UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

ÉMERSON SALVADORI VIRTI

Implementação de um IDS Utilizando SNMP e Lógica Difusa

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação

Profa. Dra. Liane M. Rockenbach Tarouco
Orientadora

Porto Alegre, outubro de 2007.

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Virti, Émerson Salvadori

Implementação de um IDS utilizando SNMP e Lógica Difusa / Émerson Salvadori Virti – Porto Alegre: Programa de Pós-Graduação em Computação, 2007.

93 f.:il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação. Porto Alegre, BR – RS, 2007. Orientadora: Liane M. Rockemback Tarouco.

1.Sistemas Detectores de Intrusões. 2.Segurança 3.Lógica Difusa. 4.SNMP. I. Tarouco, Liane M. Rockemback II. Título.

AGRADECIMENTOS

Muitas pessoas contribuíram para que esse trabalho tivesse êxito. Algumas vezes pela ajuda direta, outras pelo simples proferir de palavras animadoras, todas elas necessitariam estarem aqui referenciadas.

Agradeço a Universidade Federal do Rio Grande do Sul, ao Instituto de Informática e a todos aqueles que contribuem para que esta universidade continue sendo pública, gratuita e de qualidade.

Agradeço à professora Liane Tarouco pelos auxílios prestados, por ter propiciado minha ida à Alemanha para a apresentação no congresso promovido pela IEEE, e por, principalmente, ser um modelo de dedicação e esforço em prol da ciência.

Agradeço também ao pessoal do POP-RS (Leandro Bertholdo e João Ceron) pela disponibilidade e apoio oferecido nos testes deste trabalho.

Agradeço ao professor Paulo Martins Engel pelas dicas relativas à tecnologia difusa.

Agradeço aos meus pais, Telmo e Selita e às minhas irmãs, Jacqueline e Denise por todo o empenho dedicado à minha educação. Educação que nunca ficou restrita à necessidade de conhecimento técnico. Deles recebi os valores que hoje prego, e a eles dedico este trabalho.

Agradeço a Deus... por tudo.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

ARP	Address Resolution Protocol
CERT	Computer Emergence Response Team
CPF	Cadastro de Pessoa Física
CPU	Central Processing Unit
DDoS	Distributed Denial of Service
DNS	Domain Name Service
DNSSEC	Security Domain Name Service
DoS	Denial of Service
FTP	File Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IP	Internet Protocol
MAC	Media Access Control
MIB	Mamagement Information Base
MISO	Multiple Input Single Output
NIDS	Network Intrusion Detection System
OID	Object Identifier
OSI	Open System Interconnection
RFC	Request For Comments
RMON	Remote Monitoring
SGMP	Simple Gateway Management Protocol
SMS	Short Message Service
SMTP	Simple Mail Transport Protocol
SNMP	Simple Network Management System
TCP	Transmission Control Protocol
TOR	The Onion Router

UDP	User Datagram Protocol
UFRGS	Universidade Federal do Rio Grande do Sul
UML	Unified Modeling Language

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Este trabalho busca o estudo da segurança em redes de computadores através da implementação de um sistema detector de intrusão embasado na captura de informações pela utilização do protocolo SNMP. Para alcançar-se a diminuição no número de falsos positivo e negativo, problema peculiar à maioria dos IDS, utiliza-se a lógica difusa para, com o auxilio dos administradores de segurança de cada rede, possibilitar a construção de um sistema detector de intrusão que melhor se adeque às características das redes monitoradas.

Posteriormente, utilizando o monitoramento de uma rede de produção, avalia-se a melhora na segurança obtida com o uso do IDS implementado por esse trabalho que, atuando quase em tempo real, propicia sua adoção como mecanismo complementar à segurança de redes.

Palavras-Chave: Sistemas detectores de intrusão, segurança, lógica difusa, SNMP.

Implementation of an IDS using SNMP and Fuzzy Logic

ABSTRACT

This work develops a study about Computer Network Security through the implementation of an Instruction Detection System (IDS) based on system information captured by the SNMP protocol. To reach a reduction in the number of false positive and false negative, a peculiar problem to the majority of the IDS, it is used fuzzy logic and the assistance of Network Security Administrators. Thus it is possible to build an Intrusion Detection System better adjusted to the network characteristics that must be monitored.

At last, by monitoring a production network, it is evaluated the overall security improvement obtained by the IDS proposed in this work and considers its adoption as a complementary network security mechanism.

Keywords: Intrusion detection systems, security, fuzzy logic, SNMP.

INTRODUÇÃO

A Internet é hoje o símbolo que a tecnologia trouxe para construir um mundo cada vez mais globalizado. A sua capacidade de estabelecer comunicações rápidas e baratas, está mudando a forma como os indivíduos se relacionam. Pessoas que, há pouco tempo, precisavam despender grandes quantias para poderem se comunicar via rede telefônica, agora, usando a Internet, podem fazê-lo com voz e vídeo sem grandes gastos financeiros. Por trás do fenômeno de popularidade representado pela Internet nos últimos anos há a necessidade das pessoas de se relacionarem, mesmo que virtualmente (CASTELLS, 2003).

Muitos dos sítios mais visitados atualmente estão associados a essa necessidade de comunicação e relacionamento. Alguns sítios permitem que os usuários postem vídeos particulares (*You Tube*) outros propiciam o aumento do número de pessoas "conhecidas" com as quais é possível estabelecer comunicação (*Orkut*) ou, ainda, há aqueles que criam uma realidade virtual onde o participante tem a capacidade de construir sua personalidade, podendo modelar seu próprio corpo (ou avatar) (*Second Life*).

Essa popularidade fez o mundo ficar cada vez mais conectado à Internet. Calcula-se que, atualmente, haja mais de um bilhão de pessoas que possuem acesso à grande rede (ARGAEZ, 2007). O Brasil já é o décimo país no mundo em número de *internautas*, com 22 milhões de pessoas conectadas. O número de sítios e domínios vem aumentando diariamente. Só no "*.com.br*" já há mais de 1 milhão de domínios registrados (REGISTRO, 2007).

Pela sua capacidade de rapidamente realizar comunicações, a Internet mudou também as relações financeiras no mundo (CASTELLS, 2003). Pedidos de compras e venda de ações em bolsas podem ser encaminhados a várias partes do mundo, tornando mais fácil a especulação financeira e fazendo os países terem de se proteger contra esse tipo de capital onde o dinheiro não representa, necessariamente, produção.

No entanto, por interligar milhões de pessoas, a Internet deixou de ser um lugar confiável. Alguns usuários passaram a tentar corromper o sistema e causar danos a outros indivíduos. Seus motivos variam desde almejar ganhos financeiros até apenas satisfazerem suas necessidades de auto-afirmação e reconhecimento perante grupos fechados de indivíduos com semelhantes intenções (SCHNEIER, 2000).

Surgiu o conceito de Hacker. Antes associados às pessoas que detinham grande conhecimento técnico ligado à tecnologia, o termo passou a ser utilizado para rotular usuários que utilizam esse conhecimento para burlar a segurança dos sistemas computacionais (FERREIRA, 2003).

Várias formas e modalidades de crimes virtuais começaram a surgir. Hoje, existem desde grupos que pretendem invadir o máximo de sites possível para realizarem suas

"pichações" e poderem registrar seus feitos em determinados sítios (PREATONI, 2006) até aqueles usuários envolvidos com crimes de formação de quadrilhas para fraudes em instituições financeiras. A tentativa de ganhar dinheiro aplicando golpes na Internet também é um crime bastante aplicado (CERTBR, 2007). Envio de spams, pichações, roubo de senhas, falsificação de sítios de bancos e ataques de negação de serviço são apenas alguns exemplos do grande número de formas de ataque que se tornaram incidentes corriqueiros na Internet.

Todos os dias milhões de pessoas convivem com o recebimento de e-mails maliciosos. O envio de e-mails contendo links para páginas falsas de banco (*Fishing*) tornou-se comum. Nem a Receita Federal foi poupada, há pouco tempo, os infratores enviaram milhares de mensagens para todo o país pedindo que os contribuintes fizessem o recadastramento de CPF (Cadastro de Pessoa Física) e fornecessem dados da declaração de imposto de renda. As páginas para as quais esses contribuintes eram encaminhados continham softwares maliciosos (*malwares*) que se instalavam no computador do usuário para realizar o roubo se senhas (CAIS, 2007).

Os bancos, na tentativa de dificultar os ataques de *Fishing,* alertam os seus clientes dizendo que não enviam e-mails para a caixa de e-mails pessoal dos correntistas. O Banco do Brasil, por exemplo, além de colocar a mensagem "cuidado com links e downloads contidos em mensagens promocionais" como título de sua página de autenticação, ainda criou uma caixa de e-mails vinculada a cada conta bancária. Essa caixa só recebe mensagens do próprio banco e só pode ser acessada se o cliente estiver autenticado no sítio do banco. Com medidas semelhantes a esta, os bancos tentam se proteger contra os numerosos fraudadores.

No entanto, as fraudes não estão restritas aos sistemas computacionais conectados à Internet. No ano de 2005, por exemplo, centenas de boletos bancários foram entregues a muitas instituições que possuíam domínios registrados na Internet brasileira. Esses boletos, bastantes parecidos com aqueles enviados pelo Registro-BR (REGISTRO, 2007), órgão que regula os domínios brasileiros, faziam a cobrança por um domínio supostamente registrado na Internet. O Centro de Processamento de Dados da UFRGS (Universidade Federal do Rio Grande do Sul) recebeu um desses boletos e o encaminhou à Polícia Federal. Evidentemente, muitas instituições devem ter realizado tal pagamento por suporem se tratar da verdadeira cobrança enviada pelo órgão regulador. Os crimes na Internet, por poderem ser direcionados a milhares ou milhões de pessoas, não precisam, necessariamente, furtar muitas quantias financeiras de cada vítima.

Muito da insegurança presente na atual Internet é fruto do descuido e desconhecimento do usuário final. Os seus utilizadores não precisam, necessariamente, terem conhecimentos técnicos associados à segurança. Essa falta de compreensão dos perigos envolvidos na utilização da grande rede auxilia os atacantes na obtenção de sistemas comprometidos que serão, posteriormente, fonte de ataques a outros computadores. A técnica de utilizar ambientes invadidos para realizar outros ataques, dificulta em muito a identificação do responsável pelo incidente de segurança.

Os ataques distribuídos de negação de serviço, por exemplo, estão intimamente ligados ao surgimento das grandes redes de sistemas zumbis (*bootnets*). Essas redes são formadas por computadores que foram invadidos e operam sob a ação de um software que permite, via Internet, o controle do computador atacado. Algumas vezes, essas invasões se dão por um mero descuido na execução de arquivos desconhecidos. Com um grande número de computadores sob seus comandos, os atacantes têm a

possibilidade de ordenarem que os zumbis façam centenas de tentativas de conexões a determinado servidor. Dependendo do número de *bots*, o criminoso tem o poder de esgotar os recursos de hardware, software e de largura de banda do servidor atacado. Muitas vezes, os donos de *bootnets* chantageiam os grandes *sites* para que, em troca de dinheiro, estes não sofram ataques de negação de serviço (CAIS, 2007) (SCHNEIER, 2000). Ou seja, um simples descuido na execução de um arquivo, pode fazer o computador de um usuário final tornar-se mais uma ameaça na Internet.

Com toda a gama de possibilidades de ataques, os danos causados pelos incidentes são surpreendentes. Estima-se que em 2007 os crimes virtuais movimentem até oito bilhões de dólares (CRIMES, 2007). No entanto, apesar das fraudes serem uma atividade extremamente lucrativa, há atacantes que, mesmo assim, intentam destruir a infra-estrutura da grande rede. Recentemente, um poderoso ataque de negação de serviço originado por computadores zumbis conseguiu parar três dos treze servidores DNS (*Domain Name Service*) centrais (*root servers*) (ATAQUE, 2007). Se o incidente conseguisse parar todos os servidores, a Internet mundial estaria arruinada.

Felizmente, parodiando o mundo real, existem pessoas empenhadas para que os criminosos virtuais paguem pelas infrações cometidas, ou ao menos, tenham maior dificuldade na elaboração de incidentes de segurança. No entanto, essa atividade "policial" não é nada fácil. Um atacante cuidadoso dificilmente será pego, pois seu conhecimento técnico permite que saiba exatamente por onde pode ser descoberto. Muitas vezes a origem dos ataques é de um computador comprometido localizado em alguma remota parte do mundo. Para descobrir exatamente a localização do atacante é preciso contatar as autoridades da segurança do país origem dos ataques. Infelizmente, alguns países têm pouco ou nenhum controle sobre a segurança de sua rede nacional.

A identificação das origens dos ataques esbarra também em outros empecilhos. O anonimato na Internet não é algo difícil de ser conseguido. O encadeamento de vários *proxies*, por exemplo, fornece a possibilidade de indivíduos poderem utilizar a grande rede de forma a tornarem extremamente difícil a identificação do seu verdadeiro endereço IP (*Internet Protocol*). No entanto, muitas outras técnicas de ofuscamento de endereço IP fonte podem ser utilizadas. O Tor (*The Onion Router*) (DINGLEDINE, 2007), por exemplo, cria uma rede mundial de *proxies* onde cada pacote é encaminhado para o *proxy* seguinte e, usando criptografia, o IP origem do pacote é escondido. O roteamento entre esses *proxies* é alterado constantemente e cada nodo é um computador de um usuário que voluntariamente aderiu ao projeto. Utilizando o TOR, fica praticamente impossível verificar o endereço origem de uma tentativa de invasão.

Também contribuindo para os criminosos há inexistência de uma legislação que possa ser aplicada à Internet de todos os países conectados. Uma atitude que é condenada em uma determinada nação não necessita o ser em outra. O atacante pode utilizar um computador situado em um país que não tenha grandes tradições na contenção desse tipo de infração. Nesse caso, mesmo que se consiga saber exatamente de qual computador partiu o incidente, não há como obter-se maiores informações sobre o operador de tal máquina fonte da ameaça virtual (SCHNEIER, 2000).

Devido à grande dificuldade em encontrar os responsáveis pelos ataques, a necessidade de evitá-los torna-se ainda mais importante. A cada dia surgem novas ferramentas e técnicas que, pela utilização de diferentes métodos, prometem aumentar a segurança dos sistemas monitorados. Criptografia, *firewall*s, cerificados digitais, VPNs, biometria, honeypots e IDS formam parte do arsenal que pode ser utilizado para aumentar a credibilidade nas transações feitas na rede mundial (BENSO, 2003).

O *firewall* é o mecanismo de segurança mais aplicado (BERNARDES, 2000). Ele, geralmente, é interposto entre a rede privada e a Internet no intuito de servir de barreira às tentativas de ataques através do controle do acesso aos recursos de hardware e software da rede privada.

No entanto a utilização de um *firewall* ainda permite muitos tipos de incidentes de segurança. O crescimento no número de ataques internos às redes feitos por usuários mal intencionados ou pela má utilização e desinformação do usuário final, impõe que sejam empregadas outras tecnologias integradas ao funcionamento do *firewall*. Um exemplo dessas outras tecnologias são os sistemas detectores de intrusão.

No intuito de verificar as atividades que podem ser assinaturas de ataques, os sistemas detectores de intrusão possuem sensores que analisam o comportamento dos ambientes monitorados. Alguns desses sensores têm a capacidade de aprender sobre o normal funcionamento da rede para, em seguida, poderem detectar alguma anormalidade (VIRTI, 2005).

Outra possibilidade é utilizar *honeypots* como sensores de IDSs. Os *honeypots* são computadores configurados especificamente para sofrerem tentativas de invasões. Devido à característica de que todo o pacote destinado a um desses computadores é, por definição, malicioso (KREIBICH, 2004), a integração entre *honeypots* e sistemas de detecção de intrusão torna o IDS mais sensível às tentativas de ataque que ocorram internamente às redes (VIRTI, 2006-a).

Na atuação dos sensores outra possibilidade é embasar o seu funcionamento no protocolo SNMP (*Simple Network Management Protocol*) (CASE, 1990). Utilizando esse protocolo já padronizado, o sistema ganha em portabilidade e facilidade de implementação. As informações necessárias ao IDS podem ser armazenadas e disponibilizadas nos grupos de objetos gerenciáveis das diversas MIB (*Media Information Base*) padronizadas, assim como as MIB RMON1 (WALDBUSSER, 1997) e RMON2 (WALDBUSSER, 1997) (MENEGHETTI, 2002).

1.1 Objetivo

O objetivo deste trabalho é a construção de um sistema detector de intrusão embasado em informações fornecidas pelo protocolo SNMP, utilizando lógica difusa para o aprendizado do comportamento peculiar a cada rede. Diante da intervenção do administrador de segurança nas notificações fornecidas pelo sistema, o software será capaz de aprender sobre os eventos futuros, diminuindo o número de falsos positivos e falsos negativos comuns aos IDSs.

Em seguida, pelo monitoramento de uma rede de produção, será avaliada a melhora na segurança obtida com o uso do IDS proposto por esse trabalho. O sistema detector de intrusão que será implementado deverá atuar como mecanismo complementar à segurança da rede monitorada, não dispensando a adoção de outras ferramentas que tenham outras características de funcionamento.

1.2 Organização do Trabalho

O capítulo 2 apresenta os alguns conceitos da segurança computacional. Nesse capítulo são verificados os principais tipos de ataques, bem como as possíveis formas para a suas detecções. No capítulo 3 são analisados os conceitos e as características inerentes aos sistemas detectores de intrusão. O capítulo 4 descreve as peculiaridades da

utilização de *honeypots* como mecanismos de segurança e sua importância no uso como sensor de um IDS. O Capítulo 5 fornece a teoria e os embasamentos para o uso do protocolo SNMP e da MIB RMON2 na construção de IDSses. O capítulo 6 aborda o uso da lógica difusa para a inferência das decisões originadas dos dados obtidos através do protocolo SNMP. O capítulo 7 apresenta a modelagem do IDS proposto por este trabalho. O capítulo 8 descreve os testes realizados com este IDS aplicado sobre uma rede em produção. O capítulo 9 apresenta as conclusões deste trabalho.

SEGURANÇA

A atual situação da Internet ressalta a importância da segurança nos sistemas computacionais. Os prejuízos causados pelas fraudes somam altos valores e direcionam a atenção das corporações conectadas à grande rede. Só no Brasil, calcula-se que em 2006 foram gastos 300 milhões de reais em fraudes na Internet (FRAUDE, 2007). Se por um lado a rede mundial tornou-se indispensável à forma de vida moderna, por outro demonstrou que é necessário muito cuidado com a proteção dos sistemas a fim de diminuir as perdas com incidentes de segurança.

Embora o popular conceito de segurança esteja relacionado apenas à confidencialidade, ele é bem mais abrangente. De acordo com muitos autores, a segurança pode ser dividida em três propriedades: confidencialidade, integridade e disponibilidade (RUSSEL, 1991) (BASHAH, 2005) (FERREIRA, 2003). Em um dado ambiente, um aspecto pode ser mais importante do que o outro. A avaliação do tipo de segurança necessária para uma organização deve influenciar na tomada de decisões sobre projetos, técnicas, e produtos necessários para a característica visada. Abaixo há a descrição de cada uma dessas três propriedades.

- *Confidencialidade:* uma informação não pode ser revelada sem que exista expressa autorização para isso (RUSSEL, 1991). Esta característica é a mais conhecida por estar relacionada a muitos incidentes de segurança. Sempre que um atacante consegue invadir um sistema computacional ela é infringida. Em algumas instituições essa propriedade é fundamental. Nos sistemas bancários, por exemplo, a confidencialidade é necessária para que as informações sobre transações financeiras sejam apresentadas apenas às pessoas autorizadas.
- *Integridade:* um sistema seguro precisa manter a contínua integridade das informações nele armazenadas. A integridade classifica os sistemas que não corrompem a informação ou permitem qualquer tipo de modificação não autorizada nos dados armazenados.
- *Disponibilidade:* um sistema seguro precisa tornar as suas informações disponíveis para os seus usuários. Um sistema que tem alta disponibilidade consegue permanecer em operação por muito tempo e, caso ocorra algum problema, sua recuperação deve ser dada de forma rápida (FERREIRA, 2003).

De acordo com Russel (1991), existem três elementos básicos na operação da segurança: vulnerabilidades, ameaças e mecanismos de segurança. A vulnerabilidade é o ponto onde um ambiente está suscetível a ser atacado. A ameaça é o indício de um perigo que tem a capacidade de comprometer um sistema computacional. Esse perigo pode ser causado por uma pessoa específica (um hacker ou um espião) uma coisa (falha

em um equipamento) ou um evento (fogo ou enchente). Qualquer ameaça só pode comprometer um ambiente se esse tiver alguma vulnerabilidade que o permita. Os mecanismos de segurança atuam como ferramenta para a proteção de sistemas, a fim de dificultar as ações das ameaças sobre as vulnerabilidades.

1.3 Vulnerabilidades

Vulnerabilidades são as falhas ou as características inerentes a cada sistema que permitem que uma ameaça possa comprometer o seu normal funcionamento (RUSSEL, 1991). Como não existe ambiente completamente seguro, todos os sistemas de computadores são vulneráveis a ataques. Políticas de segurança e ferramentas específicas podem reduzir a probabilidade de uma tentativa de invasão poder ultrapassar os mecanismos de defesa ou, ainda, podem exigir que o atacante precise investir muito mais tempo e recursos na tentativa de invasão. No entanto, existem outros tipos de vulnerabilidades que também necessitam ser combatidas.

As vulnerabilidades podem ser classificadas de acordo com a natureza do risco que representam. Abaixo estão apresentadas aquelas mais comumente encontradas nos sistemas em produção.

