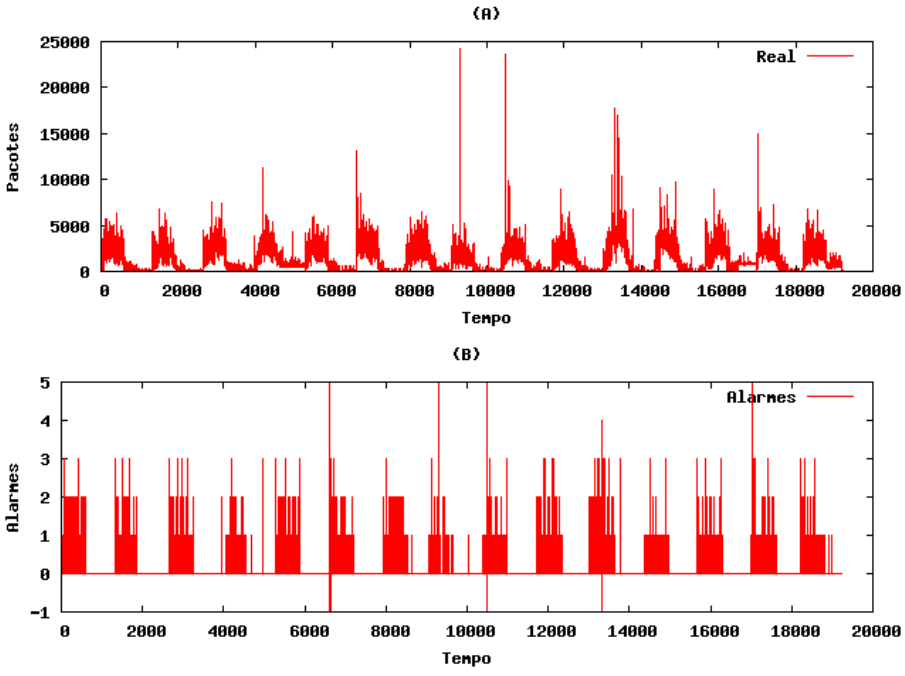

Figura 5.30: DIBSeT versão 1.2 com Alarmes do tráfego total

5.3.4 Resultados com DIBSeT versão 1.3

A figura 5.31 mostra o tráfego normal da rede com a predição e as margens inferior e superior estipuladas de forma dinâmica, conforme o DIBSeT versão 1.3. Esta versão usou em conjunto a nova classe de alarmes e as margens dinâmicas, com isso aumentou seu desempenho detectando mais ataques e os falsos positivos diminuíram muito em relação ao DIBSeT versão 1.0.

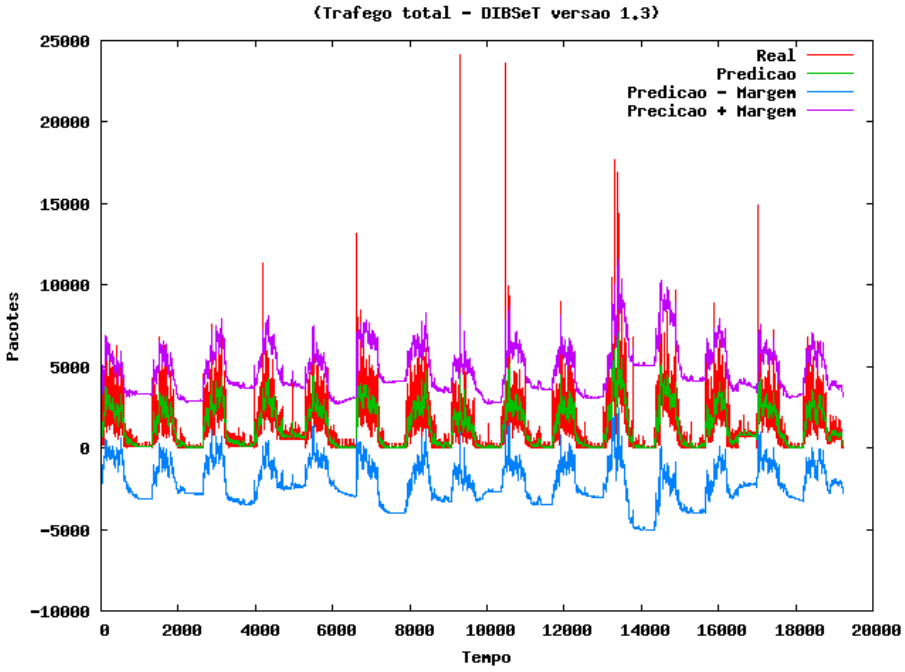

Figura 5.31: DIBSeT versão 1.3 com detalhes de todo o tráfego

Os gráficos detalhados sobre o tráfego real 5.32 (a) e sobre os alarmes gerados 5.32 (b) podem ser vistos mais abaixo no texto.