- *Vulnerabilidades físicas*: classifica aqueles sistemas onde o acesso às instalações físicas é vulnerável. Os invasores, neste caso, podem desde roubar discos rígidos até destruírem por completo os computadores. Guardas, vigilância e sistemas de biometria diminuem esse tipo de vulnerabilidade.
- *Vulnerabilidades Naturais*: computadores são completamente vulneráveis a desastres naturais e ameaças ambientais. Fogo, enchente, terremotos e raios podem arruinar o sistema e destruir as informações neles armazenadas.
- *Vulnerabilidades de Hardware e software*: certos tipos de falhas de hardware e de software podem comprometer a segurança de todo o sistema. Como exemplo de uma falha de hardware que afeta a segurança do software, no contexto de sistemas operacionais, pode ser citada a obtenção de acesso a partes reservadas de memória por um processo não autorizado. Se a segurança de memória falhar, um processo pode assumir um privilégio que não teria direito se o hardware tivesse se comportado da maneira esperada.

  As falhas de software também podem abrir brechas para que um sistema possa ser comprometido. Devido à facilidade na exploração desse tipo de vulnerabilidade, as falhas de software são constantemente atacadas. Ainda que fabricantes de software invistam muitos recursos para tentarem tornar mais seguros os seus produtos, novas vulnerabilidades de software são encontradas diariamente (HOWARD, 2003).
- *Vulnerabilidades de comunicação*: se o computador estiver conectado a uma rede, há um enorme aumento na insegurança. Mensagens podem ser interceptadas, encaminhadas de maneira errada e forjadas. Linhas de comunicação que interligam computadores ou conectam terminais com seus computadores centrais podem ser escutadas ou fisicamente danificadas.
- *Vulnerabilidades humanas*: as pessoas que administram e usam os computadores representam a principal vulnerabilidade (RUSSEL, 1991). Mesmo que um administrador de uma rede corporativa seja um técnico muito competente, dificilmente poderá impedir que um usuário desavisado

possa tomar atitudes que comprometam a segurança de tal ambiente. Além da possibilidade de um usuário autorizado decidir atacar a própria rede, pessoas podem ser subornadas ou coagidas a fornecerem senhas de acesso, abrirem portas ou realizarem quaisquer ações que coloquem em risco a segurança do sistema.

1.4 Ameaças:

As ameaças são os potenciais perigos à segurança de um ambiente. Caso haja uma ameaça, ela pode utilizar das vulnerabilidades para corromper a confidencialidade, integridade ou disponibilidade de dados. Segundo Russel (1991), as ameaças podem ser divididas em três categorias:

- *Naturais e físicas:* fogo, enchentes, terremotos, desabamentos, etc. É a ameaça existente em todos os tipos de equipamentos. Pode-se prevenir para que ela não destrua o sistema, mas, geralmente, não se pode prever quanto esses acontecimentos naturais irão ocorrer.
- *Não-intencionais:* estas são as ameaças causadas pelo desconhecimento das pessoas envolvidas na operação de um sistema. Um usuário ou administrador que ignore as medidas de segurança e opere o sistema de forma desprotegida, pode vir a comprometer a segurança de toda uma rede. Muitas vezes, um simples descuido na manutenção se servidores, por exemplo, é capaz de entregar aos atacantes informações valiosas. Isso acontece quando, por exemplo, um administrador gerencia seus computadores utilizando, para a operação remota, protocolos não criptografados. Caso o atacante consiga senhas de acesso aos sistemas do administrador, os mecanismos de segurança dificilmente conseguirão distinguir entre o acesso criminoso e o real gerenciamento do administrador.
- *Intencionais:* são as ameaças causadas por pessoas que querem, de algum modo, utilizar os recursos do sistema para obterem algum ganho. Neste caso se enquadram todos os criminosos virtuais. É sobre esse tipo de ameaça que o mecanismo de segurança descrito neste trabalho atua.

1.5 Mecanismos de Segurança

Os mecanismos de segurança têm a função de evitar que as ameaças possam corromper os ambientes computacionais. Sua importância é elevada na medida em que os sistemas vão se tornando críticos e vitais para o funcionamento das organizações.

Existem muitos estudos de técnicas para criar mecanismos de segurança que atuem automaticamente, agindo de forma rápida e correta no bloqueio de ações que visem corromper os ambientes protegidos. Essa automatização é extremamente necessária, pois sabe-se que a Internet é um meio hostil em que os sistemas nela conectados sofrem constantes ataques (VIRTI, 2006-b), não sendo possível que administradores de segurança sejam consultados na ocorrência de todas as tentativas de invasão. É nesse contexto que os mecanismos de segurança operam. Pela sua forma de funcionamento, auxiliam os administradores na tentativa de bloquear ao menos os ataques mais conhecidos ou mais fáceis de serem barrados.

Para a criação de um mecanismo realmente eficaz é necessário verificar as formas utilizadas pelos criminosos para fazerem seus ataques. A próxima seção analisa as principais técnicas utilizadas pelos criminosos virtuais.

1.6 Técnicas de Invasão e de Descoberta de Informações.

Para alguns autores (HATCH, 2002) (MCCLURE, 2000) os ataques podem ser divididos em três etapas: mapeamento de vulnerabilidades, invasão e pós-invasão. No entanto, muitas técnicas utilizadas pelos criminosos virtuais podem ser rotuladas como uma junção dessas três etapas. Atualmente, alguns softwares maliciosos verificam as vulnerabilidades existentes em sistemas conectados à Internet e as utilizam para realizar outras invasões, fazendo o computador alvo se conectar a um servidor de IRC (*Internet Relay Chat*) no intuito de ser administrado remotamente por um atacante. Neste caso, pode-se dizer que houve um misto entre mapeamento de vulnerabilidade, invasão e pós-invasão. Como meio de analisar as principais técnicas utilizadas para o comprometimento de sistemas, esta seção apresenta as formas mais comuns de incidentes e alguns métodos utilizados pelos infratores na tentativa de obter o controle dos sistemas atacados.

Muitas diferentes estratégias são utilizadas para a criação de ataques bem sucedidos. No cerne de cada variante está a tentativa da obtenção de informações que sejam vitais para a segurança do sistema vítima. Abaixo estão apresentadas algumas técnicas para a obtenção de informações importantes à segurança dos sistemas envolvidos. Alguns desses métodos, utilizados separadamente são pouco eficazes. No entanto, usados em conjunto, podem fornecer ao criminoso a possibilidade de sucesso em suas tentativas.

- *Pesquisas em listas de discussão, e-mails ou grupos de notícias*: É possível obter-se informações preciosas apenas lendo listas de e-mails. Muitas vezes, os usuários ou os próprios administradores das redes, postam mensagens tentando resolver problemas em seus ambientes. Junto com esses e-mails, seja pelo seu conteúdo, seja pelos seus cabeçalhos, acabam passando informações críticas para a segurança da instituição a que pertencem (HATCH, 2002).

  De posse desses dados, o atacante pode utilizar de outras técnicas de mapeamento ou de invasão para poder comprometer o sistema vítima. Os mecanismos de segurança não podem, é claro, impedir que um usuário revele informações críticas sobre a segurança da rede. No entanto, têm a possibilidade de verificar comportamentos estranhos provindos do ataque disparado pelo invasor ou originados de outros métodos de análise de vulnerabilidades também utilizados.

- *Farejamento de tráfego local*: também chamada de *packet sniffing*, essa técnica analisa o tráfego de rede em busca de senhas ou informações críticas à segurança dos sistemas envolvidos. A dificuldade neste método encontra-se em fazer o computador *sniffer* receber os pacotes que serão monitorados.

  No caso do computador sob os comandos do criminoso e os sistemas "farejados" estarem conectados por um dispositivo repetidor (*hub*), uma vez que os computadores ligados a esse dispositivo de rede recebem todos os pacotes trafegados, o atacante apenas precisa colocar a interface de rede em modo promíscuo para que possa analisar todo o tráfego. Esta forma é chamada de farejamento passivo (*passive sniffing*) (ERICKSON, 2003).

Já no caso de estarem ligados por um comutador (*switch*), a operação é mais trabalhosa. A vantagem de um ambiente comutado é que o dispositivo entrega para cada computador apenas os pacotes a este destinados, significando que as máquinas que operam em modo promíscuo não conseguem farejar nenhum pacote adicional. No entanto, existem algumas formas de bisbilhotar os pacotes de outros computadores, mesmo nesse tipo de ambiente. As peculiaridades dos protocolos envolvidos precisam ser combinadas e examinadas para encontrar uma forma de burlar a operação do comutador.

Um importante detalhe nas comunicações de rede é que o endereço IP fonte pode ser manipulado para diferentes propósitos. Não há garantias de que ele seja o real endereço do computador origem do pacote. A ação de forjar o endereço fonte, conhecida como *spoofing*, aumenta o número de ataques possíveis, pois muitos sistemas e protocolos fazem inferências embasadas no endereço fonte, ignorando que o mesmo possa ser falsificado.

Forjar o endereço fonte é o primeiro passo no farejamento de pacotes em uma rede comutada. Outros dois interessantes detalhes são encontrados no protocolo ARP (*Addres Resolution Protocol*). Quando um computador precisa enviar um pacote a outro presente no mesmo segmento de rede, é necessário conhecer o número MAC (*Media Access* Control) da máquina destino do pacote. Para isso utiliza-se o ARP, que, através de perguntas (*ARP requests*) e respostas (*ARP reply*), realiza a associação entre os endereços IP e os números MAC.

Primeiramente, quando um *ARP reply* vem com um endereço IP fonte que já exista na memória ARP, o sistema receptor irá sobrepor o primeiro número MAC pelo novo endereço encontrado, mesmo se a entrada na tabela ARP tenha sido explicitamente marcada como permanente. O segundo detalhe do ARP é que os sistemas aceitam um *ARP reply* mesmo que não tenham enviado um *ARP request*. Isto acontece porque o estado da informação sobre o tráfego ARP não é armazenado (ERICSON 2003).

Estes detalhes quando explorados de forma adequada permitem que um atacante consiga analisar o tráfego em uma rede comutada com uma técnica conhecida como “redirecionamento de ARP”. O atacante envia um *ARP reply* com IP forjado para certos computadores e dispositivos para que a memória ARP possa ser sobrescrita com os dados do atacante. Esta técnica é conhecida como “envenenamento de memória ARP” (*cache ARP poisoning*). Como forma de interceptar pacotes de uma comunicação entre dois computadores, A e B, o atacante precisa envenenar a memória ARP de A para que A acredite que o endereço MAC de B seja o MAC do atacante. Para B, é necessário fazer semelhante operação. Assim a máquina do atacante precisa simplesmente repassar estes pacotes para o seu destino apropriado, e todo o tráfego entre A e B continuará a ser entregue, mas ele sempre passará pela máquina do atacante (figura 2.1).

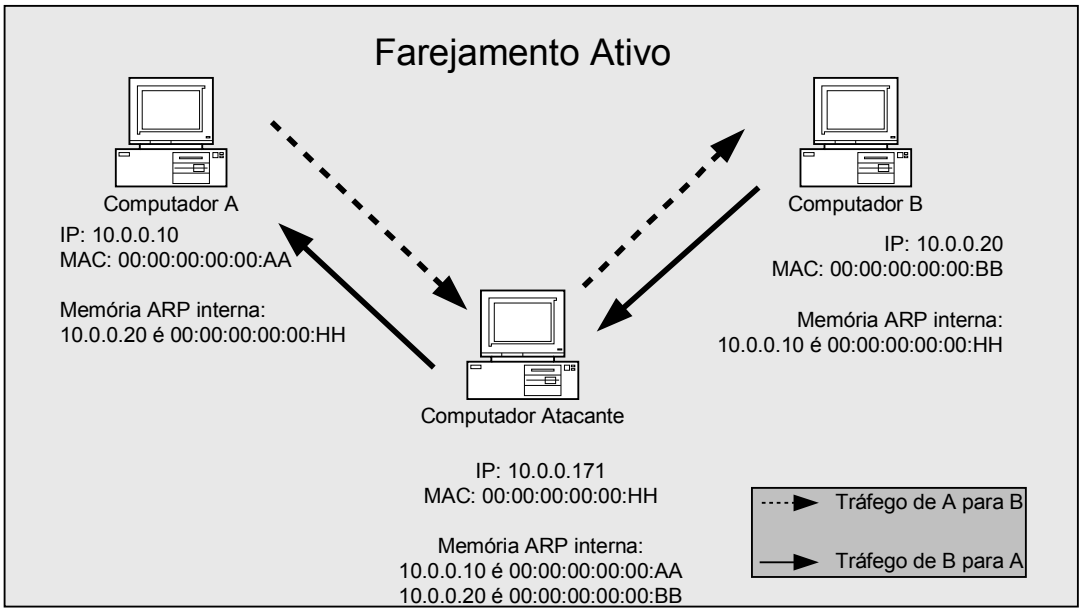

Figura 2.1: Funcionamento do farejamento ativo (*active sniffer*)

Por causa dos tempos máximos para as informações constantes na tabela ARP de cada sistema, a vítimas se atualizam periodicamente, enviando os *ARP requests* e recebendo os *ARP responses*. Para manter o ataque de redirecionamento, o atacante apenas precisa continuar enviando seus *ARP reply* forjados.

A técnica de redirecionamento de ARP é bastante empregada para analisar-se o tráfego direcionado à Internet proveniente de um computador presente no mesmo segmento de rede que o sistema monitor. Para isso, o atacante precisa interpor-se na comunicação entre o computador vítima e o *gateway* da rede local (ERICKSON, 2003).

A detecção dos farejamentos passivos pode ser feita pelo envio de pacotes IP com o número MAC do destino forjado, mas com o real endereço IP destino. Se um computador estiver operando com a sua interface de rede em modo promíscuo, ele irá absorver o tal pacote, mesmo que o endereço MAC não seja o mesmo de sua placa de rede e irá responder a esse pacote, pois o endereço IP encontrado é o seu próprio endereço. Todos os computadores que responderem a tais pacotes estarão operando em modo promíscuo.

Para os farejamentos ativos, os IDS tem a possibilidade de armazenarem a relação MAC e IP dos computadores presentes na rede supervisionada. Desta forma, quando um pacote de *ARP reply* forjado for encaminhado aos computadores da rede, o IDS tem a possibilidade de diagnosticar um ataque de redirecionamento de ARP.

- *O protocolo DNS*: O serviço de resolução de nomes (*Domain Name Service*) é vital para o funcionamento da Internet. No entanto, fornece informações importantes a respeito do endereço IP de determinados servidores de uma corporação. Se um atacante almejar penetrar nos computadores de uma dada rede, ele poderá saber quais os servidores que respondem por serviços como DNS e e-mail, por exemplo.

Associadas a essas pesquisas, podem ser feitas consultas nos serviços de registros de nomes de domínios (*Whois*). Para cada nome de domínio registrado há informações detalhadas sobre os servidores DNS responsáveis por resolverem aquele domínio específico bem como informações sobre a pessoa responsável por tal domínio. O criminoso pode utilizar as informações do *Whois* para, por exemplo, descobrir outras redes pertencentes a uma empresa que já foi invadida, no intuito de corromper outros computadores que têm uma maior probabilidade de possuírem a mesma vulnerabilidade que permitiu ao invasor comprometer a primeira rede (HATCH, 2002) (MCCLURE, 2000).

O DNS pode ser potencialmente perigoso quando permite transferência de zona a qualquer sistema que a solicitar. Uma transferência de zona possibilita que servidores primários e secundários de DNS sincronizem suas informações. Alguns servidores de DNS possuem também informações sobre os endereços relativos à rede privada. Se um atacante conseguir ter acesso a uma transferência de zona vinda de um DNS com essa característica, ele terá acesso a grande parte da estrutura da rede atacada, podendo, dessa forma, direcionar seus ataques.

Se um invasor conseguir corromper um servidor DNS de uma corporação os danos podem ser estrondosos. O criminoso tem a capacidade de forjar qualquer consulta ao DNS, direcionando para sítios falsos, por exemplo, ou para servidores que possuam os mesmos serviços almejados, mas configurados para roubo de senhas. A mesma perigosa potencialidade acontece se o criminoso utilizar da técnica conhecida por *men in the middle*, para simular as respostas do servidor de nomes. Nessa técnica, os atacantes utilizam o farejamento de tráfego para se anteciparem às respostas dos servidores, criando e endereçando pacotes com informações forjadas aos computadores requerentes.

Pouco se pode fazer para prevenir que algumas informações sobre os serviços disponibilizados na rede sejam oferecidas via DNS, uma vez que essas informações são vitais para a comunicação na Internet. A prevenção da vulnerabilidade de transferência de zona, no entanto, pode ser dada pela correta configuração dos servidores DNS, que devem permitir essa transferência a apenas determinados endereços. No que se refere aos perigos envolvidos no comprometimento de um servidor DNS, os administradores devem dispensar especial cuidado a esses tipos de servidores. Para os casos de *men in the middle*, e transferência de zona, já há a tentativa de se implantar o DNSSEC, que faz autenticação forte nas consultas enviadas ao servidor (DNSSEC, 2006).

- *Engenharia social*: o atacante pode obter informações importantes persuadindo e ludibriando pessoas para que lhe repassarem essas valiosas informações. O principal problema que torna tão eficaz esse tipo de ataque é a natural tendência do ser humano em acreditar (PEIKARI, 2004). Um ataque de engenharia social simplesmente opera com as vulnerabilidades da natureza humana. Os mecanismos de defesa das pessoas são mais fáceis de burlar que camadas de proteção de senhas, criptografias, *firewalls*, *honeypots* e sistemas detectores de intrusão. Em muitos casos os criminosos precisam apenas "perguntar".

  Um atacante pode ligar para o setor responsável pelos recursos humanos de uma empresa identificando-se como um determinado funcionário com uma matrícula específica. De posse dessas informações é possível que consiga que uma nova senha de acesso aos sistemas seja fornecida via telefone. A exploração desse tipo de vulnerabilidade é também influenciada pelo fato de não requerer

grandes conhecimentos técnicos e poder ser aplicada sem muitos recursos computacionais.

Além de campanhas internas nas corporações, outra forma de lidar com essa vulnerabilidade é contratar profissionais para testar a segurança da empresa contra ataques de engenharia social. Eles podem verificar os pontos fracos da corporação para, posteriormente, construir-se um trabalho de treinamento e reciclagem direcionado a esses pontos (PEIKARI, 2004).

É particularmente muito difícil para mecanismos de segurança detectarem um atacante que, através da engenharia social, conseguiu obter um usuário e uma senha válidos, uma vez que o atacante estará corretamente autenticado. Este tipo de ataque só será detectado se o atacante cometer algum erro: varrer outros computadores ou tentar acessar endereços de *honeypots*, por exemplo.

- *SNMP*: este protocolo foi concebido para auxiliar os administradores no gerenciamento de suas redes. Seus dados são valiosos aos atacantes que conseguirem a eles terem acesso. O problema relacionado ao SNMP é que alguns softwares e dispositivos de rede operam utilizando a primeira ou a segunda versão deste protocolo (CORCHADO, 2005). Estas versões não têm suporte a criptografia e trafegam senhas de leitura e escrita em texto claro. O capítulo 5 apresenta as peculiaridades das três versões do SNMP existentes.

- *Varreduras*: fazer uma varredura significa percorrer sistemas à procura de informações sobre os serviços disponibilizados, os sistemas operacionais e as vulnerabilidades encontradas em um computador (HATCH, 2002). Para esse tipo de mapeamento, muitos métodos são utilizados, entre eles a varredura ICMP (*Internet Control Message Protocol*), a varredura por portas e a varredura por falhas na segurança.

  A varredura de *ICMP echo* (*ping*) é um dos passos mais básicos para um atacante que quer verificar quais os computadores ativos em um determinado bloco de endereços IP. Ela permite que o atacante possa direcionar seus esforços no ataque a apenas endereços que estejam ativos.

  No entanto o *ICMP echo* (ICMP tipo 8) é apenas o inicio das artimanhas que se pode fazer usando tal protocolo. Com o *ICMP timestamp* (ICPM tipo 13), por exemplo, pode-se saber a hora local do sistema vítima, informação importante para saber se o sistema está bem configurado ou inferir sua localização no mundo. O *ICMP address mask request* (ICMP tipo 17) permite saber qual a máscara de rede utilizada pela vítima. Essa informação pode ser usada para determinar todas as sub-redes em uso naquela rede a fim de direcionar os ataques a essas sub-redes evitando endereços de broadcast, por exemplo.

  Se uma dada corporação bloquear o ICMP, o atacante apenas terá maiores trabalhos. No entanto, isso não impede ataques, apenas torna-os um pouco mais difíceis (MCCLURE 2000).

  A varredura por portas (*port scanning*) é o processo de se conectar a portas TCP (*Transmission Control Protocol*) e UDP (*User Datagram Protocol*) do sistema alvo, para determinar que serviços estão em execução. Identificar portas ativas é crucial para determinar o tipo de sistema operacional e aplicativos em uso. Serviços ativos podem permitir a um usuário não-autorizado ter acesso a

sistemas mal configurados que operem sob softwares com vulnerabilidades conhecidas (MCCLURE, 2000).

No passado os atacantes levavam muito tempo e precisavam ser excelentes técnicos para poderem invadir um sistema. No clássico Stoll (1990), o autor relata um incidente em que o criminoso virtual por 10 meses ficou tentando adentrar em um determinado sistema. Essa forma de ataque, hoje em dia, está se extinguindo.

Embora um atacante possa ganhar muito dinheiro se conseguir invadir um computador de um banco, por exemplo, é provável que, para esse ataque, precise ter mais conhecimento técnico e necessite mais tempo. Além disso, se bem sucedido, possivelmente não conseguirá retirar o dinheiro proveniente de seu golpe a tempo de não ser pego (SCHNEIER, 2000).