5.3.5 Resultados com DIBSeT-W

A figura 5.33 mostra o tráfego normal da rede com a predição e as margens inferior e superior estipuladas de forma dinâmica, conforme o DIBSeT-W. Quando os dados foram submetidos à análise com o filtro dos alarmes através de *wavelets*, a detecção dos ataques aumentou e a geração dos alarmes falsos diminuíram em relação ao DIBSeT versão 1.0, comprovando a sua eficiência em relação às outras versões do DIBSeT.

Os gráficos detalhados sobre o tráfego real 5.34 (a) e sobre os alarmes gerados 5.34 (b) podem ser vistos mais abaixo no texto.

Para melhor entendimento sobre resultados, as tabelas 5.10, 5.11 e 5.12 mostram um

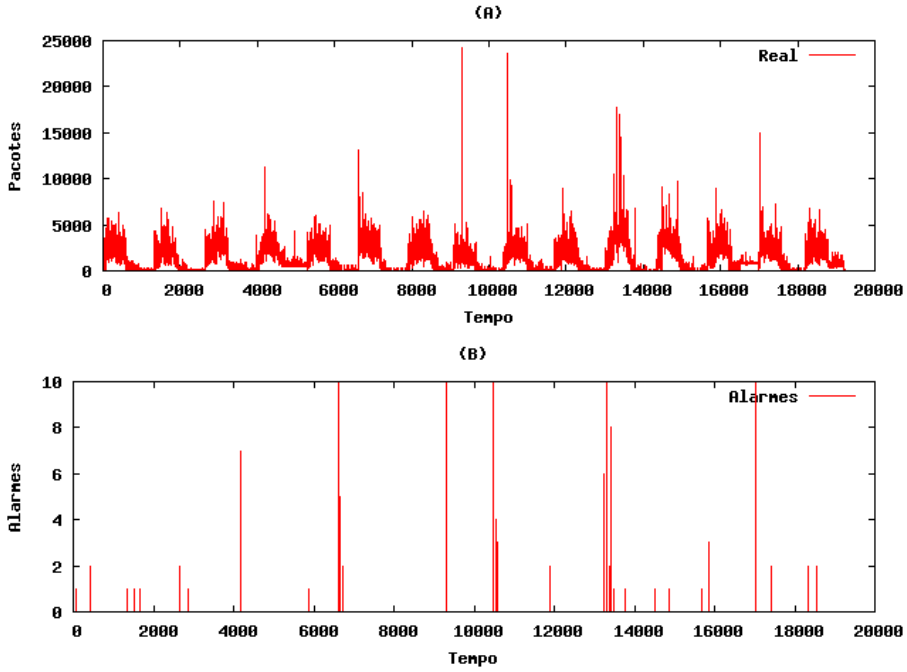

Figura 5.32: DIBSeT versão 1.3 com Alarmes do tráfego total

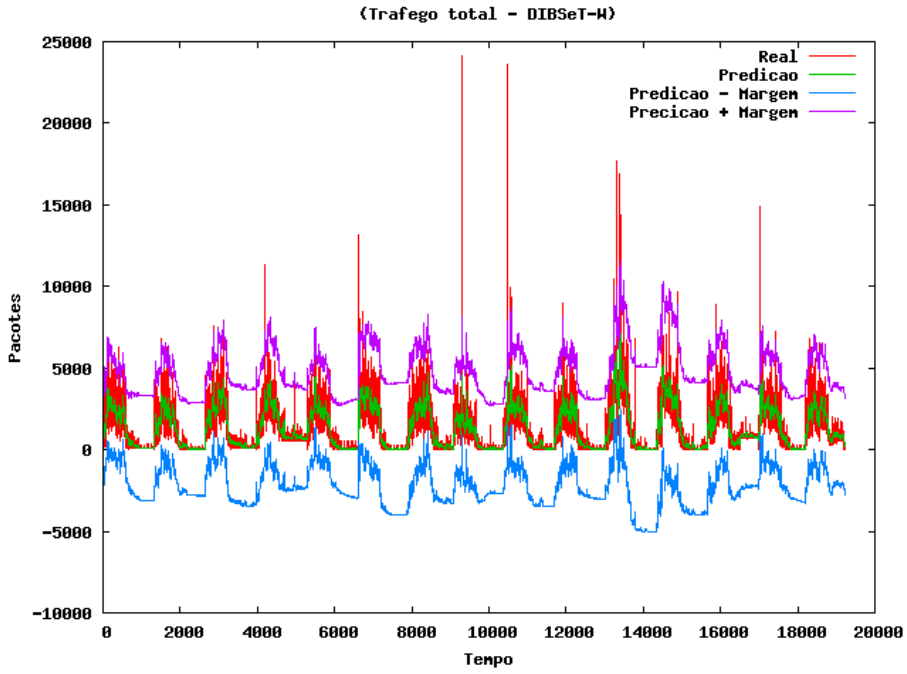

Figura 5.33: DIBSeT-W com detalhes de todo o tráfego

resumo comparativo dos alarmes encontrados.

Na documentação do DARPA foram relatados 2 ataques do tipo *Smurf* na base de dados, porém nenhuma versão do DIBSeT conseguiu detectá-los quando analisado o tráfego total dos dados, pois mesmo com a grandeza das anomalias geradas por este ataque nos pacotes ICMP, se comparadas com a totalidade do tráfego da rede ainda era muito

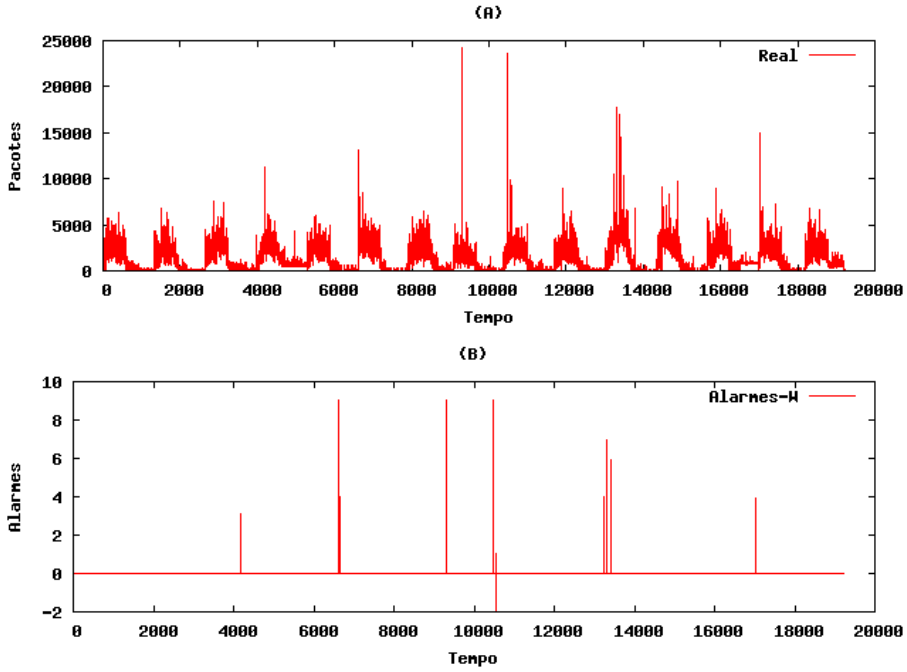

Figura 5.34: DIBSeT-W com Alarmes do tráfego total

Tabela 5.10: Lista resumo dos resultados TCP

	Verdadeiro positivo	**Falso positivo**
DIBSeT 1.0	1	177
DIBSeT 1.1	1	6
DIBSeT 1.2	1	185
DIBSeT 1.3	1	4
DIBSeT-W	1	0

Tabela 5.11: Lista resumo dos resultados ICMP

	Verdadeiro positivo	**Falso positivo**
DIBSeT 1.0	1	36
DIBSeT 1.1	1	19
DIBSeT 1.2	1	44
DIBSeT 1.3	1	9
DIBSeT-W	1	0

pequena e passando assim despercebido. Entretanto quando analisado apenas o tráfego ICMP, o ataque foi detectado por todas as versões do DIBSeT. O DIBSeT 1.1 teve 95,9% menos falsos positivos apresentados em comparação com o DIBSeT versão 1.0. O DIBSeT 1.2 apresentou uma melhora de 58,9% em relação ao DIBSeT versão 1.0 na detecção dos falsos positivos. O DIBSeT versão 1.3 usou em conjunto a nova classe de alarmes e as