A opção por invadir máquinas menos protegidas é bem mais vantajosa. O atacante não necessita grandes conhecimentos técnicos para automatizar os ataques. Criando ou utilizando programas de busca por computadores vulneráveis na Internet, é possível, em minutos, ter uma lista com alguns computadores invadidos. Se o atacante continuar sua varredura, ao final de alguns dias, poderá ter algumas centenas de sistemas com os quais poderá almejar tantos ganhos financeiros quanto aqueles que teria se houvesse invadido um sistema de banco.

Mesmo com toda a repercussão das notícias sobre danos ocasionados por incidentes de segurança, a procura por serviços mal configurados, que operem sob vulnerabilidades conhecidas, ainda é muito grande. No trabalho (VIRTI, 2006-a) foi feita uma pesquisa utilizando-se mais de 65 mil *honeypots* para determinar o perfil das tentativas de ataque a sistemas expostos à Internet. Ele demonstrou que as buscas por serviços de compartilhamento de arquivos do sistema operacional Microsoft Windows ainda lidera no número de tentativas de ataque. Ou seja, mesmo sabendo-se que liberar qualquer serviço de compartilhamento de arquivos para a Internet é muito perigoso, ainda existem sistemas que apresentam esse tipo de vulnerabilidade. O mesmo trabalho demonstra outro exemplo interessante. No dia 22 de junho de 2005 foi descoberta uma vulnerabilidade no software Veritas Backup que opera pela porta 10000/TCP. Dois dias depois alguns indicadores no mundo (DSHIELD, 2005) (CONSORCIO, 2007) (HONEYNET, 2007), já acusavam uma grande intensidade nas varreduras destinadas a tal porta TCP. Isso ocorre porque, mesmo que ainda não existisse um software específico para se beneficiar da nova vulnerabilidade, os atacantes, enquanto esperavam pelo referido software, já trataram de delimitar quais computadores no mundo seriam as potenciais vítimas do ataque. Deve-se levar em consideração também que um computador responsável pelo serviço de backup, geralmente, possui alto poder computacional e boa largura de banda. Se uma máquina com tal poder for comprometida, ela será bem mais valiosa aos atacantes que outros sistemas de menor porte.

Existem softwares específicos que, à procura de vulnerabilidades conhecidas, têm a característica de fazer massivas varreduras em um dado computador na Internet. Um exemplo desse tipo de software é o Nessus (DERAISON, 2007). Além de fornecer as falhas na segurança dos computadores

vítimas, ele informa onde pode-se obter maiores dados sobre essas vulnerabilidades.

As varreduras podem ser detectadas e barradas por mecanismos de segurança. Dependendo da velocidade com que este ataque é disparado contra os computadores monitorados, os IDS podem verificar a anomalia e alertar os administradores de segurança ou, até mesmo, realizar ações automáticas como o fechamento de *firewalls* e a mudança de rotas na tentativa de barrar o incidente. No entanto, se os ataques forem feitos de forma suficientemente lenta, como pelo envio de 1 pacote por minuto, dificilmente os IDS detectarão alguma atividade ilegal.

- *Negação de serviço*: também conhecidos pela sigla DoS (*Denial of Service*) esses ataques impedem que um recurso ou um serviço possa ser acessado. Existem duas formas gerais nesse tipo de incidente: aqueles que inundam os serviços e aqueles que corrompem os serviços disponibilizados (ERICSON, 2003).

  Ataques que corrompem serviços são, na verdade, mais parecidos com exploradores (*exploits*) de vulnerabilidades. Devido a alguma falha de segurança específica, um serviço pode ser atacado e comprometido, ficando inacessível até que seus administradores tomem alguma providência. Um exemplo desse tipo de vulnerabilidade foi o "ping da morte", que, pelo envio de um pacote ICMP *echo request* com tamanho superior a 65.535 bytes, corrompia o normal funcionamento de alguns ambintes vulneráveis. Outro exemplo foi o ataque conhecido como *Teardrop*. Esse ataque danificava a pilha TCP/IP do computador vítima fornecendo a ela vários fragmentos de pacotes IP que não eram corretamente reagrupados, tendo como resultado uma instabilidade geral que, em alguns sistemas operacionais, levava a sua reinicialização. Esses exemplos, embora não sejam mais aplicados à maioria dos sistemas operacionais, demonstram que uma eventual descoberta de uma nova vulnerabilidade pode fazer os serviços disponibilizados à Internet serem corrompidos com facilidade.

  Os ataques de inundação não ensejam necessariamente corromper os serviços fornecidos por um dado servidor, mas almejam torná-los inacessíveis temporariamente. Uma forma de realizar esse tipo de ataque é utilizar toda a largura de banda do computador alvo através do massivo envio de pacotes IP. Essa técnica exige que o computador do qual o ataque foi disparado possua maior largura de banda que aquele vítima (ERICKSON, 2003). No entanto, mesmo se o atacante não tiver esse recurso a sua disposição, ele poderá realizar o ataque utilizando uma técnica conhecida por *smurf*. Para isso é necessário encontrar uma rede que consiga responder a pacotes broadcast. Uma rede assim configurada tem um poder de amplificar um ataque feito com pacotes que contenham o IP fonte forjado. Criando pacotes IP com o endereço fonte do computador vítima, o atacante pode enviá-los para os endereços broadcast dessas redes, fazendo com que, em resposta, toda a rede responda ao computador vítima.

  Outra possibilidade de realizar um ataque DoS por inundação, diferentemente daquela que pretende consumir toda a banda disponível, é feita pelo esgotamento de recursos do hardware e software do servidor atacado. Uma maneira de criar um ataque com esse objetivo é explorar a forma como o

protocolo TCP inicia conexões. O massivo envio de requisições de abertura de conexão (pacotes SYN) ao computador vítima pode levá-lo a consumir todos os seus recursos. Isso acontece porque, para cada pacote SYN recebido, o TCP consome memória, enviando um pacote SYN/ACK ao requerente e aguardando por uma confirmação que, no caso de um ataque, nunca virá (HATCH, 2002).

Caso um computador ou poucos computadores simultaneamente tentem atacar os sistemas de uma dada corporação, é possível entrar em contato com o seu provedor de acesso Internet para que realizem o bloqueio do tráfego originado desses computadores. No entanto, quando o ataque provém de centenas ou milhares de máquinas localizadas em diferentes partes do mundo, não há muito a se fazer. Essa técnica conhecida por DDos (*Distributed Denial of Service*), é um grande problema para qualquer sistema conectado à Internet. Atualmente, com a descoberta de redes de computadores zumbis com milhões de computadores, os criminosos que operam-nas, podem fazer grandes estragos, mesmo a sítios com enormes recursos (CAIS, 2007).

- *Vírus Vermes e Cavalos de Tróia:* são softwares maliciosos (*malwares*) que tem a capacidade de comprometer a segurança dos sistemas onde forem executados. Os Cavalos de Tróia (*Trojans*) são programas autônomos, que não podem propagar a si mesmo, instalados em um computador pelo auxílio de um usuário que executou algum software contaminado. Geralmente esses cavalos de tróia abrem portas para que o computador possa ser administrado remotamente (RUSSEL, 1991). Os vírus diferem dos Cavalos de Tróia por terem a capacidade de se auto-replicarem. Tanto os vírus quanto os cavalos de tróia não possuem a capacidade de se propagarem a outras máquinas sem a ajuda de um usuário, pelo envio de um e-mail com um anexo contaminado, por exemplo (HATCH, 2002).

  Os vermes são programas que têm a capacidade de se replicar tanto na máquina local quanto em outros computadores da rede, automaticamente. Eles conseguem penetrar nas ferramentas de defesa dos computadores ligados em rede pela exploração de vulnerabilidades conhecidas.

  Muitos programas maliciosos, no entanto, são um misto desses três tipos de softwares. Alguns vírus, por exemplo, além de se replicarem no interior da máquina local, também abrem portas, para que o sistema possa ser administrado remotamente, e tentam se propagar na rede enviando cópias de si mesmo para endereços de e-mail encontrados no computador (HATCH, 2002).

Cada uma dessas técnicas apresentadas exige, para ser bem sucedida, diferentes conhecimentos e diferentes necessidades de recursos computacionais. As estratégias adotadas pelos atacantes estão relacionadas aos pontos onde presumem que a segurança é passível de ser explorada para uma invasão.

Muitos mecanismos têm sido planejados e desenvolvidos como uma forma automática de garantir a segurança das redes onde são aplicados. A melhora na segurança de uma rede privada pode ser alcançada quando esta operar sob a ação conjunta de diferentes mecanismos de segurança, como biometria, criptografia, *firewalls*, honeypots e IDS (BENSO, 2000). O próximo capítulo analisa os principais aspectos relacionados à classificação, estrutura e operação dos sistemas detectores de intrusão.

SISTEMAS DE DETECÇÃO DE INTRUSÃO

Os estudos sobre os sistemas de detecção de intrusão já têm mais de 25 anos. Um dos primeiros trabalhos sobre o assunto, intitulado *Computer Security Threat Monitoring and Surveillance*, foi publicado em 1980 por James Anderson. Sete anos depois, Dorothy Danning escreveu o artigo *An Intrusion Detection Model*, criando um modelo onde, pela percepção de que as atividades maliciosas poderiam ser refletidas nos parâmetros monitorados, tentava criar um sistema que conseguisse detectar essas anomalias (SHAH, 2003).

Uma intrusão pode ser definida como o uso não autorizado, mau uso ou abuso feito por usuários conhecidos ou invasores desconhecidos em um determinado sistema computacional (TILLAPART, 2002). Outra definição se foca nos conceitos de segurança: uma intrusão é um conjunto de ações que visa comprometer a integridade, confidencialidade ou a disponibilidade dos dados ou do sistema (FERREIRA, 2003).

A detecção de intrusão é o ato de identificar indivíduos que estão utilizando um sistema computacional sem autorização e também aqueles legítimos, ou seja, autenticados, que abusam dos seus privilégios (TILLAPART, 2002). Esta definição referencia os perigos dos invasores internos. Os usuários conhecidos do sistema, que muitas vezes estão dentro da própria estrutura de rede local, mas que por algum motivo, decidem atentar contra a segurança dos sistemas, podem ser grandes geradores de incidentes. Outra definição faz referência à análise das políticas de segurança: a detecção de intrusão é o processo de, inteligentemente, monitorar eventos ocorrendo em um sistema computacional ou rede e os diagnosticar por meio de violações das políticas de segurança (IDRIS, 2006). Essa segunda definição, pela menção do termo "inteligentemente", traz a idéia do uso de sistemas de inteligência artificial para melhor conduzir o processo de detecção de intrusão.

Os muitos métodos e mecanismos existentes para a proteção de sistemas computacionais podem ser guiados por diferentes abordagens. Em Benso (2003) são listadas seis abordagens para a prevenção às intrusões:

- *Prevenção*: prever ou coibir uma intrusão. Neste caso enquadram-se os *firewalls* onde, pelo filtro de pacotes, consegue-se prevenir que algumas tentativas de invasão cheguem a computadores por ele protegidos.
- *Preempção:* agir contra o possível atacante antes que ele ataque o seu sistema. Existem IDS que fornecem esse recurso (IDRIS, 2006), no entanto, devido a possibilidade de que seja forjado o IP fonte dos ataques, essa medida pode fazer o IDS ser utilizado para organizar outros incidentes.
- *Persuasão:* persuadir o atacante a não executar o ataque ou mesmo suspender um possível ataque em andamento. Muitas vezes, diante de

ataques de distribuídos de negação de serviço, por exemplo, essa persuasão pode se dar através de quantias monetárias fornecidas aos invasores.

- *Ilusão*: iludir o intruso, fazendo-o pensar que o ataque foi bem sucedido. É a técnica utilizada nos *honeypots* (capítulo 4).
- *Detecção*: identificar tentativas de invasão do sistema e acionar os mecanismos apropriados. Os IDS são enquadrados nessa abordagem.

As detecções feitas pelos sistemas detectores de intrusão permitem que organizações protejam seus sistemas das ameaças existentes na grande rede. Dada a natureza e o nível dessas ameaças, muitas vezes, a questão para os administradores não é se devem utilizar um detector de intrusão e sim quais os recursos e capacidades de deve ter o IDS utilizado (BACE, 2001). Neste contexto, os IDS têm ganhado aceitação pela sua introdução em um grande número de infra-estruturas de segurança. No entanto, para justificar os gastos na implantação de um desses sistemas em uma rede corporativa é necessário verificar as razões técnicas para a sua implantação (BACE, 2001):

- Prevenir comportamentos maliciosos e punir os envolvidos nos abusos cometidos aos sistemas monitorados;
- Detectar violações na segurança que não são verificadas por outros mecanismos;
- Detectar e agir contra os sintomas iniciais de ataques em construção;
- Documentar a existência de ameaças aos sistemas monitorados;
- Providenciar informações usuais sobre intrusões ocorridas, permitindo uma melhora no diagnóstico, recuperação e correção dos fatores causadores dos incidentes.

A identificação das intrusões não é uma tarefa simples. As redes atuais são extremamente complexas, interligando sistemas operacionais diferentes, com diversos serviços e operando sobre muitos protocolos. A velocidade com que novos protocolos e aplicações surgem e são incorporados torna ainda mais difícil a tarefa de detectar intrusões (SCHNEIER, 2000).

Os principais problemas encontrados nos IDS estão relacionados à precisão na distinção entre eventos normais e aqueles indicativos de uma efetiva intrusão (KABIRI, 2005). O IDS deve ser eficaz a ponto de não gerar alarmes para situações normais (falso-positivo), ou deixar de gerá-los para situações de intrusão (falso-negativo). Esta é a razão porque, em muitos casos, os IDS são usados para reagir a ataques apenas se estiverem operando em conjunto com especialistas humanos. Caso houvesse uma maior confiança nas decisões dos sistemas de detecção, a automatização de respostas como o bloqueio de *firewalls*, por exemplo, não seria temida. Há o medo de que as conseqüências das reações dos IDS sejam tão danosas quanto as ações dos próprios ataques.

Dessa forma, muitos sistemas de detecção de intrusão atuam como ferramentas de auxílio ao trabalho de seres humanos, não sendo confiáveis a ponto de poderem ter sua operação automatizada. A razão disso é a sua inabilidade em operar com ataques desconhecidos ou com aqueles conhecidos, mas que sofreram alterações em seus padrões (KABIRI, 2005).

A operação de um IDS não deve interferir em muito sobre o normal funcionamento dos sistemas monitorados. No entanto, devido ao grande número de transações e

variáveis a serem analisadas, eles frequentemente consomem grande poder de processamento e banda de rede na tentativa de criar ambientes onde consigam ter condições de corretamente detectar intrusões.

O tempo envolvido na análise, identificação e envio de alertas aos administradores responsáveis pela segurança também deve ser considerado. O ideal para um sistema de detecção de intrusão é que o reconhecimento seja em tempo real para que ações de contenção e identificação de envolvidos possam ser realizadas a tempo de evitar maiores danos aos sistemas alvos (BENSO, 2003).

Para Ferreira (2003) as seguintes características são desejáveis no comportamento e operação de um sistema detector de intrusão:

- Deve funcionar continuamente, sem interação humana, e ser seguro o suficiente para que sua operação em segundo plano não comprometa a operação dos sistemas monitorados;
- Precisa ser tolerante a falhas, ou seja, a sua base de conhecimento não deve ser perdida em caso de falha de algum sistema envolvido na tarefa da detecção de intrusão;
- É necessário que possa monitorar a si próprio, evitando qualquer mudança na sua base;
- Deve causar o mínimo de impacto no funcionamento da rede, sob pena de poder sobrecarregá-la e, por causa de sua própria operação, alterar os parâmetros monitorados;
- Não pode permitir a subversão do sistema, ou seja, não deve aceitar que o atacante possa modificar a operação do IDS para forçar a ocorrência de falso negativo.
- Precisa detectar o menor número possível de falsos positivos;
- Não deve gerar falso negativo;

Buscar o modelo e a construção de um sistema detector de intrusão com as características acima apresentadas é o objetivo de muitos estudos na área de segurança computacional. No entanto, alguns itens são bastantes complexos de serem satisfeitos. O objetivo de criar uma implementação que não gere falsos negativos, por exemplo, geralmente, faz o sistema estar mais propenso a falsos positivos. Para Stanger (2002), os falsos positivos podem ser causados pelos seguintes casos:

- O IDS pode não ter sido configurado adequadamente e por isso reage ao comportamento legítimo dos sistemas monitorados;
- Os dados monitorados podem ter mudado e o sistema detector não ter sido informado sobre a mudança;
- O IDS pode não possuir recursos para detectar alguns ataques.

A criação de detectores mais eficientes exige o conhecimento das técnicas de intrusão utilizadas pelos atacantes. Assim como visto no capítulo 2, uma intrusão pode possuir muitas formas e padrões diferentes e os seus estudos criaram uma importante área de pesquisa (AXELSSON, 2000).

A operação de um IDS exige que sejam conhecidos os componentes que o fazem ser capaz de determinar as atividades ilegais na rede monitorada. Para Bace (2001), os sistemas detectores de intrusão podem ser divididos em três componentes fundamentais:

- *Fonte da Informação*: referencia as diferentes fontes de informação usadas para determinar se uma intrusão está sendo criada. Estas fontes podem ser compostas de diversas formas de captura dos dados.
- *Análise*: a parte de um sistema detector de intrusão que organiza e entrega lógica a eventos derivados das fontes de informação.
- *Resposta*: o conjunto de ações que o sistema toma quando detecta uma intrusão. Tipicamente essas reações são divididas em passivas e ativas, onde as ativas envolvem algumas intervenções automáticas em partes do sistema monitorado e as passivas incluem o envio de alertas para especialistas para que tomem providências diante do ataque.

Na parte de análise de um IDS, a construção da base de conhecimento bem como a forma de inferência sobre essa base é de fundamental importância para o correto diagnóstico dos incidentes de segurança. A detecção de problemas de rede envolve tirar conclusões de um conjunto de sintomas. Tais conclusões são baseadas numa combinação de pesquisa e experiência, mas a complexidade do problema às vezes torna impossível chegar a uma conclusão exata. Dessa forma, a utilização de lógica inexata ou lógica difusa (*Fuzzy logic*) pode melhor representar a relação entre conhecimento adquirido e entradas dos sensores de rede. Com o raciocínio difuso são associados coeficientes de certeza tanto às premissas nas quais se apóiam certas conclusões como às próprias conclusões (TAROUCO, 1990).

1.7 Classificação dos IDS

Um sistema detector de intrusão automatiza o processo de detecção através de técnicas de monitoramento, visando identificar possíveis alterações nos padrões ou atividades previamente definidas como maliciosas (PEIKARI, 2004). Estes sistemas podem ser classificados quanto ao tipo da informação analisada e quanto ao modo como analisam as informações coletadas.

1.7.1 Local de Captura da Informação

O local de captura das informações que são processadas nos IDS fornece as características, capacidades e limitações que são inerentes a sua operação. Quanto à origem das informações analisadas, os sistemas de detecção de intrusão podem ser classificados da seguinte forma:

- *IDS de rede* (*Network Intrusion Detection System*): utilizam os dados capturados do monitoramento dos pacotes que trafegam pela rede onde o sistema é instalado. A análise dos dados que passam pela rede fornece a possibilidade de espalhar as detecções de intrusão a toda a rede supervisionada, independentemente de que dispositivos, sistemas operacionais e serviços estiverem implantados (KABIRI, 2005).

  O sistema detector de intrusão proposto neste trabalho é um IDS de rede, pois utiliza de agentes RMON2 para monitorar o tráfego em diferentes segmentos da rede monitorada.

- *IDS de computador* (*Host Intrusion Detection System*): monitoram o comportamento apenas do sistema (computador ou dispositivo de rede) onde é implantado. Pela análise de dados tais como arquivos de log, chamadas de sistema ou número de conexões abertas, produzem informações que permitem avaliar se o sistema está sob um ataque. Para Peikari (2004) esse tipo de verificação possui melhores resultados que a anterior, pois o IDS pode ser configurado para somente detectar os ataques bem sucedidos e não as tentativas, dessa forma, diminuindo o número de falso-positivos. No entanto, para Kabiri (2005) essa abordagem é muito limitada, pois acaba tecendo análises restritas a um computador, ignorando os incidentes compostos pelo relacionamento entre vários sistemas dentro da rede privada.

1.7.2 Forma de Detecção de Intrusão

A detecção de intrusão torna-se possível a partir da captura de dados da entidade a ser protegida. Esses dados podem ser coletados na rede ou em estruturas de controle de sistemas como logs, tabelas de processos, entre outros. As técnicas aplicadas na identificação de uma invasão analisam as informações coletadas na busca de desvios de comportamento ou padrões de ataques conhecidos.

Na classificação por modelo de análise das informações, os sistemas são identificados em:

- *Detecção por assinatura*: baseia-se na identificação de um padrão de ataque já conhecido. Os IDS que fazem detecção por assinatura têm em sua formação uma base de conhecimentos onde, através da comparação com os padrões encontrados nas variáveis monitoradas, intentam constatar se existe uma intrusão em curso (AXELSON, 2000). Esses sistemas possuem a desvantagem da necessidade de especificação prévia e qualificada das assinaturas.