Tabela 5.12: Lista resumo dos resultados das 3 semanas de dados do DARPA

	Verdadeiro positivo	**Falso positivo**	**Falso negativo**
DIBSeT 1.0	4	4902	5
DIBSeT 1.1	4	197	5
DIBSeT 1.2	4	2014	5
DIBSeT 1.3	5	39	4
DIBSeT-W	7	5	2
Ataques documentados	9	0	0

margens dinâmicas, com isso aumentou seu desempenho detectando 55,5% dos ataques e os falsos positivos diminuíram em 99,3% com relação ao DIBSeT versão 1.0. Por fim, quando os mesmos dados foram submetidos à análise com o filtro dos alarmes através de *wavelets*, a detecção dos ataques foi de 77,77% e a os alarmes falsos gerados diminuíram 99,9% em relação ao DIBSeT versão 1.0.

5.3.6 Resultados com o IAS

A intenção dos testes com essa base de dados é analisar, identificar e usar os contadores de alguns protocolos de rede, a fim de encontrar possíveis anomalias no tráfego da rede. Para os testes a seguir foi usado o tráfego total da rede. Os mesmos dados foram testados com o DIBSeT versão 1.3 e DIBSeT-W.

5.3.6.1 DIBSeT versão 1.3

Os resultados dos testes sobre a base de dados do IAS podem ser observados nas figuras 5.35 e 5.36.

5.3.6.2 DIBSeT-W

Os gráficos dos testes sobre a base de dados do IAS podem ser observados nas figuras 5.37 e 5.38.

Os dados do IAS submetidos aos testes com o DIBSeT versão 1.3 indicou 6 alarmes durante seu funcionamento, como pode ser visto no gráfico 5.36. Após análise pelo DIBSeT-W, não foram encontrados alarmes. Com isso acredita-se que a UFSM, durante o período correspondente aos dados coletados em sua rede, não sofreu ataques que pudessem gerar anomalias no tráfego normal da sua rede.