- *Detecção por anomalia*: nas detecções por anomalia são procuradas as anormalidades na operação do ambiente monitorado. Segue-se a premissa de que o comportamento anormal de um sistema é, por princípio, suspeito (AXELSON, 2000). Os IDS baseados na identificação de anomalias também possuem uma base de conhecimento. A sua implementação necessita estar focada sobre como aprender a discernir entre o que pode ser classificado como anormal embasado nos valores das variáveis analisadas. Essa detecção precisa ser modelada para ser flexível, permitindo a identificação de situações que fogem ao normal padrão de comportamento. A desvantagem de IDS baseados em anomalias é que se torna difícil fazer o sistema distinguir entre o que realmente pode ser considerado um ataque e o que é apenas um pequeno desvio de comportamento. Assim, muitos desses IDS podem gerar alarmes do tipo falso positivo ou, se o sistema for configurado para gerar alarmes apenas quando o desvio for muito acentuado, ocorrer os falso-negativos. Uma possível solução para esse tipo de problema é utilizar a inteligência artificial. Muitas técnicas de IA já foram exploradas para criar IDS baseados no conhecimento do comportamento passado e no uso normal. Elas se mostraram poderosas na detecção de anomalias (IDRIS, 2006).

- *Detecções compostas*: essa classificação está ligada aos sistemas de detecção que avaliam tanto o normal funcionamento do sistema supervisionado quanto os padrões de comportamento advindos de intrusões conhecidas. Esses

detectores operam pela análise de intrusões contra uma base de conhecimentos relativos ao comportamento normal no sistema. Ao menos em teoria, eles têm melhores chances de filtrar corretamente os eventos que são interessantes à análise, uma vez que conhecem os padrões de intrusão e podem relacioná-los ao comportamento normal dos ambientes monitorados. Com esse tipo de lógica, há a criação de um IDS mais avançado que consegue, com maior eficácia, qualificar as respostas pela diminuição de alarmes falso-positivos e falso-negativos (AXELSSON, 2000). Os sistemas detectores de intrusão que operam sob essa lógica podem ser dotados de uma capacidade de auto-aprendizado dos normais comportamentos dos objetos supervisionados. Os padrões indicativos de intrusões podem ser informados ao IDS através de simulações de situações de ataque (GOMEZ, 2002-a) (LEE, 2004), através de linguagens específicas para detecção de comportamentos (GASPARY, 2002) (MENEGUETTI, 2002), ou através da criação de regras indicativas de ataque (CORDON, 2001) (DIKERSON, 2000) (DIKERSON, 2001) (TILLAPART, 2002) (SILVEIRA, 2004) (IDRIS, 2005) (IDRIS, 2006).

1.7.3 Tipo de Intrusão Detectada

Diferentes formas de detecções geram distintas informações sobre a maneira como os IDS analisam e conseguem se atualizar diante das descobertas de novas vulnerabilidades e ameaças. Segundo Axelsson (2000), de acordo com o tipo de intrusão detectada, os IDS podem ser classificados em três categorias:

- *Intrusões bem conhecidas:* são aquelas já analisadas pela comunidade de segurança. Essas intrusões são simples de serem detectadas, pois têm poucas possibilidades de variações devido ao fato de necessitarem ser bastante diretas para que a exploração da vulnerabilidade possa ser bem sucedida.
- *Intrusões mutáveis:* apesar de também serem conhecidas, esses tipos de intrusões são mais difíceis de serem detectadas, pois podem apresentar uma série de variantes. No entanto, por possuírem o mesmo cerne, a utilização de sistemas de lógicas difusas podem ajudar na sua detecção.
- *Intrusões desconhecidas:* são as principais inimigas dos IDS por assinatura. Dificilmente são detectadas por esses sistemas detectores de intrusão. Um IDS por anomalia, no entanto, mesmo não conhecendo a nova intrusão, pode detectar comportamentos que o levem a concluir que existe uma intrusão em andamento (AXELSSON, 2000).

1.7.4 Outros Tipos de Classificação

Os modelos utilizados para a construção de sistemas detectores de intrusão estão cada vez mais complexos. Por essa razão, muitos tipos de classificação podem ser deduzidos pela observação de suas estruturas e formas de operação. Para Axelsson (2000), os IDS podem ainda ser classificados de acordo com os seguintes princípios:

- *Tempo para a detecção*: nesta abordagem, dois grupos principais são criados: aqueles que detectam intrusões quase em tempo real e aqueles que levam muito tempo para constatar essas intrusões, impossibilitando uma possível reação rápida por parte da equipe de segurança do ambiente monitorado.

- *Granularidade do processamento de dados*: esta categoria diferencia os sistemas que operam continuamente a verificação dos dados e aqueles que analisam as informações em intervalos regulares de tempo.
- *Respostas às detecções de intrusões*: por essa abordagem são divididos em passivos e ativos. Os IDS passivos em caso de uma intrusão notificam os administradores, mas não realizam nenhuma outra operação que sirva para evitar que maiores danos sejam causados pelo ataque. Os sistemas ativos, por sua vez, na ocorrência de uma invasão, além de alertar seus administradores, podem alterar regras de *firewall* e dados de roteamento para que o ataque possa ser barrado ou amenizado.
- *Local do processamento de dados:* as informações auditadas podem ser processadas em um computador central ou em diversos computadores distribuídos ao longo da rede.
- *Local da captura dos dados:* os dados coletados têm a possibilidade de provirem tanto de um computador central, responsável por essa captura, ou de outros computadores ligados à rede monitorada.
- *Segurança:* a habilidade em poder barrar intrusões destinadas ao próprio sistema detector de intrusão.
- *Grau de interoperabilidade:* informa quando o IDS é capaz de operar em conjunto com outros mecanismos de segurança.

1.8 Arquitetura

A arquitetura de um IDS está diretamente ligada à forma como seus componentes interagem. Dois fatores principais influenciam diretamente na arquitetura: a localização e a fonte dos dados.

A localização dos membros da arquitetura define se é um sistema centralizado, hierárquico ou distribuído. No primeiro caso, tem-se todos os componentes do IDS localizados no mesmo ponto, realizando as funções de coleta de dados, processamento, configuração e gerência do mecanismo. Em uma arquitetura hierárquica, os componentes são parcialmente distribuídos, mas com fortes relações de hierarquia entre eles. Geralmente, existe a figura de um elemento principal, que, por sua vez, pode também estar submetido a outros membros. Já em uma arquitetura distribuída, os componentes estão espalhados pelo ambiente, com uma relação mínima de hierarquia entre eles.

A fonte dos dados pode ser centralizada, ou distribuída e as análises embasadas no comportamento de computadores ou no comportamento da rede. As arquiteturas distribuídas, embora tenham uma implementação um pouco mais complexa, ganham em tolerância à falhas, em escalonabilidade - já que é possível a instalação de novos nodos sem precisar, necessariamente, alterar todo o IDS - e em custos, pois, como não há uma grande quantidade de dados em cada nodo, não existe a necessidade da aquisição de computadores com alto poder de processamento.

Outra forma de modelagem dos sistemas detectores de intrusão é embasar o seu funcionamento em sensores espalhados pela rede. Os sensores são sistemas responsáveis por propiciar a coleta dos dados necessários à detecção feita pelo IDS (VIRTI, 2005). Nessa arquitetura distribuída, o processamento das informações obtidas

pode tanto ser feito localmente quanto ser entregue a um computador central que, pela criação de uma matriz de eventos, faz a análise conjunta os dados coletados, de forma a monitorar o panorama da rede.

A atuação dos sensores tem a possibilidade de ser embasada nas informações obtidas pelo protocolo SNMP (CASE, 1990) (BALASUBRAMANIYAN, 1998) (CABREIRA, 2003), obtendo dessa forma, maior portabilidade na interação com outras ferramentas de segurança e gerencia de redes. As informações transmitidas ao IDS podem ser agrupadas e apresentadas com os objetos gerenciáveis das diversas MIB já padronizadas, como, por exemplo, as MIB RMON1 (WALDBUSSER, 1995) e RMON2 (WALDBUSSER, 1997) (MENEGETTI, 2002) (GASPARY, 2005). O capítulo 5 apresenta os detalhes desse protocolo e as suas aplicações no contexto da detecção de intrusão.

Assim como analisado no capítulo 1, há também a possibilidade da utilização de *honeypots* como sensores para os sistemas de detecção de intrusão. Pela premissa de que todos os pacotes destinados aos *honeypots* são maliciosos, a utilização desses mecanismos como indicadores de intrusão, contribui para a eficácia dos IDS (KAIBIRI, 2005). Algumas ferramentas para a criação de *honeypots* permitem iludir os atacantes pela concepção de redes virtuais, com diferentes computadores, roteadores e serviços (SINGARJU, 2004). O capítulo 4 traz maiores detalhes sobre a operação dos *honeypots*.

1.9 Aplicação da Inteligência Artificial nos IDS

A aplicação da inteligência artificial é bastante utilizada no propósito da detecção de intrusão. Pesquisadores têm gerado muitos modelos diferentes para criarem sistemas que permitam distinguir entre os comportamentos normais e aqueles indicativos de intrusão. Alguns estudos têm se preocupado com a aplicação de métodos baseados em regras (CORDON, 2001) (DIKERSON, 2000) (DIKERSON, 2001) (TILLAPART, 2002) (ASTITHAS, 2002) (SILVEIRA, 2004) (ZHANG, 2004-a) (IDRIS, 2005) (IDRIS, 2006). Há estudos em que métodos de mineração de dados (*data mining*) são empregados em conjunto com ferramentas embasadas em regras na tentativa de criar sistemas que melhor se adaptem às redes monitoradas (BRIDGES, 2000) (IDRIS, 2006). No entanto, a geração dessas regras exige que especialistas em segurança tenham de analisar o funcionamento do ambiente para poder criar regras que traduzam o comportamento esperado do IDS. Alguns estudos tentam cunhar essas regras automaticamente, seja pela utilização de métodos numéricos (LEE, 2004), seja pela elaboração de redes neurais (GOMEZ, 2002-a) (IDRIS, 2005) (CORCHADO, 2005).

Outros trabalhos têm proposto aplicações do conceito da lógica difusa para o problema da detecção de intrusão no intuito de, pela abordagem nebulosa, fazer o sistema detector de intrusão ser provido de uma maior capacidade de lidar com dados imprecisos provenientes de seus sensores (DIKERSON, 2000) (BRIDGES, 2000) (DIKERSON, 2001) (MANIC, 2001) (GOMEZ, 2002-a) (GOMEZ, 2002-b) (TILLAPART, 2002) (ORFILA, 2003) (SHAH, 2003) (SILVEIRA, 2004) (LEE, 2004) (IDRIS, 2006). O capítulo 6 traz maiores análises sobre a aplicação da lógica difusa em IDS e faz exames mais detalhados sobre as abordagens propostas nesses diferentes trabalhos.

Alguns pesquisadores têm empregado a metodologia Bayasiana para resolver o problema da detecção de intrusão. A idéia principal por trás desse método é se utilizar de sua capacidade de detalhar as formas e os cálculos utilizados na tomada de determinada decisão. Esse recurso entrega ao administrador do IDS a possibilidade de

entender exatamente quais os dados que levaram o sistema a discernir entre os comportamentos aceitáveis e aqueles sintomáticos de invasões (KAIBIRI, 2005).

Há ainda os pesquisadores que tratam do problema da detecção de intrusão pela abordagem de sistemas multiagentes (BERNARDES, 2000) (BALASUBRAMANIYAN, 1998). Os autores, com a utilização dessa abordagem, tentam implementar IDS que sejam tolerante à falhas, portáveis e escaláveis.

As muitas abordagens que utilizam inteligência artificial para tratar o problema da detecção de intrusão demonstram os esforços da comunidade científica na busca de uma solução definitiva para tal problema. No entanto, alguns trabalhos pecam pela complexidade das estruturas envolvidas, como os apresentados por Dikerson (2000), Bernardes (2000), Bridges (2000) e Idris (2006); outros pela necessidade de apresentar ao IDS exemplos de intrusão, como em Gomes (2000-a) e Lee (2004); e, ainda, pelos excessos no consumo de processamento e recursos destinados a atividade da detecção de intrusão, como em Bernardes (2000). Muitos desses trabalhos precisam ser aprimorados, o que torna a detecção de intrusão um importante ramo de pesquisas da comunidade mundial envolvida com a segurança de ambientes.

HONEYPOTS

A comunidade de segurança necessita de constantes informações sobre as técnicas utilizadas para a criação de incidentes na Internet. Em muitas invasões apenas tornam-se conhecidas as danosas conseqüências desses ataques e não os pormenores da ação dos invasores, pois, geralmente, as equipes de administração de redes, seja por falta de conhecimento, seja pela urgência no restabelecimento dos sistemas comprometidos, apenas reconfiguram os computadores atacados, ignorando o aprendizado dos métodos utilizados para criar tal incidente de segurança (HOLZ, 2005-b). Os *honeypots* são mecanismos criados para suprir tal deficiência de informações sobre os detalhes das ações dos criminosos.

Em 1999, Lance Spitzner conectou à Internet um computador operando com um sistema operacional Red Hat 5.1 e, propositalmente, executando aplicativos com vulnerabilidades conhecidas. Spitzner surpreendeu-se, pois em menos de 15 minutos, seu host já havia sido atacado e comprometido. Um honeypot é "um recurso de rede cuja função é de ser atacado e comprometido" (SPITZNER, 2002). Os *honeypots* não fazem nenhum tipo de prevenção e fornecem informações adicionais de valor inestimável. Geralmente, possuem falhas de segurança, reais ou virtuais, colocadas de maneira proposital, a fim de que sejam invadidos e que os resultados destas invasões possam ser estudados. No episódio de Spitzner, o atacante acabou percebendo que estava sendo monitorado e apagou os logs do sistema. Tornou–se necessária a construção de ambientes que pudessem melhor gerenciar e monitorar as atividades dos criminosos. Surgiram as chamadas Honeynets: redes que têm em sua arquitetura, sub-redes de *honeypots*. Nessas redes, os administradores podiam criar ambientes mais seguros, tendo a possibilidade de guardarem em diferentes computadores os logs gerados (FRANCO, 2004).

O aumento no número de redes que empregam *honeypots* em suas arquiteturas criou a necessidade da troca das informações relativas aos ataques e às descobertas de novos recursos a serem empregados no monitoramento da ação dos invasores. Tendo essa finalidade, atualmente, existem várias instituições no mundo integradas na chamada *Honeynet Research Alliance* (HONEYNET, 2007). Sendo um dos integrantes dessa aliança, no Brasil, o Consórcio Brasileiro de Honeypots (CONSORCIO, 2007), que conta com a cooperação de vinte e três instituições espalhadas em todo pelo país (figura 4.1), tem por objetivo aumentar a capacidade de detecção de incidentes e avaliar as tendências de ataques no espaço da Internet brasileira. O estudo dos dados gerados nessas alianças e o trabalho em conjunto com os Grupos de Resposta a Incidentes de Segurança tem ajudado a agilizar as respostas da comunidade de segurança mundial às ameaças existentes.

A instalação e configuração de um honeypot, dependendo da escolha do tipo de mecanismo a ser criado, pode ser uma tarefa complexa. Cada instituição deve analisar

as alternativas existentes e os ganhos pretendidos com a adoção de *honeypots* como mecanismos de segurança. Dependendo das características desses mecanismos, é necessário muito empenho da equipe de segurança na tentativa barrar as possíveis ações de atacantes que conseguirem comprometer um *honeypot*.

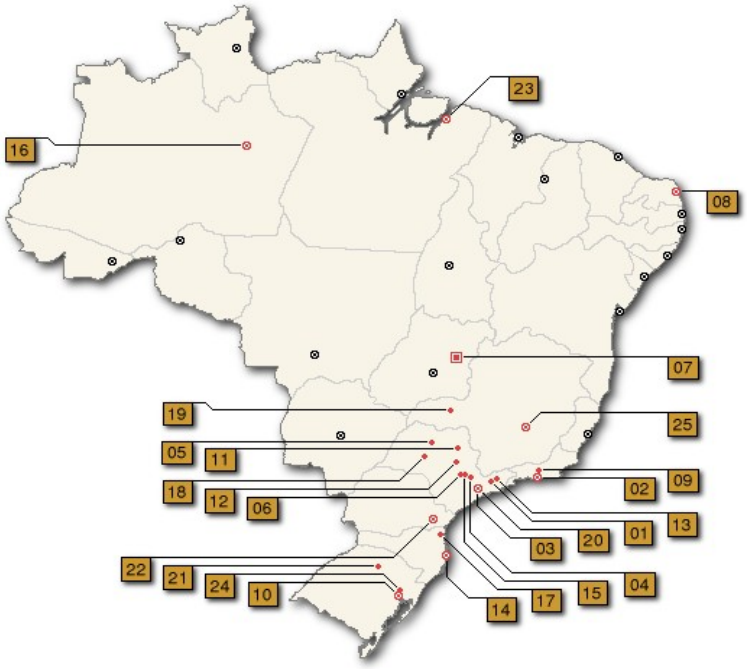

Figura 4.1: Consórcio Brasileiro de Honeypots

1.10 Classificação

Os *honeypots* podem ser classificados em alta e baixa interatividade. Dependendo do grau de segurança necessário para poder barrar uma possível intrusão bem sucedida, para a adoção de um desses mecanismos, deve-se avaliar os riscos e benefícios envolvidos.

1.10.1 Honeypot de alta interatividade

Em um honeypot de alta interatividade o atacante, ao fazer uma invasão, tem acesso ao sistema operacional real do computador invadido. Geralmente nesses sistemas há alguns serviços vulneráveis que servem de iscas para atrair e iludir os invasores de forma a não perceberem que seu alvo é um computador configurado para sofrer tais ataques. Esses *honeypots* têm a capacidade de armazenarem históricos de todas as ações do atacante e, dessa forma, proporcionarem dados para o estudo das principais técnicas utilizadas pelos criminosos.

Os *honeypots* de alta interatividade possibilitam um estudo aprofundado das ações de um atacante, pois, tendo acesso ao sistema operacional real, os criminosos podem, dependendo do nível de segurança a que esse sistema está submetido, instalar ferramentas e realizar ações que são estudadas pelas equipes de segurança.

No entanto, configurar um honeypot de alta interatividade necessita maiores conhecimentos e maiores recursos para a criação de limitações às ações dos atacantes. Atualmente já há alguns sistemas operacionais construídos especificamente para

atuarem como *honeypots* de alta intereratividade (VIECCO, 2007). Mesmo assim, a equipe de segurança precisa saber exatamente onde são os pontos fracos do sistema a fim de poder reagir rapidamente diante de um comprometimento que possa vir a causar maiores danos aos sistemas de produção. Se ocorrer alguma falha nas restrições de acesso impostas ao honeypot, o atacante pode, por exemplo, invadir outros sistemas através do honeypot, ou ainda, ter acesso a importantes serviços internos à rede onde o honeypot foi instalado.

Os *honeypots* de alta interatividade são recomendados para equipes de segurança que queiram ter conhecimentos mais aprimorados sobre os tipos de ataques que estão sofrendo. Para fazer uma análise forense, por exemplo, são os que mais se adequam. Faz-se necessário que a equipe tenha tempo disponível para monitorar as ações dos invasores. Caso não se disponha de tantos recursos humanos e materiais, a opção por *honeypots* de baixa interatividade pode ser uma idéia mais apropriada (FRANCO, 2004).

1.10.2 Honeypot de baixa interatividade

Um honeypot de baixa interatividade realiza a emulação de serviços de forma que o atacante não tem acesso ao sistema operacional, mas a emulações onde tenta-se recriar os comportamentos e respostas de um ambiente real.

Os *honeypots* de baixa interatividade não fornecem tantas informações quanto os de alta (KUWATLY, 2004). No entanto, exigem poucos recursos materiais e humanos por parte da equipe que o gerencia e, ao menos teoricamente, há uma maior segurança associada a esse tipo de mecanismo, pois as respostas fornecidas são apenas simulações criadas para ludibriar os atacantes.

Geralmente, esses honeypots permitem que os próprios administradores criem softwares que respondam a determinados serviços. Esse recurso é importante para a emulação de serviços onde, sabidamente, uma dada rede esteja sofrendo tentativas de ataque ou onde existam servidores em produção que necessitem ser protegidos dessas tentativas de ataque.

1.10.2.1Honeyd

O Honeyd é um dos softwares para a criação de *honeypots* de baixa iteratividade mais utilizado no mundo (PROVOS, 2005). Ele possui a capacidade de emular o funcionamento de diversos computadores, podendo, em cada um, simular a operação de um sistema operacional diferente. É possível, ainda, criar uma complexa rede com vários computadores, roteadores, serviços e sistemas operacionais, podendo emular, inclusive, conexões defeituosas que acarretem perda de pacotes entre os nodos dessa rede (figura 4.2). Uma das vantagens da utilização de um software com essas características é a capacidade de espelhar sistemas em produção como forma de protejer os sistemas reais e críticos.

Supondo, por exemplo, que em uma determinada rede privada queira-se aumentar a segurança de um servidor de arquivos do MS Windows. É possível instalar um computador onde, com o auxílio da ferramenta Honeyd, esteja configurada a simulação da existência de dez servidores de arquivos desse mesmo sistema operacional. Assim, para o atacante remoto, existirão na rede onze computadores responsáveis pelos compartilhamentos de arquivos. Se o criminoso disparar um ataque, haverá grandes chances de atingir um honeypot (RAZZARI, 2003) e como todos os

pacotes destinados a esses mecanismos são, por princípio, maliciosos (KREIBICH, 2004), poder-se-á alertar os administradores de segurança.

O mesmo pode ser feito para uma rede que necessite ser protegida, devido à alta importância das operações nela realizadas. É possível criar no Honeyd um ou mais espelhos dessa rede como forma de atrair os atacantes para as redes emuladas.

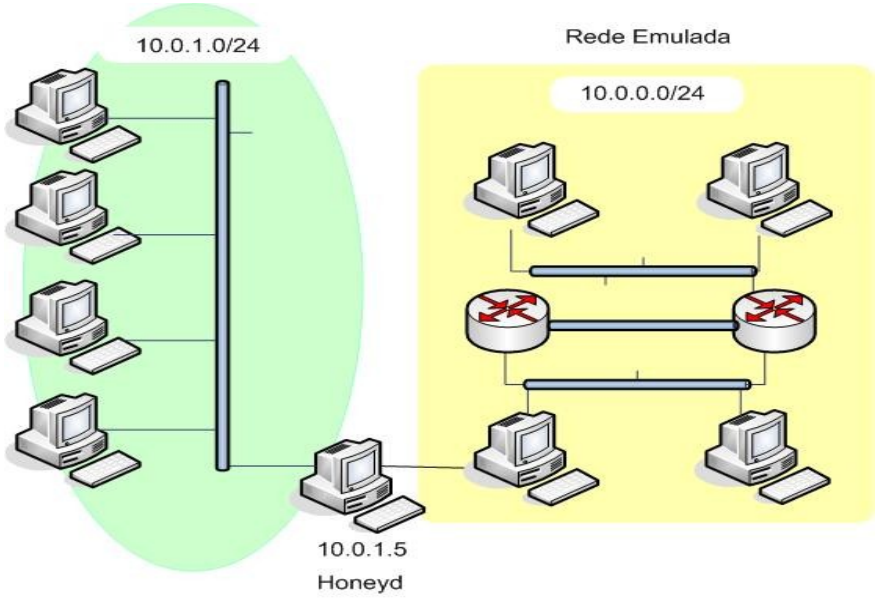

Figura 4.2: Simulação de uma rede utilizando Honeyd

A figura 4.2 apresenta a criação de uma rede simulada de computadores onde só existem 5 computadores reais: 4 estações de trabalho e um computador operando com Honeyd. Apesar de possuir o endereço 10.0.1.5, neste caso o Honeyd foi configurado para de responder a toda conexão direcionada a qualquer IP existente no bloco 10.0.0.0/24. Dos 255 endereços disponíveis no bloco, apenas 6 foram utilizados: 4 para simular *desktops* e dois para emular roteadores.

Para os objetivos desse trabalho, a principal característica desejada foi a possibilidade do honeyd registrar qualquer tentativa de acesso a um dos IPs simulados. Dessa forma, integrando o honeypot ao sistema detector de intrusão deste trabalho, pode-se detectar ações anômalas provindas dos próprios computadores pertencentes à rede monitorada. Assim, o honeypot comporta-se como mais um sensor do IDS.

1.11O honeypot como mecanismo de segurança

O honeypot deve ser um complemento aos mecanismos de segurança já existentes. Os *firewalls* e os sistemas detectores de intrusão têm seus escopos de operação limitados às suas características de ação. Um *firewall* pouco pode fazer com um ataque originado de um programa baixado ou recebido por e-mail, por exemplo. Os IDS, geralmente, não possuem a capacidade de detectar tentativas de acessos a endereços não utilizados ou a serviços não disponibilizados.

Schneier (2002) decompõe a segurança em três domínios distintos: prevenção, detecção e reação. Os *honeypots* são úteis nas três categorias.

Um honeypot não impede um atacante de entrar na rede, mas como a totalidade do tráfego originado pelo intruso fica registrada e pode ser analisada, é possível obter informações que permitem, em outra ocasião, bloquear o mesmo ataque. Além do mais,

é sempre preferível que os invasores invistam seus esforços em recursos secundários (emulados) em vez de tentarem penetrar em servidores de produção.

No que se refere à detecção, os ganhos são mais consideráveis. Se as ferramentas complementares, como os IDS de rede, forem expostas a grandes fluxos de tráfego, terão dificuldade em processá-los. Separar o tráfego útil do tráfego malicioso é, por vezes, muito difícil. Uma das estratégias dos atacantes consiste em ocupar um IDS, de modo a fazê-lo gerar um grande número de alarmes. Os falsos positivos e a filtragem dos dados úteis continuam a ser quesitos onde os IDS ainda necessitam fazer progressos. Um honeypot não tem este problema, pois todo o tráfego destinado aos computadores emulados é, por definição, suspeito. Embora isto não signifique que os falsos positivos sejam impossíveis, a possibilidade de acontecer é bem menor do que com um IDS.

A detecção de pouco vale quando não se tem uma capacidade de resposta adequada. Através da análise dos logs gerados no honeypot, a equipe de segurança pode reagir tomando medidas técnicas para a proteção contra a vulnerabilidade explorada ou para a identificação e a punição legal dos atacantes.

No que se refere a adoção dos *honeypots* como mecanismos efetivos para a melhora da segurança das redes onde são instalados muito se têm pesquisado. Kreibich (2004) propôs utilizar *honeypots* para a criação automática de assinaturas de ataque que, posteriormente, são encaminhadas para sistemas detectores de intrusão. O autor utilizou a ferramenta Honeyd para encontrar similaridades nos pacotes recebidos.

Razzari (2004) gera um sistema detector de intrusão embasado na criação de cinco tipos diferentes de agentes em Java. A portabilidade oferecida pela linguagem permite que o IDS possa, dinamicamente, emular serviços em diferentes sistemas operacionais em detrimento da sua performance. Esses agentes se comunicam na tentativa de armazenarem o histórico das ações dos atacantes sobre os protocolos SNMP (porta 25/TCP), FTP (portas 20 e 21/TCP) e Telnet (porta 23/TCP) na tentativa de iludirem os criminosos e impedirem que os mesmos atinjam sistemas em produção.

A criação de honeypots dinâmicos é proposta por Kuwatly (2004). Esses mecanismos analisam a rede onde estão inseridos e, continuamente, criam *honeypots* nos endereços IP vagos, espelhando os serviços disponibilizados pela rede supervisionada. O artigo propõe a criação de um sistema que automaticamente se adapte ao ambiente onde for instalado.

O envio de todo o tráfego malicioso para um honeypot é proposto por Franco (2003) e Carter (2004). Operando sobre dispositivos de rede, os artigos propõem o direcionamento a um honeypot do tráfego que, enviado à rede monitorada, não é destinado a endereços em uso real.

Os estudos proporcionados pelos dados coletados por *honeypots* podem ser aplicados a diversas áreas da segurança de sistemas. Alguns trabalhos utilizam *honeypots* até para analisarem a operação de *botnets* (BOTNETS, 2005) (HOLZ, 2005). No entanto, a ilusão causada por um honeypot dificilmente engana o atacante mais experiente. No caso de um ataque direcionado a um honeypot de baixa interatividade, a detecção pode ser feita pelo simples envio de um comando não suportado pelo software de emulação. Já nos mecanismos de alta interatividade, tanto a tentativa de iludir o atacante quanto as técnicas usadas para a detecção do *honeypots* são mais complexas. No trabalho feito por Holz (2005-b) há alguns métodos para a detecção de um honeypot

de alta e de baixa interatividade. No entanto, como essas técnicas necessitam que o atacante se conecte ao sistema, sua ação criminosa é facilmente verificada.

1.12 Honeypot como Sensor para o IDS

Os escopos de ação dos IDS e dos *honeypots* são diferentes e complementares. A integração entre esses dois mecanismos tem a capacidade de melhorar as decisões tomadas pelos IDS, pois, pela menor possibilidade da geração de falsos positivos, os dados provenientes dos *honeypots* podem auxiliar na precisão das decisões emanadas pelos sistemas detectores de intrusão.

Os sensores dos sistemas detectores de intrusão são mecanismos que monitoram as redes onde o IDS é instalado, auxiliando na verificação do panorama da rede monitorada. No entanto, no caso dos *honeypots*, o simples fato de um dado endereço pertencente à própria rede supervisionada tentar acessar o honeypot, confere ao computador fonte de tais pacotes uma grande possibilidade de estar sob a ação de um atacante ou software malicioso. Dessa forma as informações obtidas dos *honeypots* têm grande peso na geração de alarmes ou reações por parte do IDS.

Utilizando a ferramenta para *honeypots* de baixa interatividade, Honeyd, foi visto que é possível espelhar sistemas em produção e até redes inteiras (KUWATLY, 2004). Este trabalho utiliza essa ferramenta como um tipo de sensor do IDS que, operando em conjunto com os agentes SNMP, tem o poder de auxiliar na eficácia do sistema detector de intrusão proposto.

O PROTOCOLO SNMP

Criado em 1988 para satisfazer a crescente demanda por uma padronização na gerência de redes IP, o termo SNMP (*Simple Network Management Protocol*) é hoje utilizado para descrever uma série de especificações incluindo o próprio protocolo, a definição da estrutura de dados e outros conceitos associados (STTALINGS, 1996).

O objetivo geral da gerência de redes é monitorar e controlar os elementos físicos e lógicos de uma rede, assegurando um dado nível de qualidade de serviço (LOPES, 2003). A crescente gama de utilizações das redes de computadores fez seu gerenciamento tornar-se demasiado complexo. Compostas por múltiplos componentes, indo desde equipamentos periféricos, como impressoras e scanners, passando por dispositivos de redes, como roteadores, comutadores e repetidores e culminando em computadores pessoais, as complexas redes exigem protocolos e ferramentas que automatizem o seu gerenciamento. Os administradores, no entanto, podem ser auxiliados por um sistema de gerência, definido como uma coleção de ferramentas integradas para o monitoramento e o controle da rede. Esse sistema fornece dados sobre a rede monitorada e pode apresentar um conjunto poderoso e amigável de comandos que são usados para executar quase todas as tarefas de gerência.

A arquitetura geral de um sistema de gerenciamento de redes pode ser separada em quatro elementos básicos: sistema gerenciado, estações de gerência, protocolos de gerência e informações de gerência.

- Os *sistemas gerenciados* são os elementos da rede para os quais o sistema de gerenciamento destina suas atividades.
- As *estações de gerência* são responsáveis por fazerem a leitura e a escrita de informações chaves à atividade de gerenciamento.
- O *protocolo de gerência* propicia a comunicação entre os elementos gerenciados e os gerentes de uma rede.
- As *informações de gerência* definem os dados que podem ser referenciados em operações do protocolo de gerência.

A idéia chave do SNMP consiste em um conjunto de operações feitas sobre o protocolo UDP onde os resultados obtidos dão ao administrador a possibilidade de monitorar o estado dos elementos da rede que possuem suporte ao SNMP. Enquanto o seu predecessor o SGMP (*Simple Gateway Management Protocol*) foi desenvolvido para somente gerenciar roteadores, o SNMP pode fazê-lo em sistemas Unix, Windows, impressoras, comutadores, roteadores, entre outros (MAURO, 2001).

1.13 Versões do SNMP

Ao longo da historia do SNMP iniciada em 1988, o protocolo sofreu algumas mudanças. Abaixo, são apresentadas as principais características das três versões até agora existentes.

- *SNMP versão 1:* é a versão inicial do protocolo (RFC 1157) (CASE, 1990). A segurança do SNMPv1 é embasada em "comunidades", uma forma de senhas em texto claro que permite o acesso de leitura e escrita sobre as informações de gerenciamento. Existem tipicamente três tipos de comunidades SNMPv1: somente-leitura, leitura-escrita e trap (armadilha). Embora este padrão seja apenas histórico ele ainda é suportado por uma gama de dispositivos e software de gerenciamento.
- *SNMP versão 2*: Herda do SNMPv1 a utilização das comunidades como senhas para autenticação. As melhoras no SNMPv2 estão ligadas a criação de outros tipos de objetos gerenciados e de outras formas de comunicação entre os elementos do sistema de gerência. Esta versão foi definida pelas seguintes RFCs: RFC 3416, RFC 3417 e RFC 3418.
- *SNMP versão 3:* a última versão deste protocolo traz grande contribuição para a segurança. Foi adicionado o suporte a autenticação forte e a comunicação cifrada entre os dispositivos gerenciados. Em 2002 ocorreu sua publicação como padrão. As seguintes RFCs definem este padrão: RFC 3410, RFC 3411, RFC 3412, RFC 3413, RFC 3414, RFC 3415, RFC 3416, RFC 3417, RFC 3418 e RFC 2576.

Mesmo que atualmente muitas implementações do SNMP já estejam habilitadas para utilizar a versão 3 do protocolo, é comum encontrar redes em produção em que ainda é utilizada apenas a versão 1. Essa falha na segurança pode trazer aos atacantes importantes informações a respeitos dos sistemas gerenciados.

1.14 Agentes e Gerentes

No SNMP existem dois tipos de entidades: os agentes e os gerentes. O gerente é um computador que executa algum tipo de software que realiza tarefas de gerenciamento de rede, seja pela busca de informações (*poll*ing), seja pelo recebimento de *traps* (armadilhas) dos agentes na rede. No contexto do gerenciamento de redes fazer *polling* significa buscar, continuamente, em um dado agente alguma informação necessária. A *trap* é o meio utilizado pelo agente para se comunicar de forma assíncrona com o gerente (figura 5.1).

A segunda entidade, os agentes, são softwares que operam nos sistemas gerenciados, tendo a finalidade de fornecer todas as informações necessárias para propiciar o gerenciamento do sistema em que operam (ROSE, 1991). Eles podem ser programas separados (*daemons* nos sistemas Unix) ou podem já estarem incorporados no sistema operacional (como em alguns modelos de roteadores).

Hoje, muitos dispositivos de rede, periféricos e sistemas operacionais já têm suporte a algum tipo de agente SNMP. Os fabricantes implementam esses agentes como forma de tornar mais fácil o gerenciamento de seus produtos. Em um roteador, por exemplo, utilizando esse protocolo, pode-se observar o estado de cada uma das interfaces e tomar as medidas necessárias se alguma delas não estiver operando da forma correta. Nestes

casos, alguns dispositivos de rede podem ser configurados para enviar uma *trap* informando que o sistema voltou a operar normalmente.

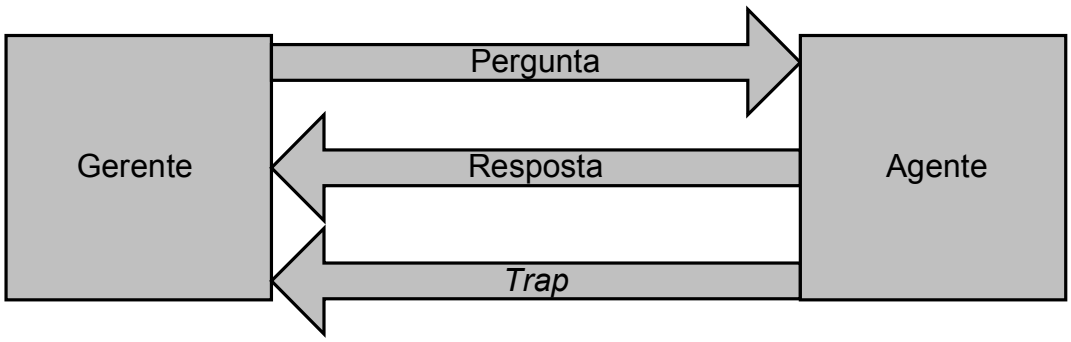

Figura 5.1: Modelo de comunicação entre agentes e gerentes SNMP.

1.15 Operações em SNMP

O SNMP define um conjunto de operações destinadas a efetivar e organizar as comunicações entre os elementos desse protocolo. Abaixo são apresentadas algumas das operações possíveis.

- *Get*: é iniciada pelos gerentes que, para obterem alguma informação encontrada em uma determinada MIB (seção 5.4), enviam uma solicitação para o agente.
- *Get-response*: é o modelo de respostas para a operação de *get*, utilizado pelos agentes para a entrega dos valores constantes nos objetos das MIBs aos gerentes que os solicitaram.
- *Get-next:* é a operação que permite aos gerentes obterem a totalidade dos objetos de um determinado grupo das MIBs as quais os agentes tem suporte. Uma vez que os objetos buscados são representados por uma seqüência de números inteiros, a operação *get-next* faz o agente devolver o valor do objeto imediatamente posterior aquele anteriormente informado.
- *Get-bulk* (SNMPv2 e SNMPv3): permite que, com uma única comunicação, uma aplicação de gerenciamento obtenha uma larga seção de uma tabela MIB. Embora a operação de *get* também permita que sejam buscados mais de um objeto em uma mesma operação, se o agente não conseguir obter todos os objetos procurados, irá responder erro, com nenhum dado retornado. A operação de *get-bulk,* por sua vez, faz o agente devolver tantos dados quantos forem encontrados.
- *Set:* é usada para alterar o valor de um objeto gerenciado. Os objetos que são definidos como leitura-escrita, usando essa operação, permitem a mudança de seus valores. É possível que um gerente altere mais de um objeto ao mesmo tempo.
- *Trap:* é a forma utilizada pelos agentes para informarem aos seus gerentes, de maneira assíncrona, a ocorrência de algum evento previamente configurado. Como esse tipo de mensagem não tem confirmação, o agente não tem meios para saber se a informação foi recebida. Embora seja menos confiável, em uma rede em produção, sua implantação é extremamente útil (MAURO, 2001).

As operações realizadas sobre o protocolo SNMP seguem algumas características influenciadas pelo protocolo de transporte utilizado: o UDP. O emprego desse protocolo

sem confirmação e sem entrega garantida, deixa a cargo da aplicação o tratamento das possíveis perdas de pacotes na comunicação. Todas as operações de *get* e a de *set* possuem confirmações que são as próprias respostas dos pedidos. No entanto, ao utilizar-se o *trap* não há nenhuma garantia de que o pacote enviado foi corretamente entregue ao gerente.

Para o monitoramento automático, o administrador deve levar em consideração a estrutura de suas redes para evitar erros provenientes de perda de comunicação entre agentes e gerentes. Se, por exemplo, houver um roteador sobrecarregado, interligando a estação de gerência e algum agente monitorado, essa poderá, por problemas na infra-estrutura, concluir que o dado agente está inoperante.

1.16 As MIB

No SNMP as bases de conhecimento são os dados contendo as informações sobre os elementos a serem supervisionados. Esses dados são referenciados pelas bases de informações de gerenciamento – MIB (*management information base*). Cada recurso a ser monitorado é representado como um objeto onde a MIB é estruturada como um conjunto desses objetos. As MIBs possuem uma estrutura em árvore. Cada sistema mantém uma MIB que reflete o status dos dados dos recursos gerenciados nesse sistema. Um gerente pode monitorar os recursos de qualquer sistema pela leitura dos valores dos objetos presentes na MIB e pode controlar os recursos desse sistema pela modificação desses valores.

Os objetos usados para representar um particular recurso precisam ser os mesmos em cada sistema monitorado. Considerando as informações armazenadas relativas ao protocolo TCP em um dado sistema, por exemplo, o número total de conexões abertas em um período de tempo consiste nas conexões passivas e ativas. A MIB do sistema poderia armazenar qualquer dois dos três valores relevantes (número de conexões passivas, ativas e número total de conexões) de forma que o terceiro pudesse ser facilmente derivado. No entanto, se diferentes sistemas selecionassem distintos pares de objetos para armazenar esses dados, seria difícil escrever um protocolo que acessasse as informações necessárias. Dessa forma, para o TCP/IP a MIB explicitamente determina que sejam monitorados os números de conexões ativas e passivas.

1.16.1 Estrutura das MIBs

Todos os objetos gerenciáveis pelo SNMP são arranjados em uma estrutura hierárquica de árvore. Os objetos folha das árvores são os próprios objetos gerenciáveis, que representam algum tipo de recurso ou atividade que possa ser gerenciada. A estrutura de árvore define agrupamentos de objetos em conjuntos lógicos relacionados.

Associado a cada objeto pertencente a uma MIB existe um tipo de identificador "*object identifier*" ASN.1 (*Abstract Syntax Notation One*). Este identificador, além de nomear os objetos, permite localizá-los na estrutura da árvore de uma determinada MIB. A identificação do objeto da MIB RMON "*etherStatsTable*", por exemplo, é dada pela hierarquia apresentada na figura 5.2. A tabela 5.1 demonstra a numeração utilizada para endereçar tal objeto.

Tabela 5.1: Identificação do objeto *etherStatsTables* da MIB RMON

iso	org	Dod	internet	Mgmt	mib-2	rmon	statistcs	etherStatsTable

1	3	6	1	2	1	16	1	1

Assim, o objeto "*etherStatsTable"* é normalmente referenciado como 1.3.6.1.2.1.16.1.1. Nas comunicações entre agentes e gerentes, as solicitações de informações contidas nesse objeto serão feitas utilizando-se a essa notação.