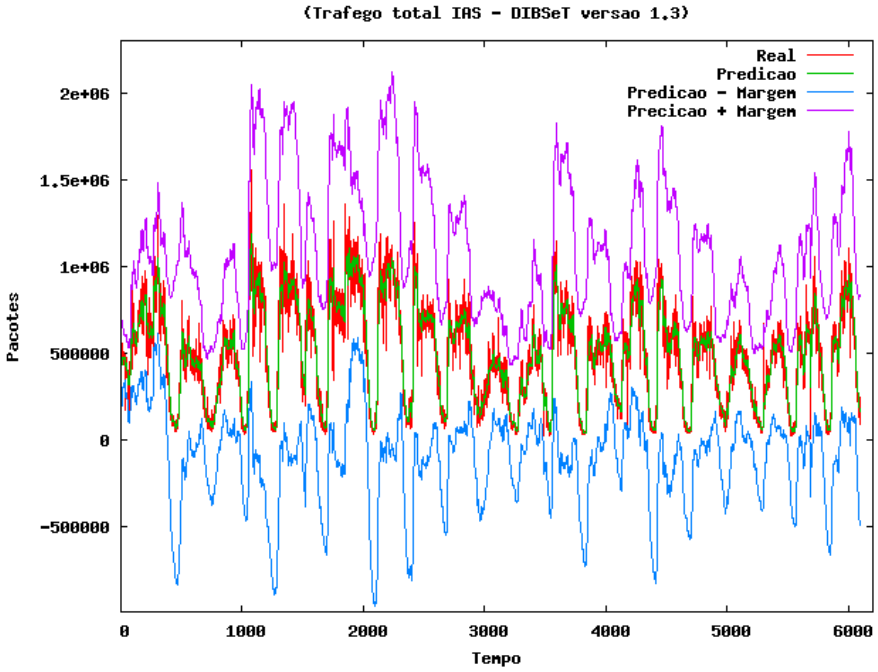

Figura 5.35: DIBSeT versão 1.3 com dados do IAS

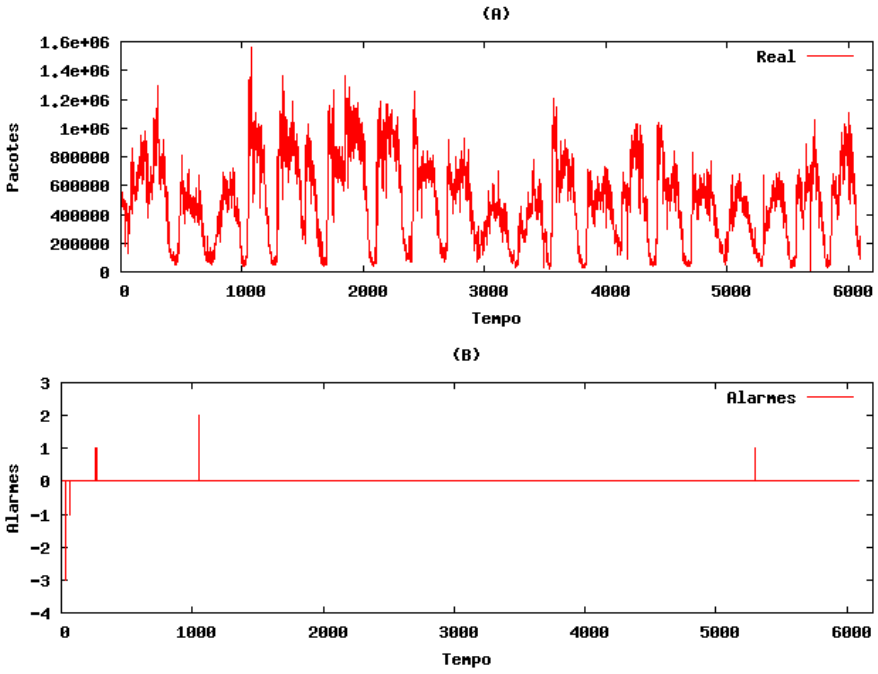

Figura 5.36: DIBSeT versão 1.3 com Alarmes do IAS

5.4 Resumo do capítulo

O uso de uma janela pequena nos primeiros testes do DIBSeT-W possibilitou que as análises fossem feitas em um ambiente controlado tornando mais fáceis as observações dos resultados. Este capítulo apresentou as provas de conceito com todas as versões do

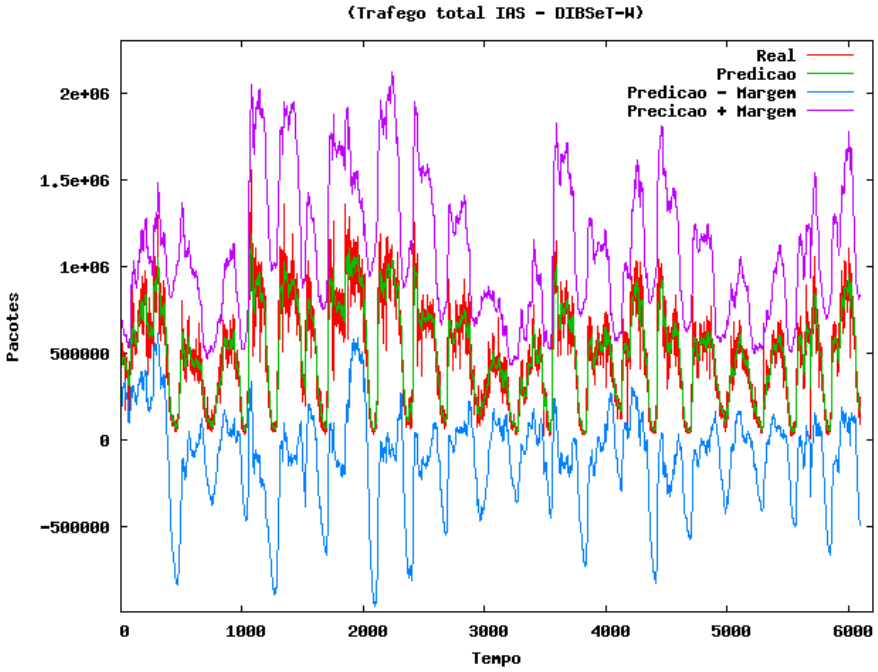

Figura 5.37: DIBSeT-W com dados do IAS

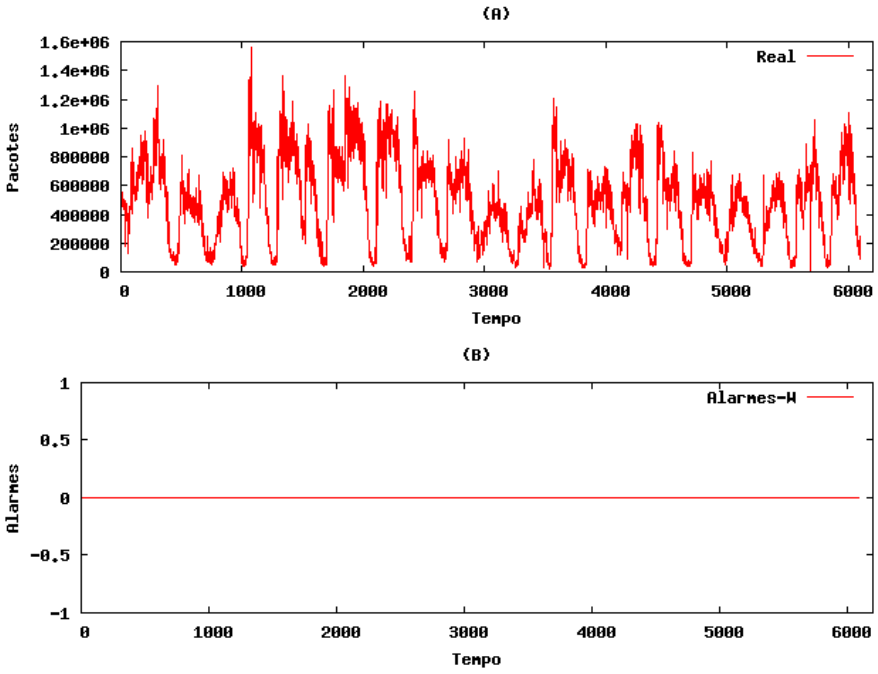

Figura 5.38: DIBSeT-W com Alarmes do IAS

DIBSeT e DIBSeT-W. Por final é apresentado o teste com a base de dados do IAS, a qual representa o tráfego real da Universidade Federal de Santa Maria. Os resultados gerados neste capítulo mostraram que com o uso de *wavelets* a detecção dos ataques teve um ganho aumentando de 55,55% para 77,77% e os alarmes falsos diminuíram 99,9% em relação ao DIBSeT 1.0. Todos estes resultados evidenciam o quanto foi eficiente as

melhorias no detector de intrusão baseado em anomalias.

6 CONCLUSÕES

Um dos principais objetivos dos sistemas de detecção de intrusão é a rápida detecção de anomalias com uma baixa taxa de alarmes falsos. Este trabalho apresentou técnicas sobre a detecção de anomalias através de cálculos de predições elaborados com Séries Temporais e com os alarmes gerados foi aplicado um método de filtragem baseado em *wavelets*. Primeiramente, a base de dados do DARPA foi usada para entrada dos dados pelo motivo de ser conhecida e documentada. Os dados do DARPA foram uma boa alternativa, auxiliando a ferramenta para a análise, através dos quais foi possível a validação do algoritmo e geração de bons resultados para tornar o trabalho comparável com outros.

Com os gráficos exibidos no capítulo anterior, pode-se notar que a partir da implantação da nova classe de alarmes na versão 1.1 os alarmes não ficaram dependentes de um valor fixo, podendo moldar-se e adaptar-se aos novos padrões estabelecidos no tráfego da rede. O grande destaque foi a minimização dos alarmes falsos.

Após a mudança da classe de margem estática para classe de margem dinâmica na versão 1.2 também não houve mudança no número de ataques detectados, mas quando usado em conjunto com a nova classe de alarmes o sistema tornou-se mais eficiente. Uma melhora significativa na detecção dos ataques deu-se depois que os alarmes foram submetidos ao filtro através de *wavelets*, comprovando a sua eficiência em relação às outras versões do DIBSeT.

Apesar dos falsos positivos encontrados, os resultados finais ficaram melhores após a utilização de *wavelets*. Os ataques detectados aumentaram de 55, 55% para 77,77%. Porém, o grande destaque vai para a queda no número de alarmes falsos gerados, eles diminuíram 99,9% em relação ao DIBSeT 1.0. Todos estes números mostram a evolução do processo de detecção de anomalias em uma rede de computadores.

6.1 Trabalhos futuros

Como continuação deste trabalho, pode-se aplicar este método de detecção de anomalias em um tráfego real de forma online. E em um futuro próximo, estes resultados podem ser comparados com o trabalho em desenvolvimento na *Fachhochschule Gelsenkirchen* na detecção de intrusão através de redes neurais, além disso, as comparações de sua eficiência podem ser realizadas em relação a outros IDS.