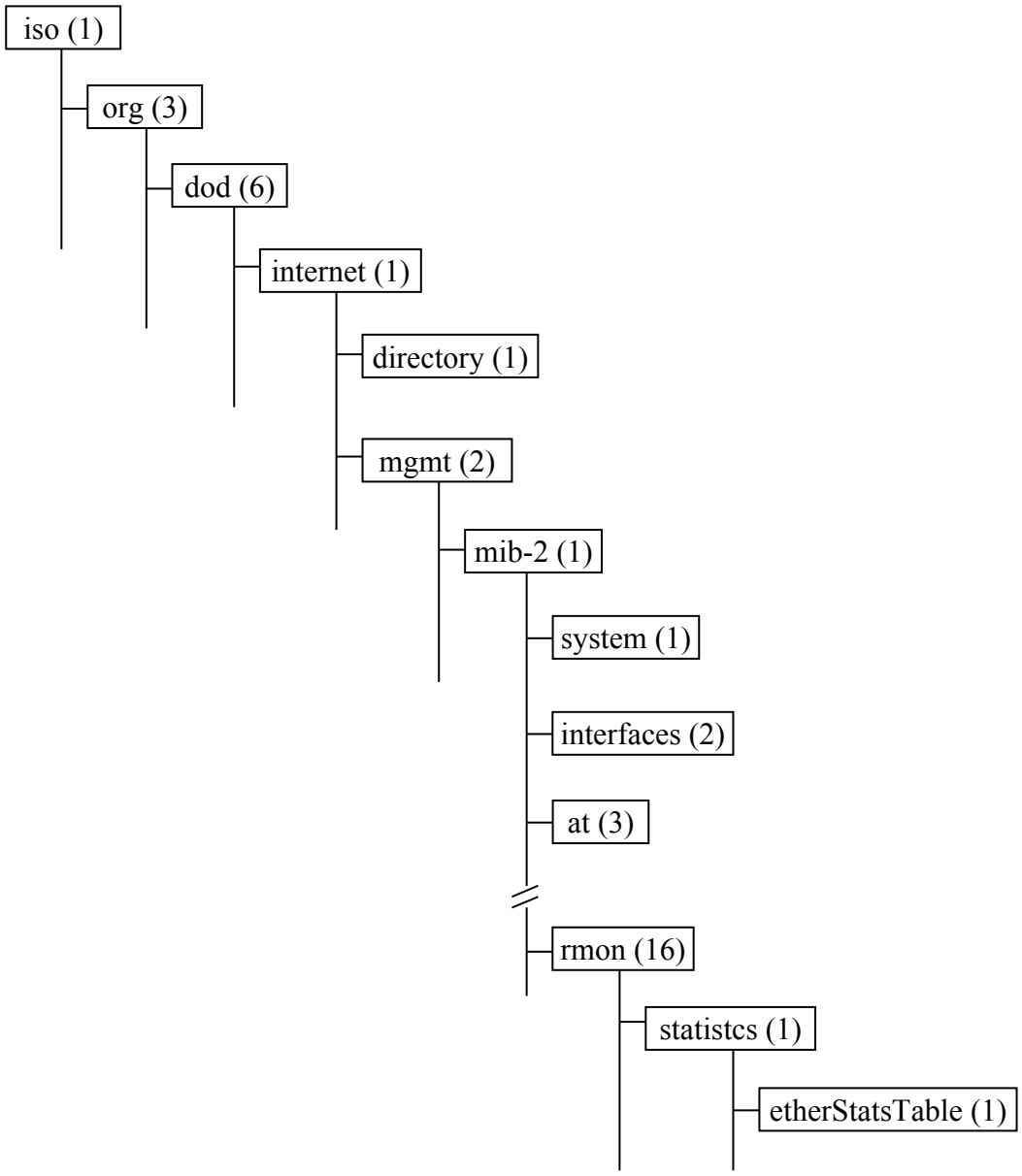

Figura 5.2: Estrutura da árvore na qual o objeto *etherStatsTables* da MIB RMON está inserido

De acordo com a RFC1155, que dispõe sobre o SMI (*Structure of Management Information*), e a RFC2579 que cria uma nova versão para o SMI, em um dado objeto de uma MIB é possível armazenar vários tipos de dados. Abaixo são apresentados alguns desses tipos:

- *Integer:* número inteiro de 32 bits geralmente utilizado para representar enumerações. Pela SMIv2 é conhecido como *integer32*.
- *Ipaddress:* é um valor de 32 bits que representa um endereço do protocolo IP versão 4.

- *Networkaddress:* o mesmo que o *Ipaddress,* mas pode armazenar diferentes tipos de endereços de rede.
- *Counter:* é um inteiro, não negativo, de 32 bits que permite apenas o acréscimo de seu valor. Quando esse contador atinge o máximo, a numeração começa novamente do zero. Este tipo de conteúdo é utilizado para verificar o número de octetos enviados ou recebidos em uma interface de rede de um determinado sistema. O contador é monoliticamente incrementado e o seu valor, em uma operação normal não decresce. No entanto, quando um agente é reiniciado todos os contadores podem ter sido zerados. As operações sobre esses contadores normalmente se dão pela diferença entre os dados coletados no instante e aqueles armazenados no *pooling* imediatamente anterior. Na SMIv2 é apresentado pelo nome *counter32.*
- *Counter64:* presente apenas na SMIv2, tem a mesma forma de funcionamento do *counter32,* porém opera um contador de 64 bits.
- *Gauge:* é um inteiro, não negativo, de 32 bits em que pode ser acrescido ou decrescido algum valor. Se o valor máximo for atingido, o *gauge* permanece travado nesse valor até ser reajustado. Recebe o nome de *gauge32* na SMIv2.
- *Timeticks:* é um inteiro não negativo que conta o tempo em centésimos de segundos desde determinada época. Essa época é dada na definição do objeto que utilizar esse tipo de valor.
- *Opaque:* dados que representam uma codificação arbitrária. Este tipo de dado pode ser utilizado como um mecanismo de escape às restrições impostas pelo SMI (ROSE, 1991). Os valores armazenados com esse tipo de codificação são interpretados como um vetor de octetos.
- *Octet string:* é um atributo composto por zero ou mais octetos (bytes) geralmente usado para representar texto e endereços físicos (MAC).

A identificação dos tipos de dados armazenados nos objetos a serem gerenciados é importante para a correta utilização dos dados provenientes da base de informações. As peculiaridades de cada objeto devem ser verificadas no intuito da construção de ferramentas que possam monitorar esses valores constantes nas MIBs. No caso de objetos definidos como *counters*, por exemplo, é indispensável verificar como o sistema irá se comportar quando este contador atingir o seu limite e for reiniciado.

1.16.2 O RMON1

Uma das mais importantes adições feitas ao conjunto de padrões do SNMP foi a especificação do monitoramento remoto (*remote monitoring – RMON*). O RMON foi uma grande evolução para o gerenciamento de redes (STALLINGS, 1996). Ele define MIBs de gerenciamento remoto, que estendem a MIB-II e proporcionam ao gerente de redes informações vitais da rede monitorada. A vantagem do RMON é que mesmo sendo apenas uma especificação de MIB, sem alterar no protocolo SNMP, representa significantes extensões das funcionalidades do SNMP.

Os esforços para a criação da RMON1 foram iniciados em 1990 com a criação do grupo de trabalho da RMON1 no IETF (*Internet Task Force*). Em 1991 foi publicada a RFC 1271 enfocada somente nas redes ethernet. A RFC 1513 veio em 1993 para dar

suporte também às redes Token Ring. Finalmente, em dezembro de 1994, foi criada a RFC 1757 que define a MIB RMON1.

Os dispositivos de monitoramento RMON, também chamados de *probes*, são mecanismos que tem a funcionalidade de gerenciar as redes às quais estão conectados, muitas vezes entregando grande parte de seu processamento nessa tarefa. Uma organização pode utilizar muitos desses dispositivos, um por segmento de rede, para gerenciar a sua rede privada. Além do mais, estes dispositivos podem ser utilizados para que provedores de serviços de gerenciamento de redes possam realizar os seus trabalhos mesmo remotamente conectados (WALDBUSSER, 1995).

A adoção da tecnologia RMON em uma rede pode entregar aos seus administradores as seguintes informações (3COM, 2006):

- Estatísticas correntes e históricas de um segmento de redes, de um computador específico ou das comunicações entre computadores.
- Um versátil sistema de alarmes e mecanismos embasados em eventos para criar gatilhos para o envio de avisos aos administradores de redes.
- Um flexível sistema de filtros que captura pacotes que podem ser utilizados para criarem análises detalhadas das comunicações presentes no segmento monitorado.

Enquanto grande parte dos objetos presentes no RMON1 são criados para o gerenciamento de qualquer tipo de rede, existem alguns específicos para o gerenciamento de redes Ethernet. Estes objetos são os seguintes: *etherStatsTable*, *etherHistoryTable*, e alguns atributos de *filterPktStatus* e *capturBufferPacketStatus*. A figura 5.3 apresenta todos os grupos do RMON1. Abaixo segue um resumo das funcionalidades de cada um desses grupos.

- *Statistcs* (1.3.6.1.2.1.16.1): contém estatísticas sobre todas as interfaces de rede monitoradas pelo *probe* RMON1.
- *History* (1.3.6.1.2.1.16.2): registra amostras estatísticas periódicas de redes ethernet e as armazena para consultas posteriores.
- *Alarm* (1.3.6.1.2.1.16.3): permite a configuração dos intervalos de *pooling* e a delimitação dos intervalos de operação para quaisquer objetos registrados pelo *probe.*
- *Hosts* (1.3.6.1.2.1.16.4): registra as estatísticas de tráfego para qualquer computador presente no mesmo segmento de rede.
- *HostTopN* (1.3.6.1.2.1.16.5): contém estatísticas utilizadas para a geração de relatórios contendo uma lista de computadores ordenada por algum parâmetro presente no grupo *Hosts.*
- *Matrix* (1.3.6.1.2.1.16.6): armazena estatísticas sobre erros e informações de utilizações para conjuntos de dois endereços de rede.
- *Filter* (1.3.6.1.2.1.16.7): embasado na criação de filtros, encontra pacotes que se adequam aos filtros criados, podendo armazenar o tal pacote ou criar um evento associado a essa captura.
- *Capture* (1.3.6.1.2.1.16.8): permite que pacotes sejam capturados caso se adequarem aos filtros existentes no grupo *Filter.*

- *Event* (1.3.6.1.2.1.16.9): controla as definições para os eventos RMON1.

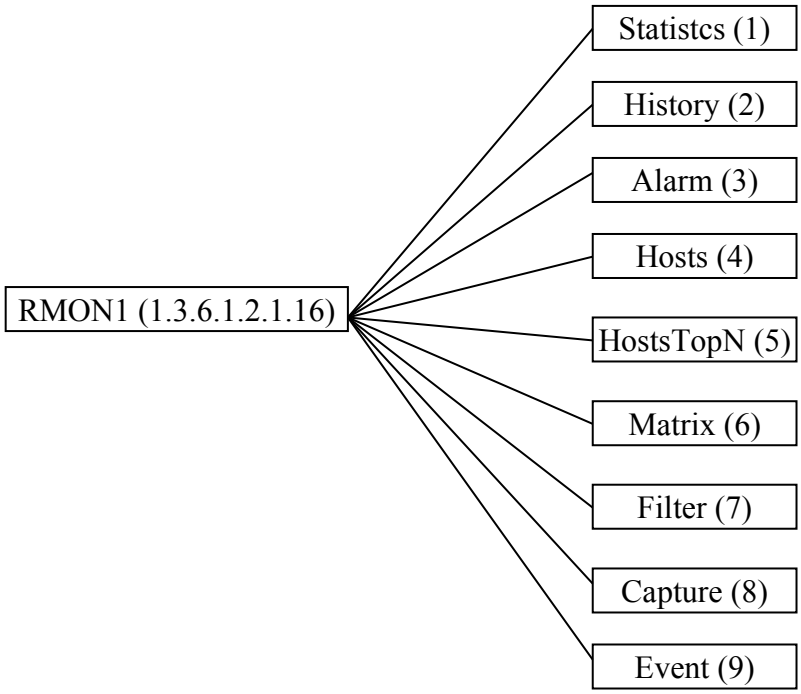

Figura 5.3: Estrutura da árvore da MIB RMON1

O RMON é desenvolvido para coletar informações e processá-las diretamente no agente. Esta forma de operação reduz o tráfego de rede e a capacidade de processamento necessária para a estação de gerência. Os *probes* RMON permitem a análise de tendências, agindo de maneira pró-ativa na detecção de problemas, mas também identificando aqueles já existentes, como a perda de pacotes ocasionada por excesso de tráfego.

1.16.3 O RMON2

Os esforços para a geração do RMON2 começaram em julho de 1994 com a criação de outro grupo de trabalho no IEFT. A meta era a elaboração de um padrão que proporcionasse estatísticas das camadas superiores do modelo OSI. Em 1997 foi especificada a RFC 2021 que definiu os padrões da MIB RMON2.

Um dos grandes benefícios do RMON2 é a monitoramento das camadas superiores ao nível de enlace, proporcionando uma visão mais completa da rede, não restrita ao segmento monitorado (figura 5.4). O RMON2 criou mais seis grupos gerenciados além daqueles definidos pelo RMON1 (figura 5.5). A seguir segue um resumo das funcionalidades de cada um desses grupos.

- *Protocol Directory* (1.3.6.1.2.1.16.11): contém um conjunto de informações sobre todos os protocolos suportados pelo *probe.* Este grupo lista os protocolos que podem ser decodificados. Esses protocolos pertencem a diferentes níveis do modelo OSI.
- *Protocol Distribuition* (1.3.6.1.2.1.16.12): coleta estatísticas agregadas da distribuição de tráfego dos protocolos.

- *Address Mapping* (1.3.6.1.2.1.16.13): faz o mapeamento de cada endereço MAC para os endereços do nível de rede.
- *Network Layer Host* (1.3.6.1.2.1.16.14): monitora os pacotes do tráfego de rede capturados pelo *probe,* permitindo ao gerente verificar através do roteador, por exemplo, todo o tráfego passante por este. Ele controla tanto o nível de rede quando o nível de aplicação.
- *Network Layer Matrix* (1.3.6.1.2.1.16.15): armazena estatísticas embasadas em informações de pares de endereços do nível de rede. As tabelas de dados são semelhantes àquelas apresentadas no RMON1 no grupo *hostTopN.*
- *Application Layer Host* (1.3.6.1.2.1.16.16): apresenta estatísticas sobre o volume de tráfego de entrada e saída das estações com base em endereços do nível de aplicação.
- *Application Layer Matrix* (1.3.6.1.2.1.16.17): disponibiliza estatísticas embasadas em pares de endereços de rede e analisa as aplicações envolvidas nessas comunicações.
- *User History* (1.3.6.1.2.1.16.18): combina mecanismos encontrados nos grupos *alarm* e *history.* Este grupo tem a finalidade de coletar e armazenar amostras de objetos previamente especificados.
- *Probe Configuration* (1.3.6.1.2.1.16.19): define parâmetros padrões de configuração para os recursos dos *probes.* Este grupo possibilita que as estações gerentes possam configurar automaticamente os seus agentes.
- *Rmon Conformance* (1.3.6.1.2.1.16.20)*:* descreve requisitos de conformidade para a MIB RMON2.

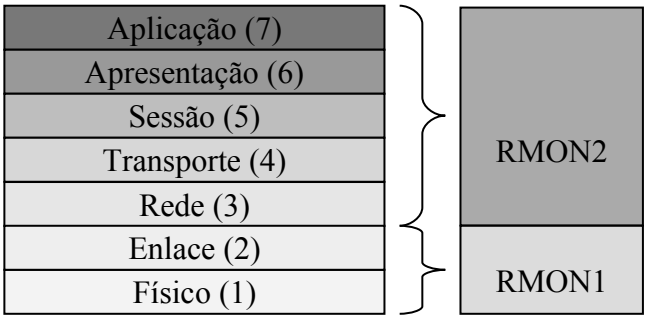

Figura 5.4: Monitoramento das MIBs RMON1 e RMON2 relativas aos níveis OSI

1.17A Utilização de SNMP nos Sistemas Detectores de Intrusão

A aplicação do protocolo SNMP para a implementação de sistemas detectores de intrusão é uma abordagem citada por muitos trabalhos da comunidade científica. Astithas (2002) buscou a integração entre tecnologias de gerenciamento e segurança na proposta de implementação do chamado SIDS (*Simple Intrusion Detection System*).

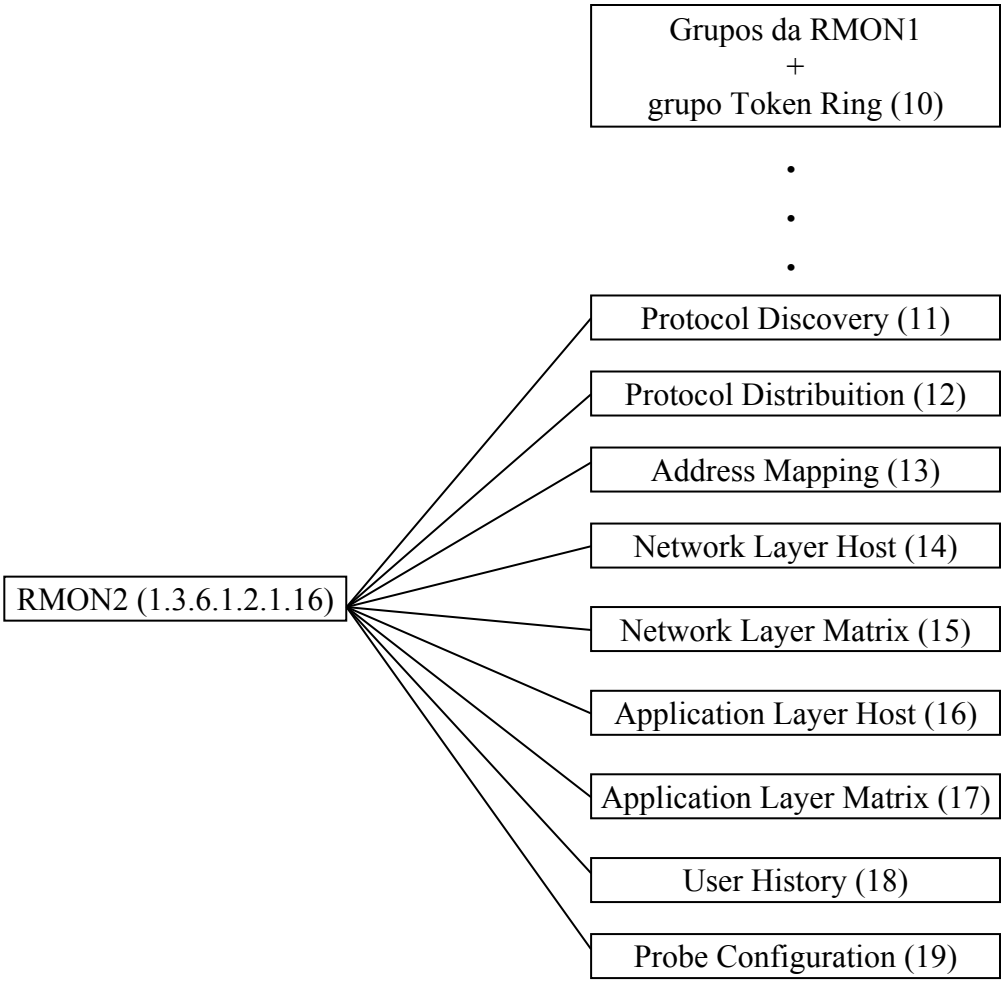

Figura 5.5: Estrutura da árvore da MIB RMON2

Trata-se de um sistema misto entre IDS de rede e IDS de *host,* de arquitetura distribuída, que possui entidades como coletores e analisadores de tráfego que utilizam SNMPv3 para a troca de informações. No entanto, essa troca de informações é feita através da modelagem de uma MIB própria. No artigo não fica muito claro quais técnicas são utilizadas para o tratamento das informações obtidas pelo analisador. Apenas é dito que se trata de um sistema embasado em regras. Zhang (2004-a) faz uma análise do SIDS e propõe o NM-IDS (*Network Management Intrusion Detection System*). Trata-se de uma Estrutura ainda mais complexa, mas um pouco melhor explicitada. O sistema de decisões é embasado em regras criadas no modelo de regras do IDS Snort. No entanto sistema descrito nesse artigo não foi implementado.

Para Qin (2002) há uma grande necessidade de se integrar as tecnologias de gerenciamento de rede e as tecnologias de segurança. Ele propõe a criação de um sistema detector de intrusão distribuído e hierárquico onde existam agentes de computador e agentes de rede. A estrutura desse IDS é dividida quatro módulos: agentes, intregradores, gerenciadores e gerenciadores de rede. Os agentes de computador são aplicados a servidores críticos da *intranet* e os agentes de rede podem ser instalados em dispositivos de rede. Esses agentes de computador são compostos por uma máquina de assinaturas, responsável por detectar os ataques conhecidos, uma máquina de análise de comportamento, que faz a detecção de anomalias nas variáveis monitoradas e uma máquina de análise de MIB, que, pelos comportamentos anômalos encontrados nos valores dos objetos gerenciados, tenta verificar comportamentos

condizentes com intrusões. O IDS utiliza os dados da MIBII para, através das variáveis dos grupos IP, ICMP, UDP e TCP, verificar anomalias no funcionamento dos sistemas monitorados. Nos testes do sistema implementado foram detectados corretamente ataques de negação de serviço feitos por *SYN-flood, ping-flood, UDP-flood* e *mix-flood.* No entanto, o próprio autor diz que a utilização de dados provenientes apenas da MIB II trouxe limitações ao sistema, o que poderia ser evitado caso trabalhasse com outras MIB combinadas.

Gaspary (2005) modela e implementa um sistema detector de intrusão distribuído e hierárquico que utiliza a linguagem PTSL (*Protocol Trace Specification Language*) para, através de quatro componentes (agente de ação, agente de monitoramento, gerenciador de nível médio e estação de gerenciamento) modelar assinaturas de ataques a serem detectados. O sistema utiliza extensões da MIB RMON2 para postar objetos que indicam o resultado de contadores associados ao número de vezes que determinada assinatura é encontrada.

As muitas abordagens para a combinação entre protocolos de gerência de rede e os mecanismos de segurança IDS têm no seu cerne a necessidade da integração entre essas duas tecnologias como forma de economizar recursos materiais e humanos e interligar essas duas áreas da ciência da computação. No entanto, em muitos casos, os sistemas criados são tão complexos que exigem que o administrador de segurança seja um especialista nas estruturas criadas pelo sistema detector de intrusão. Além disso, alguns dos IDS propostos têm a grande desvantagem de consumirem muitos recursos da rede monitorada destinados exclusivamente a atividade da detecção.

A LÓGICA DIFUSA

A lógica difusa (*Fuzzy*) nasceu em 1965 quando Lotfi Zadeh, da Universidade de Berkeley, publicou o artigo intitulado "*Fuzzy Sets*" (ZADEH, 1965). Este artigo esteve finalizado por dois anos sem, no entanto, nenhuma revista técnica tê-lo aceito devido ao teor inovador das idéias nele contidas (MCNEILL, 1993). Na época, era inconcebível admitir a lógica difusa. O artigo só foi publicado porque Zadeh era também editor da revista onde foi apresentado à comunidade científica (TANAKA, 1997).