Outros métodos estatísticos podem ser estudados na detecção das anomalias. As análises podem ser mais eficientes quando usada a correlação dos alarmes com vários fluxos de dados, separando os dados de entrada em protocolos distintos, ou separando pelo tipo de pacote, por exemplo: separando TCP/SYN de TCP/FIN.

REFERÊNCIAS

BEJTLICH, R. The Tao of Network Security Monitoring Beyond Intrusion Detection. **Addison Wesley**, [S.l.], 2004.

BOLZ, C.; ROMMEY, W.; ROGERS, B. L. Safely Train Security Engineers Regarding the Dangers Presented by Denial of Service Attacks. **ACM Conference on Information Technology Education, Session Security II**, [S.l.], p.62–72, 2004.

BOWERMAN, B. L.; O'CONNELL, R. T. Forecasting and Time Series: an applied approach. **Belmont: Duxbury Press**, [S.l.], 1993.

Centro de Atendimento a Incidentes de Segurança. Disponível em: http://www.rnp.br/cais/, último acesso em outubro de 2008.

CERT - Denial-of-Service Attack via ping. Disponível em: http://www.cert.org/advisories/CA-1996-26.html, último acesso em outubro de 2008.

CERT - Denial-of-Service Developments. Disponível em: http://www.cert.org/advisories/CA-2000-01.html, último acesso em outubro de 2008.

CERT - Denial-of-Service Tools. Disponível em: http://www.cert.org/advisories/CA-1999-17.html, último acesso em outubro de 2008.

CERT - Denial-of-Service Vulnerabilities in TCP/IP Stacks. Disponível em: http://www.cert.org/advisories/CA-2000-21.html, último acesso em outubro de 2008.

CERT - TCP SYN Flooding and IP Spoofing. Disponível em: http://www.cert.org/advisories/CA-1996-21.html, último acesso em outubro de 2008.

DALMAZO, B. L.; VOGT, F.; PERLIN, T.; NUNES, R. C. Detecção de Intrusão baseado em Séries Temporais. **Anais do VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**, [S.l.], 2008.

Defense Advanced Research Projects Agency. disponível em: http://www.ll.mit.edu/IST/ideval/index.html. Último acesso em outubro de 2008.

DERI, L.; SUIN, S.; MASELLI, G. Design and Implementation of an Anomaly Detection System: an empirical approach. **Proceedings of Terena Networking Conference (TNC 03)**, Zagreb, Croatia, 2003.

DINDA, P. A. A Toolkit for Resource Prediction in Distributed System. **RPS**, [S.l.], 2005. disponível em: http://rps.cs.northwestern.edu/. Último acesso em novembro de 2008.

DONOHO, D. L.; JOHNSTONE, I. M. **De-noising by soft-thresholding**. 1995.

DWYER, D. Network Intrusion Detection. **New Riders Publishing**, [S.l.], v.2003, 2003. 3rd Edição.

EHLERS, R. S. **Análise de Séries Temporais.** 2005, 3rd Edição. Departamento de Estatística, Universidade Federal do Paraná, PR.

ELETRONIC Industries Association and Telecommunications Industry Association. Standard ANSI TIA EIA 568 A, último acesso em outubro de 2008.

GAO, J.; HU, G.; YAO, X.; CHANG, R. Anomaly Detection of Network Traffic Based on Wavelet Packet. **Asia-Pacific Conference on Communications.**, [S.l.], 2006.

GOODALL, J. R. Visualizing Network Traffic For Intrusion Detection. **ACM Symposium on Designing Interactive Systems**, [S.l.], p.343–364, 2006.

HUANG, C. T.; THAREJA, S.; SHIN, Y. J. **Wavelet based Real Time Detection of Network Traffic Anomalies**. Columbia, SC: Securecomm and Workshops, 2006. Departamant of Computer Sci. e Eng.

JAVA Technology. Disponível em: http://java.sun.com/, último acesso em outubro de 2008.

KARGL, F.; MAIER, J.; WEBER, M. Protecting Web Servers from Distributed Denial of Service Attacks. **ACM Proceedings of the 10th international conference on World Wide Web**, Deptartament of Computer Sci and Eng, South Carolina University, Columbia, SC, 2001.

KIM, D. M.; CHO, J. M.; LEE, H. S.; JUNG, H. S.; KIM, J. O. Prediction of Dynamic Line Rating Based on Assessment Risk by Time Series Weather Model. **PMAPS 2006 International Conference on Probabilistic Methods Applied to Power Systems**, [S.l.], 2006.

KOMPELLA, R. R.; SINGH, S.; VARGHESE, G. On Scalable Attack Detection in the Network. **IEEE/ACM TRANSACTIONS ON NETWORKING**, University of California, San Diego, v.15, n.1, 2007.

KOZAKEVICIUS, A.; NUNES, R. C.; RODRIGUES, C. R.; FILHO, R. G. Adaptive ECG Filtering and QRS Detection Using Orthogonal Wavelet Transform. **IASTED International Conference on BioMedical Engineering (BioMed 2005)**, Universidade Federal de Santa Maria, RS, 2005.

LAUREANO, M. A. P. Sistemas para Identificação de Invasão. **Anais do 6º Fórum Internacional Software Livre - FISL. Porto Alegre**, Pontifícia Universidade Católica do Paraná, PR, 2005. Dissertação de Mestrado.

LUNARDI, R.; DALMAZO, B. L.; AMARAL, E.; NUNES, R. C. DIBSet: um detector de intrusão por anomalias baseado em séries temporais. **Anais do VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**, [S.l.], 2008.

MALLAT, S. G. A theory for multiresolution signal decomposition: the wavelet representation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [S.l.], v.11, p.674–693, 1989.

MARTINS, C. S.; SANTOS, A. L. M.; MATTOS, C. L.; HORA, E. C. Refinando análises do SNORT através de correlação de eventos com o estado ativo da rede. **Anais do IV Simpósio sobre Segurança em Informática**, [S.l.], 2002.

MIRKOVIC, J.; REIHER, P. A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. **ACM SIGCOMM Computer Communications Review**, [S.l.], v.34, n.2, 2004.

NAKAMURA, E. T.; GEUS, P. L. **Segurança de redes em ambientes cooperativos**. São Paulo, Brasil: Editora Novatec, 2002.

NOGUEIRA, T. J. P. **Invasão de redes, Ataques e Defesas.** Rio de Janeiro: Editora Ciência Moderna Ltda, 2005.

NORTHCUTT, S. **Network Intrusion Detection: an analyst's handbook.** New Riders Publishing: Editora Ciência Moderna Ltda, 1999.

NUNES, R. C. **Adaptação dinâmica do timeout de detectores de defeitos através do uso de séries temporais.** Tese de Doutorado pela Universidade Federal do Rio Grande do Sul in UFRGS. Porto Alegre, RS, 2003.

PENG, T.; LECKIE, C.; RAMAMOHANARAO, K. Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems. **ACM Computing Surveys**, The University of Melbourne, Australia, v.39, n.1, 2007.

POHLMANN, N.; PROEST, M. Internet Early Warning System: the global view. **Vieweg, Securing Eletronic Business Process**, [S.l.], p.377–386, 2006.

RFC 793. Information Sciences Institute - Edited by Jon Postel. Available at http://rfc.sunsite.dk/rfc/rfc793.html.

SNORT - http://www.snort.com.br. Último acesso em outubro de 2008.

STRANG, G. Wavelet Transforms versus Fourier transforms. **Bulletin of the American Mathematical Society**, [S.l.], 1993.

TANENBAUM, A. S. **Redes de Computadores**. Rio de Janeiro: Editora Campus, 2003.

TCPDUMP - http://www.tcpdump.org. Último acesso em outubro de 2008.

TCPSTAT - http://www.frenchfries.net/paul/tcpstat/. Último acesso em outubro de 2008.

THOTTAN, M.; JI, C. Anomaly Detection in IP Networks. **IEEE Transactions on Signal Processing**, [S.l.], v.51, n.8, 2003.

TRAN, N.; REED, D. A. ARIMA Time Series Modeling and Forecasting for Adaptive I/O Prefetching. **ACM 15th International Conference on Supercomputing**, Sorrento, Italy, 2001.

USEVITCH, B. E. A Tutorial on Modern Lossy Wavelet Image Compression: foundations of jpeg 2000. **IEEE Signal Processing Magazine**, [S.l.], 2001.

WANG, X. Research on effect of frequency band energy leakage to wavelet denoising. **7th World Congress on Intelligent Control and Automation**, [S.l.], 2008.

WHEELWRIGHT, S. C.; MAKRIDAKIS, S. **Forecasting Methods for Management.** New York: John Wiley & Sons Inc, 1985.

XU, Y.; WANG, G.; GU, Y.; LIU, H. A Novel Wavelet Packet Speech Enhancement Algorithm Based On Time-Frequency Threshold. **Second International Conference on Innovative Computing, Information and Control**, [S.l.], 2007.