O fato chave para a lógica difusa veio em 1974 quando Ebraham Mamdani da Universidade de Londres aplicou os novos conceitos para controlar, pela primeira vez, uma máquina a vapor. No entanto, uma aplicação industrial só apareceria seis anos depois e somente no começo da década de 1990 a nova técnica seria utilizada em larga escala (TANAKA, 1997).

As bases da lógica difusa estão ligadas ao fato dos seres humanos, baseados em situações imprecisas ou aproximadas, serem capazes de lidar com processos bastante complexos. A estratégia adotada pela mente humana é imprecisa, mas geralmente possível de ser expressa em termos lingüísticos. A teoria de conjuntos difusos e os conceitos da lógica difusa podem ser utilizados para a modelagem matemática dessa forma imprecisa de raciocínio.

O critério para utilização de termos como "alto" ou "jovem" depende da avaliação de cada pessoa, não existindo uma fronteira exata entre ser ou não "alto". Se, por exemplo, for feita a análise da altura de um homem com 1,80 m, alguns poderiam dizer se tratar de um indivíduo alto. No entanto, se jogadores de basquete forem fazer tal análise, certamente não concordarão com a primeira. As palavras ou expressões lingüísticas têm seus significados um tanto subjetivos e dependem do contexto de avaliação. A lógica difusa tenta modelar matematicamente esse tipo de raciocínio humano.

As teorias mais conhecidas para tratar da imprecisão e da incerteza são a teoria dos conjuntos e a teoria de probabilidades. No entanto, elas nem sempre conseguem capturar as peculiaridades das informações dadas por uma pessoa. A teoria dos conjuntos não é capaz de tratar o aspecto vago da informação e a teoria das probabilidades, na qual a probabilidade de um evento determina completamente a probabilidade de evento contrário, é mais adaptada para tratar de informações repetitivas do que aquelas fornecidas por seres humanos (SANDRI, 1999).

A lógica difusa compreende uma ampla teoria que inclui os conjuntos difusos, o raciocínio difuso e a lógica difusa. Os conjuntos difusos provêem da teoria de conjuntos tradicional. A lógica difusa estende a lógica convencional (binária). O raciocínio difuso utiliza a lógica difusa como mecanismo básico à construção de sistemas de inferência.

1.18Conjuntos difusos

Todos os dias, nas conversações entre técnicos, frases vagas como "o tráfego é alto" são processadas pelo cérebro humano sem se saber exatamente a precisão de tal informação, ou seja, sem se saber o quanto é alto o tráfego citado. Os conjuntos difusos modelam matematicamente essas afirmações vagas. Pode-se criar, por exemplo, o conjunto difuso dos "tráfegos altos" e associar aos elementos desse conjunto um número que diga o quanto o elemento é compatível com o conjunto analisado.

Na teoria clássica dos conjuntos, também conhecidos como *crisp,* expressões vagas não são permitidas. Nela, não se pode modelar um conjunto de "tráfegos altos", pois é necessário tornar a expressão exata para poder utilizar a teoria clássica. No entanto, não há objeções à modelagem clássica do "conjunto dos tráfegos maiores do que 10 Mbit/s", por exemplo, visto que existe um limiar exato entre pertencer ou não a tal conjunto.

Nos conjuntos *crisp*, o conceito de pertinência de um elemento a um conjunto fica bem definido. O processo pelo qual se determina quais elementos do universo estão contidos em um dado conjunto é definido como função característica. Ou seja, dado um conjunto A em um universo X, os elementos deste universo pertencem ou não àquele conjunto da forma expressa pela função característica f_A:

$$\mu_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$$

Estendendo o exemplo, podem-se definir ainda os conjuntos "tráfegos baixos", "tráfegos médios" e "tráfegos altos" da forma abaixo exposta:

Tráfegos baixos $\Rightarrow$ *até 5Mbit/s*

Tráfegos médios $\Rightarrow$ *5 a 10Mbit/s*

Tráfegos altos $\Rightarrow$ *mais de 10Mbit/s*

Dessa forma a função característica para tais conjuntos poderia ser representada de acordo com o gráfico da figura 6.1.

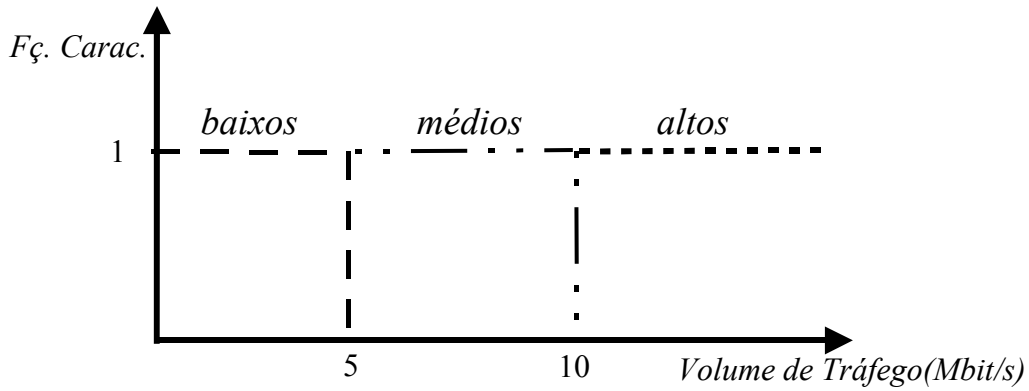

Figura 6.1: Funções características para os conjuntos *crisp* baixos, médios e altos.

Na teoria de Zadeh, no entanto, há uma generalização da função característica de modo que ela possa assumir um número infinito de valores no intervalo *[0,1]*. Dá-se o nome de grau de pertinência ao valor que indica o quanto o elemento tem a característica do dado conjunto. De forma semelhante ao que acontece na teoria

clássica, à função matemática que mapeia os graus de pertinência de cada elemento pertencente a um universo *X* dá-se o nome de função de pertinência. Assim, um conjunto difuso *A* em um universo *X* é representado por um conjunto de pares ordenados da seguinte forma:

$$A=\{\mu_A(x)/x\}\ x\in X$$

A expressão μ_A indica o quanto *x* é compatível com o conjunto *A*. Ou seja, retomando o exemplo dos "tráfegos altos", pode-se dizer que um determinado tráfego *x*, de 13Mbit/s, pertence ao referido conjunto com grau de pertinência = 0,8 (figura 6.2).

$$\mu_{Tráfegos\ altos}(x)/x = 0{,}8/x,\ \ x \in \text{conjunto dos tráfegos altos}$$

Na teoria difusa, as transições entre os conjuntos podem ser bem mais suaves. Dá-se o nome de variáveis lingüísticas à variável em cujos valores são compostos por conjuntos difusos (SANDRI 1999). A figura 6.2 demonstra um possível desenho das funções de pertinência para a variável lingüística *volume de tráfego*, com os conjuntos difusos tráfego baixo, tráfego médio e tráfego alto e apresenta o mapeamento do tráfego de 13Mbit/s do exemplo anterior.

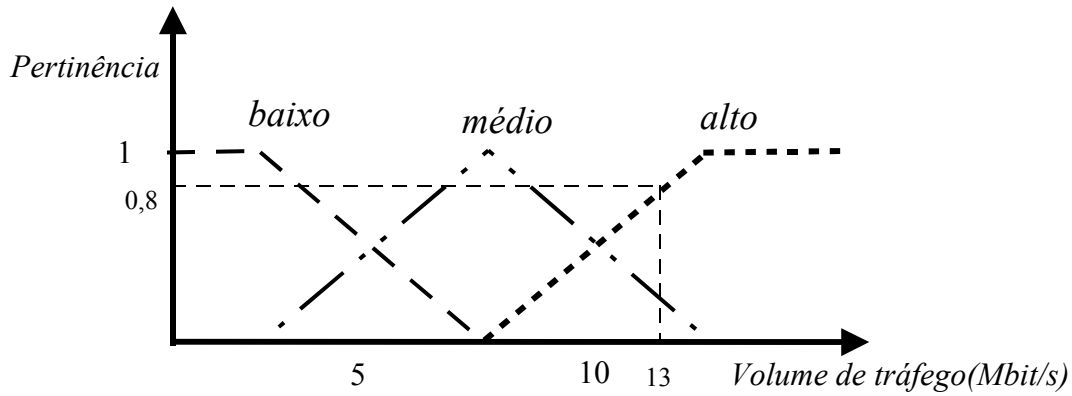

Figura 6.2: Funções de pertinência para a variável lingüística tamanho de tráfego.

A principal função das variáveis lingüísticas é fornecer uma maneira sistemática para uma caracterização aproximada de fenômenos complexos (SANDRI, 1999). Formalmente, uma variável lingüística é caracterizada pela quíntupla *(N,T(N),X,G,M)*, abaixo apresentada:

- *N:* nome da variável;
- *T(N):* conjunto de nomes dos valores lingüísticos;
- *X:* universo;
- *G:* regra sintática que gera os valores de *N* como uma composição dos valores de *T(N);*
- *M*: regra semântica para associar cada valor gerado por *G* um conjunto difuso em *X.*

Utilizando o exemplo apresentado, tem-se a seguinte formalização para a variável lingüística "Volume de Tráfego":

- *N:* volume de tráfego;
- *T(N):* baixos, médios e altos;
- *X: 0 a 1000* Mbit/s*;*

- *G:* não muito alto, muito baixo, muito médios, pouco altos, etc
- *M:* tráfegos baixos não são altos e não muito médios; tráfegos médios são não muito baixos e não muito altos; tráfegos altos não são baixos e nem muito médios.

1.18.1 Operações sobre conjuntos

Utilizando a teoria dos conjuntos *crisp*, dados os conjuntos A e B sendo subconjuntos do universo X, pode-se definir as operações de união, intersecção e complemento da seguinte maneira:

União: $A \cup B = \{x \mid x \in A \text{ ou } x \in B\};$

Intersecção: $A \cap B = \{ x \in A \text{ e } x \in B\};$

Complemento: $\overline{A} = \{ x \notin A \};$

Ou ainda, utilizando os diagramas de *Venn*, as operações referidas podem ser apresentadas de acordo com a figura 6.3.

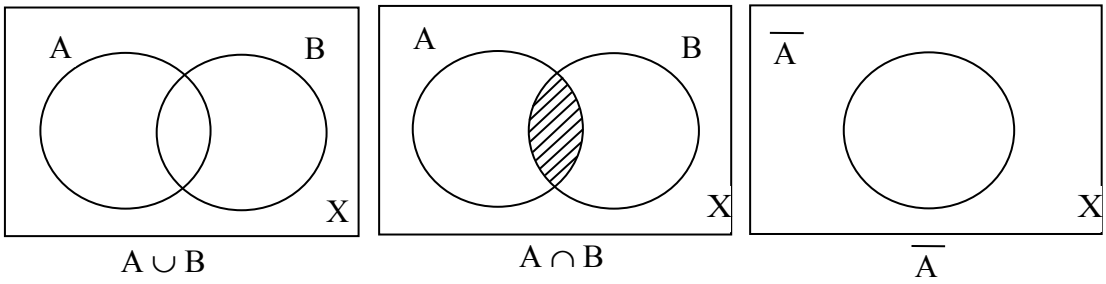

Figura 6.3: Diagramas de *Venn* para operações em conjuntos *crisp*

Para os conjuntos difusos, as operações de união, intersecção e complemento se comportam de maneira mais abrangente de tal forma que, sendo A e B conjuntos difusos temos as seguintes proposições:

- *União:* é o conjunto definido pela função de pertinência $\mu_{A\cup B}(x)=\mu_A(x) \vee \mu_B(x)$, onde:

$$\mu_A(x) \vee \mu_B(x) = \begin{cases} \mu_A(x) & \text{se } \mu_A(x) \geq \mu_B(x) \\ \mu_B(x) & \text{se } \mu_A(x) < \mu_B(x) \end{cases}$$

A união de conjuntos difusos é usualmente denotada por $\mu_A(x) \vee \mu_B(x)=max\{\mu_A(x),\mu_B(x)\}$. A notação *max* retorna sempre o valor mais alto da aplicação das duas funções de pertinência. Assim, a operação $max\{\mu_{TráfegoMedio}(x), \mu_{TráfegoAlto}(x)\}$ sempre devolve o maior valor entre as duas funções. O gráfico resultante dessa operação é mostrado na figura 6.4.

No intuito de estabelecer a notação utilizada, na figura 6.4, baixos, médios e altos são os conjuntos *fuzzy* que definem a variável lingüística volume de tráfego. No eixo das abscissas (x), dá-se o nome de *universo de discurso* ao mapeamento de todos os valores que a variável *volume de tráfego* pode assumir. Ou seja, o *universo de discurso* dessa variável garante que para todos os volumes de tráfego sempre haja um correspondente valor de pertinência (eixo das ordenadas).

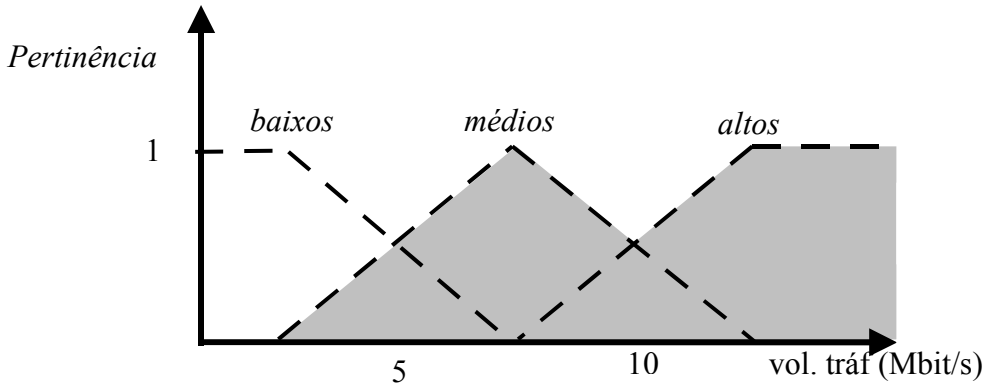

Figura 6.4: Operação de união para os conjuntos *fuzzy* médios e altos

- *Intersecção:* é o conjunto definido pela função de pertinência $\mu_{A\cap B}(x)=\mu_A(x)\wedge\mu_B(x)$, onde:

$$\mu_A(x)\wedge\mu_B(x)=\begin{cases}\mu_A(x) & \text{se } \mu_A(x)\leq\mu_B(x)\\ \mu_B(x) & \text{se } \mu_A(x)>\mu_B(x)\end{cases}$$

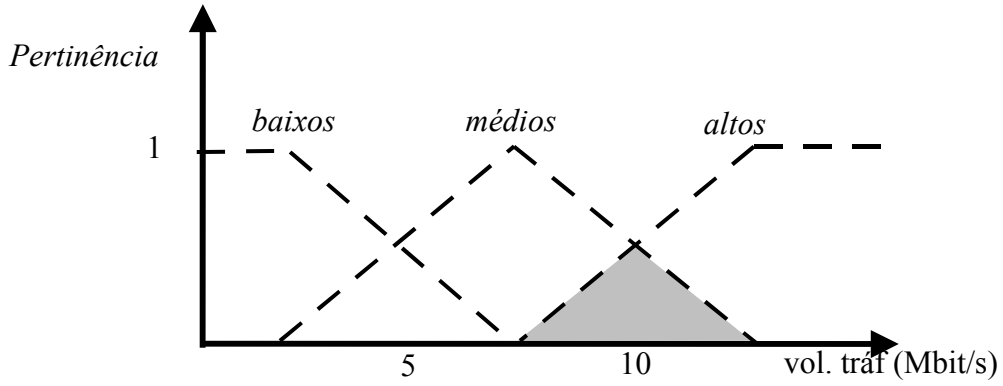

Figura 6.5: Intersecção entre os conjuntos *fuzzy* médios e altos

A intersecção de conjuntos difusos pode ser apresentada por $\mu_A(x)\wedge\mu_B(x)=min\{\mu_A(x),\mu_B(x)\}$. O operador *min* devolve o menor valor resultante da aplicação das duas funções de pertinência. Seguindo o exemplo, a operação $min\{\mu_{TráfegoMédio}(x), \mu_{TráfegoAlto}(x)\}$ retorna o menor valor entre as duas funções. O gráfico resultante dessa operação é mostrado na figura 6.5.

- *Complemento*: é o conjunto difuso definido pela função de pertinência:

$$\mu_{\neg A}(x)=1-\mu_A(x)$$

A operação $\mu_{\overline{TráfegoMédio}}$ resulta no gráfico mostrado na figura 6.6.

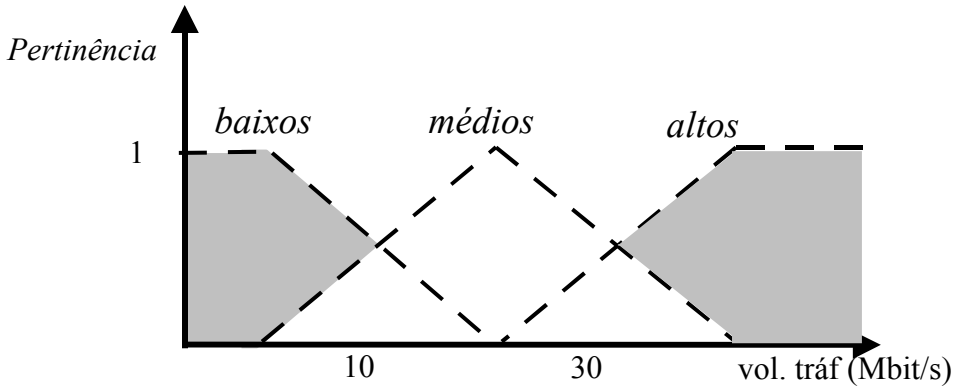

Figura 6.6: Operação de negação para o conjunto *fuzzy* médio.

1.19 Sistemas de Inferência Difusa (*Fuzzy*)

Um sistema de inferência difuso é o processo de, dados um conjunto de entradas, mapear uma saída pelo uso da lógica difusa (JANG, 1995). Os sistemas de inferência *fuzzy* tiveram sucesso nas aplicações como controle de automação, visão computacional, análise de decisão, classificação de dados e sistemas especialistas. Por causa de sua natureza multidisciplinar, eles são conhecidos por uma série de sinônimos: sistemas baseados em regras difusas, modelo difuso, controle *fuzzy* e sistemas *fuzzy*. A utilização desses sistemas é bastante eficiente para obter alto nível de flexibilidade e, pela representação do conhecimento armazenada nas suas base de dados, para permitir a modelagem de sistemas especialistas que se adequem às imprecisões (SINGARJU, 2004).

As técnicas de inferência *fuzzy* surgiram das pesquisas e projetos de E. H. Mamdani e ganharam espaço como áreas de estudo em diversas instituições de ensino, pesquisa e desenvolvimento, sendo até hoje uma importante aplicação da teoria dos conjuntos difusos. Ao contrário dos controladores convencionais em que o algoritmo de controle é descrito analiticamente por equações algébricas ou diferenciais, em um controle *fuzzy* utilizam-se de regras lógicas no algoritmo de controle, com a intenção melhor expressar a experiência, intuição e heurística humanas no controle de processo.

Os controladores difusos são robustos e de grande adaptabilidade, incorporando conhecimento que outros sistemas nem sempre conseguem acomodar. Também são versáteis, principalmente quando o modelo físico é complexo e de difícil representação matemática (SANDRI, 1999).

A estrutura típica dos sistemas de inferência difusos é apresentada na figura 6.7. As entradas desse sistema são valores precisos (*crisp*) que devem ser delimitados pelo conjunto de valores que podem assumir as variáveis monitoradas (universo de discurso). Geralmente, nos sistemas de controle, as entradas estão ligadas a sensores que monitoram algum parâmetro físico a fim de processar esses dados e gerar uma saída condizente à operação de algum mecanismo (JANG, 1995).

A etapa de *fuzificação* faz a identificação dos valores das variáveis de entrada, caracterizando o estado do sistema pelos valores encontrados nos universos de discurso. Estes valores são então *fuzzificados*, com a transformação da entrada "crisp" em conjuntos difusos para que possam se tornar instâncias de variáveis lingüísticas.

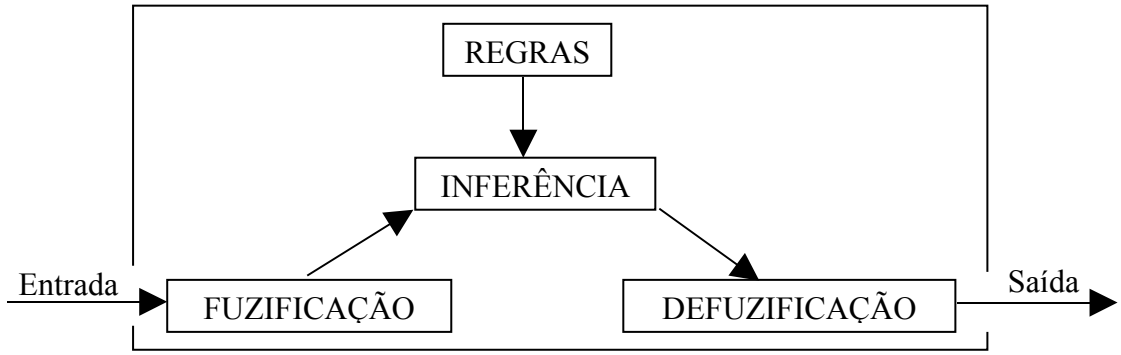

Figura 6.7: Estrutura de um sistema de inferência difuso.

A base de conhecimento de um sistema de inferência *fuzzy* consiste de uma base de dados e uma de regras que têm a função de caracterizar a estratégia de controle para o sistema criado. Na base de dados ficam armazenadas as definições sobre as variáveis lingüísticas, e na base de regras existe o conhecimento que será responsável pela inteligência do sistema especialista criado (SANDRI, 1999).

A base de regras é formada por estruturas do tipo:

Se <premissa> Então <conclusão>

As regras, juntamente com os dados provindos da etapa de *fuzzificação*, são avaliadas pelo procedimento de inferência, que, pela avaliação dessas regras, cria a situação que representa o estado do sistema no momento da aplicação da etapa de inferência.

Em geral as regras utilizadas na inferência têm a forma de um sistema de múltiplas entradas e uma saída (*MISO*). Como exemplo deste tipo de regra, tem-se a expressão abaixo.

Se tráfego_entrada é alto e tráfego_saída é baixo Então servidor_parado é grande

A regra acima poderia ser utilizada para modelar problemas em um servidor de e-mail,situado no interior de uma rede privada, protegido por um *firewall* corporativo e responsável pelo repasse (*relay*) de mensagens, por exemplo. Para esse tipo de serviço, devido à natureza das operações realizadas, pode-se concluir que em média o tráfego de entrada deve ser bastante parecido com o tráfego de saída. A utilização da regra acima poderia alertar aos administradores de segurança sobre possíveis problemas encontrados na operação de tal servidor.

Nas regras criadas para os sistemas de inferência difusos, as premissas são relacionadas pelos conectivos lógicos, dados pelo operador de conjunção (*e*), pelo operador de disjunção (*ou*) e pelo operador de negação (*não*) (GOMEZ, 2002-b). Os valores resultantes da junção de expressões como "*tráfego_entrando é alto"* e "*tráfego_saindo é baixo"* depende do método de inferência escolhido. Supondo que o resultado da expressão *tráfego_entrando é alto* fosse *0,7* (premissa *a*) e *0,9* o da expressão "*tráfego_saindo é baixo"* (premissa *b*), a tabela 6.1 mostra alguns operadores de inferência que poderiam ser usados para avaliar o resultado das relações entre as duas expressões. A utilização de Mamdani, por exemplo, resultaria 0,7.

Dependendo do método de inferência escolhido, os resultados da etapa de inferência serão diferentes, afetando as saídas desejadas. A figura 6.8 mostra uma inferência feita pelo método de Mamdani.

Tabela 6.1: Operadores de inferência.

IMPLICAÇÃO	NOME DO OPERADOR
max(1-a,b)	*Kleene-Diemes*
min(1-a+b,1)	*Lukasiewicz*
1-a+a.b	*Reichenbach – Estocástica*
Max(1-a,min(a,b))	*Zadeh-Wilmott*
Min(a,b)	*Mandani*
a.b	*Larsen*

Fonte: SANDRI, 1999

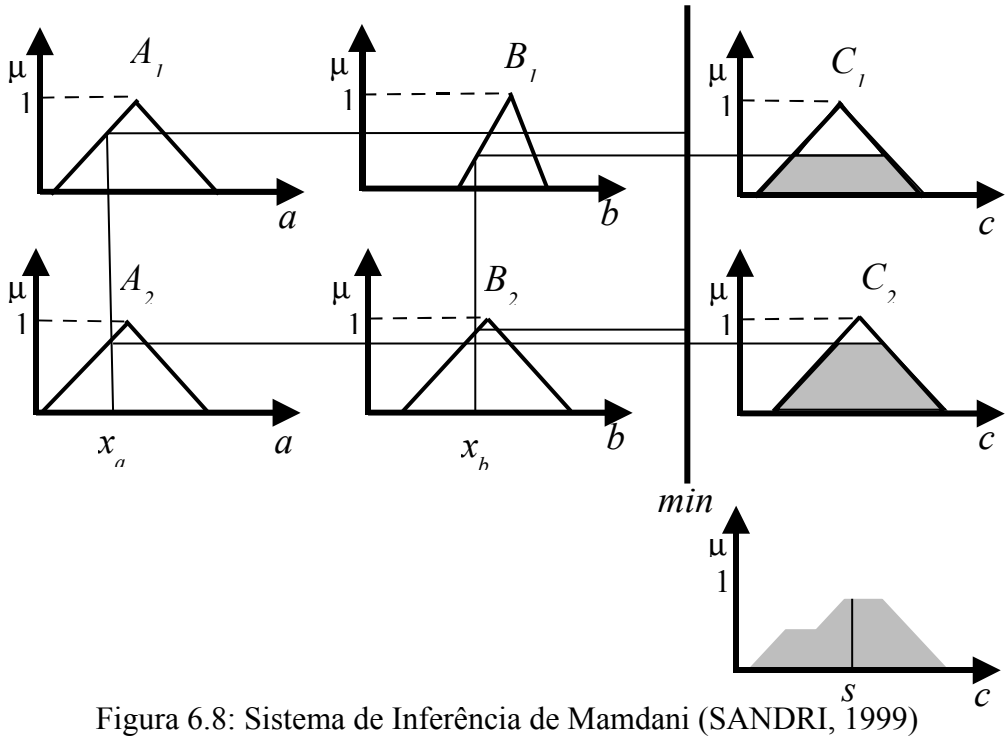

Figura 6.8: Sistema de Inferência de Mamdani (SANDRI, 1999)

A etapa de *defuzzificação* é, geralmente, utilizada para obter uma única ação de controle precisa. O procedimento consiste na identificação do domínio das variáveis de saída em um correspondente universo de discurso e, com a ação de controle difusa inferida, concluir-se uma saída precisa (SANDRI, 1999).

Os métodos de *defuzzificação* mais utilizados estão abaixo apresentados.

- *Primeiro máximo:* encontra o resultado de saída através do ponto em que o grau de pertinência dos dados provindos da etapa de inferência atinge o primeiro valor mais alto.
- *Média dos Máximos:* resulta no ponto médio entre os valores que têm o maior grau de pertinência.
- *Centro de Área:* a saída é o centro de gravidade dos resultados da etapa de inferência.

O projeto de um sistema de inferência difuso deve verificar quais os melhores métodos a serem aplicados no processo de inferência e no processo de *defuzificação*. De acordo com a natureza do controle que se deseja construir, a escolha desses métodos

direciona as ações resultantes do processo de inferência. Os tipos de modelos de inferência mais encontrados na literatura são os modelos clássicos como o de Mandani e Larsen e os modelos de interpolação como os de Takagi-Sugeno e o de Tsukamoto (MCNEILL, 1993).

No trabalho aqui proposto, optou-se pela utilização dos modelos de Mamdani para a inferência e dos primeiros máximos e centro de área para a *defuzificação*. O modelo de Mamdani foi escolhido por propiciar um relacionamento mais condizente com o esperado de um IDS, pois, por resultar sempre o menor valor da aplicação das premissas das regras, impede que picos nos valores de alguma premissa sejam transferidos ao resultado final do processo. O método de primeiros máximos proporciona a diminuição no número de falsos-positivos pela captura apenas do primeiro ponto mais alto do gráfico resultante da inferência de Mamdani. Ele foi utilizado na abordagem da criação de regras apenas para os comportamentos condizentes com ataques disparados contra as redes monitoradas. O método do centro de área foi usado para modelar o comportamento da abordagem onde os administradores de segurança também criam regras indicativas de comportamento normal.

1.20 Sistemas Especialistas Difusos

Aplicar métodos *fuzzy* para o desenvolvimento de IDS traz algumas vantagens comparadas com os métodos clássicos (IDRIS, 2006). Muitos sistemas detectores de intrusão têm sido desenvolvidos como sistemas especialistas que utilizam lógica difusa (MANIC, 2001) (DIKERSON, 2001) (ORFILA, 2003) (SILVEIRA, 2004) (IDRIS, 2006). Sistemas especialistas são programas de inteligência artificial que atuam como consultores inteligentes. Eles permitem que o conhecimento e a experiência de um ou mais especialistas sejam usados para a construção de uma base de conhecimento que é utilizada nas decisões tomadas pelo sistema (TAROUCO, 1990).

O ponto inicial na construção de um sistema especialista *fuzzy* é obter uma série de regras do tipo "Se-Então" provenientes de especialistas humanos. O passo seguinte é combinar essas regras em um único sistema onde os princípios dessa combinação delimitam as características do sistema criado (SINGARAJU, 2004).

Um sistema especialista consiste de três componentes principais: uma base de conhecimento, uma máquina de inferência e uma interface para o usuário (figura 6.9).

A base de conhecimento contém os fatos, idéias, relacionamentos e interações de um domínio limitado. A máquina de inferência tem o poder de derivar conclusões que são apresentadas ao usuário. A base de conhecimentos pode ser constituída de regras de produção "Se-Então", que fazem a experiência técnica dos especialistas poder ser aplicada à máquina de inferência. A interface, além de implementar a comunicação com o usuário, permite que novos conhecimentos sejam entregues ao sistema. O usuário pode também requerer que o sistema especialista explique o processo pelo qual uma dada conclusão foi alcançada. (TAROUCO, 1990).

Existem muitos métodos para descrever o conhecimento armazenado em um sistema especialista. A classificação desses métodos consiste na verificação de como o conhecimento é obtido e para quem ele é destinado. Existem os sistemas especialistas consultores, que espelham os diálogos humanos; os monitores, baseados no sensoriamento indicativo do estado do ambiente; os servidores, baseados em diálogos; os agentes que são baseados em sensores (tabela 6.2).

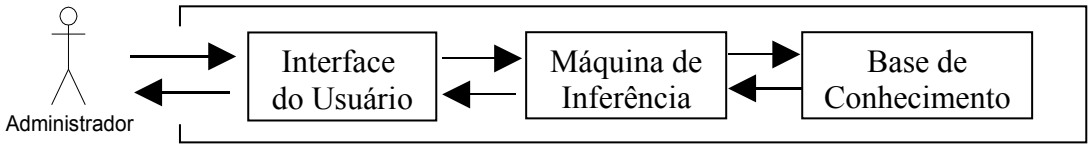

Figura 6.9: Esquema geral de um sistema especialista (TAROUCO, 1990)

Tabela 6.2: Categoria de sistemas especialistas.

		ENTRADA DE	
		PESSOAS	MUNDO
SAÍDA PARA	PESSOAS	Consultores	Monitores
	MUNDO	Servidores	Agentes

Fonte: TAROUCO, 1990

1.21 A *Clusterização* Difusa.

A *clusterização* ou análise de cluster é a tarefa de agrupar ou classificar objetos em categorias distintas (BACKER, 1995). Esse processo é uma das atividades mais comuns e primitivas do homem e vem sendo intensificada em função do volume de dados disponíveis em nossa sociedade da informação.

O problema de *clusterização* pode ser tratado segundo a abordagem convencional, na qual cada objeto deve ser classificado única e totalmente em um determinado conjunto, e segundo a abordagem *fuzzy*, mais flexível, na qual um objeto pode ser classificado em várias categorias, com diferentes graus de associação em cada uma delas. O uso da teoria de conjuntos difusos torna-se conveniente, uma vez que muitas categorias comumente encontradas e empregadas na *clusterização* possuem limites vagos e podem pertencer a diferentes conjuntos com diversas pertinências (BACKER, 1995).

A criação desses conjuntos difusos pode ser feita de acordo com alguns algoritmos. Este trabalho descreve e utiliza o *fuzzy c-means*, criado por J. C. Bezdek em 1981 e descrito a seguir.

1.21.1 O Algoritmo *Fuzzy* C-means.

O método de *clusterização fuzzy c*-means pode ser desenvolvido por meio de um algoritmo baseado na minimização de um índice de desempenho, que indica o quanto a partição gerada é condizente com a partição esperada. Os conjuntos que apresentam *clusters* compactos, bem separados, são mais rapidamente solucionados utilizando este método.

O algoritmo aqui apresentado assume como entrada os seguintes parâmetros:

- c: o número de centros a serem criados;
- m: o índice de *fuzificação* pertencente ao intervalo $(1,\infty)$. Está relacionado à distribuição dos conjuntos *fuzzy*, informando se a passagem entre uma partição e outra será mais suave ou mais abrupta. Tipicamente esse valor é configurado entre 1 e 2. Os testes apresentados no capítulo 8 utilizam o valor 1.2 para esse índice.
- ε: um número maior que zero, usado como um critério de parada.

O algoritmo pode ser descrito por três passos básicos. Assim sendo, dado um conjunto de dados $X = \{x_1,x_2,...,x_n\}$, uma partição *fuzzy* de X é uma família de c subconjuntos *fuzzy* de X, denotada por $P = \{A_1, A_2, ..., A_c\}$. Os termos A_i representam o conjunto das pertinências de todos os elementos de X associados ao conjunto A_i.

- ***Passo 1:*** Cálculo dos centros dos clusters. Na primeira vez que o laço é executado podem ser atribuídos valores aleatórios para os centros. Nos passos seguintes, a aplicação da expressão abaixo fornece os novos centros das partições criadas.

$$v_i = \frac{\sum_{k=1}^{n} [A_i(x_k)]^m x_k}{\sum_{k=1}^{n} [A_i(x_k)]^m}$$

  Nessa expressão $A_i(x_k)$ simboliza a pertinência do elemento x_k ao conjunto *fuzzy* A_i.

- ***Passo 2:*** Cálculo das pertinências dos novos centros dos clusters para todos os elementos de X. A expressão abaixo realiza essa tarefa.

$$A_i^{(t+1)}(x_k) = [\sum_{j=1}^{c} (\frac{| x_k - v_i^{(t)} |^2}{| x_k - v_j^{(t)} |^2})^{\frac{1}{m-1}}]^{-1}$$

  Nessa expressão, $A_i^{(t+1)}$ representa o novo valor da pertinência do elemento x_k na partição A_i e v_i é o centro da partição *fuzzy* i.

- ***Passo 3:*** Cálculo da condição de parada. Esse cálculo é feito pela comparação entre as novas pertinências geradas e aquelas imediatamente anteriores. Se, para todos os elementos de X, o máximo de diferença entre a pertinência anterior e a nova pertinência for menor do que ε, o algoritmo deve parar. Do contrário o fluxo volta para o passo inicial.

Para a modelagem e implementação do sistema detector de intrusão descrito por este trabalho, o algoritmo *fuzzy c-means* foi usado para criar os conjuntos difusos a partir dos dados das variáveis SNMP monitoradas. A adoção desse algoritmo trouxe maior confiabilidade aos conjuntos *fuzzy* criados.

1.22 Tecnologia Difusa na Implementação de IDS.

Devido à natureza inexata dos dados provenientes de sensores bem como das informações passadas pelos especialistas, a lógica difusa tem sido muito utilizada na modelagem e implementação de sistemas detectores de intrusão. Estatísticas quantitativas e outros métodos tradicionais que IDS de reconhecimento de assinaturas utilizam, pela natureza estática com que realizam esses reconhecimentos, podem gerar muitos falso-positivos (MANIC, 2001). A utilização da tecnologia difusa, por sua vez, faz as entradas dos sistemas especialistas serem quantizadas de acordo com as variáveis lingüísticas construídas e, consequentemente, as respostas dos sistemas especialistas difusos tendem a ser melhores adaptadas ao ambiente monitorado (ORFILA, 2003).

Muitos autores enaltecem a aplicação da lógica difusa em IDS. Em Orfila (2003), por exemplo, foi feita uma comparação entre os resultados provindos de IDS que operam sob a teoria dos conjuntos *crisp* e outros que trabalham com tecnologia difusa.

No artigo, há a indicação das melhoras na detecção de anomalias e na diminuição de falso-positivos com o uso de *fuzzy*.

Os métodos e modelos para a criação de sistemas detectores de intrusão embasados em tecnologia difusa, possibilitam a construção de IDS com características díspares que utilizam de diferentes tecnologias em prol da detecção de intrusão. Em Manic (2001), são usados modelos de Markov para verificar o espectro de funcionalidades da rede monitorada e é aplicado *fuzzy* na diferenciação entre os estados de ataque e não-ataque.

Dikerson (2000) constrói o *Fuzzy Intrusion Recognition Engine* (FIRE), sistema que faz a criação das variáveis lingüísticas utilizando alguns dados de cabeçalhos IP destinados a determinadas portas TCP. Para cada uma dessas portas, é feita uma análise da quantidade, variância e unicidade dos pacotes capturados. No FIRE os administradores criam as regras que são utilizadas para, com o método de Mamdani de inferência, detectar ataques de varredura de portas.

O IDS de Idris (2006), no momento em que descobre que um determinado endereço IP enviou pacotes maliciosos, passa a encaminhar ao endereço fonte todos os pacotes enviados após essa detecção. Esse contra-ataque é problemático por poder ser utilizado pelos criminosos para, através do forjamento de IP fonte, realizarem outros ataques usando a própria rede onde o IDS foi instalado. Para evitar essa situação, o autor cita os filtros colocados em roteadores a fim de evitar que, de redes conhecidas por esses roteadores, possam ser enviados pacotes com endereços IP fonte forjados. No entanto, essa configuração precisaria ser adotada por todos os roteadores da malha Internet.

Gomez (2002-a) afirma que o problema da detecção de intrusão se resume na distinção entre duas classes: a meta é classificar padrões de comportamento do sistema em duas categorias (normal e anormal), usando padrões de ataques conhecidos, que pertencem a classe "anormal", e padrões do comportamento normal. Essa é a grande dificuldade na utilização de classificadores *fuzzy.* Exigir que os administradores de segurança simulem comportamentos anôma-los para fazer o IDS distinguir entre os comportamentos é, por vezes, impraticável.

Silveira (2004), coletando periodicamente os logs de um *sniffer*, analisou o tráfego de um dado segmento de rede para criar, através do algoritmo *fuzzy cmeans,* quatro conjuntos difusos (pequeno, normal, aceitável e anormal) associados a contadores e somadores no intuito de tecer um perfil do tráfego passante. A modelagem do IDS foi feita utilizando o método de Mamdani, onde as regras de inferência foram aplicadas sobre esses quatro conjuntos nebulosos. O artigo apresenta uma pequena porção dessas regras que, confeccionadas pelos administradores de uma determinada rede, não são analisadas no intuito de apresentar sobre qual embasamento foram feitas.

Para a implementação do sistema detector de intrusão proposto por este trabalho foi utilizado o algoritmo de Mamdani, operando sobre regras, construídas pelos administradores das redes onde o IDS for instalado, e aplicadas sobre variáveis lingüísticas criadas através da aplicação do método *fuzzy c-means*. Como será visto no próximo capítulo, o administrador necessita informar ao sistema quais os objetos que precisam ser avaliados para posteriormente ter a capacidade de criar os gráficos das variáveis lingüísticas. De posse dessas variáveis lingüísticas, o passo seguinte é indicar quais as regras que se deseja utilizar para a detecção de ataques.

MODELAGEM DO IDS

A modelagem do sistema de detecção de intrusão proposto por este trabalho se embasou na divisão entre um software para configuração, chamado Configurador, um programa para a criação de conjuntos difusos, chamado Clusterizador e um software para a operação do IDS, chamado de Fids. Essa divisão foi necessária para modelar e implementar os componentes desse sistema que têm diferentes funções.

O Fids precisa ser rápido, por isso foi implementado em uma linguagem de boa performance: c++. Além da performance, essa linguagem foi escolhida por propiciar a programação orientada a objetos. De forma semelhante, o Clusterizador também foi desenvolvido em c++. Embora possa ser executado em paralelo à operação do Fids, para esse módulo também é desejável uma linguagem mais eficiente sob pena de despender muito processamento na geração dos conjuntos difusos. No caso do Configurador, no entanto, foi necessária que sua implementação fosse realizada sobre alguma linguagem que de fácil acesso a recursos gráficos, que permitisse criar aplicativos web. A linguagem escolhida foi PHP.

Para a descrição da modelagem do sistema aqui proposto serão utilizados diagramas UML no intuito de apresentar as especificações e características principais desta modelagem. A figura 7.1 expõe o diagrama de componentes onde é possível constatar a independência na operação dos três componentes supracitados.

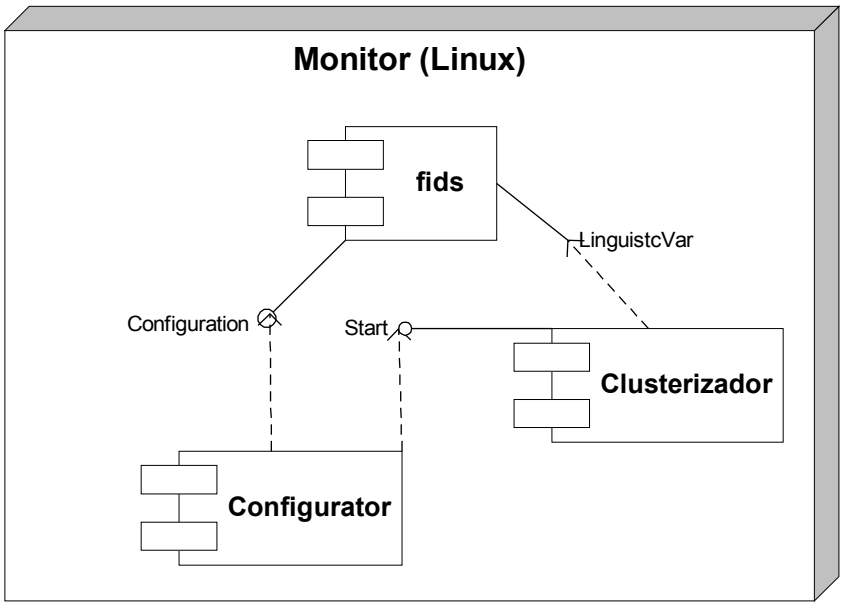

Figura 7.1: Diagrama de componentes do IDS

1.23 Modelagem do Configurador

Inicialmente, como forma de introdução na descrição deste IDS, será analisada a modelagem do Configurador. Neste componente do sistema não há a necessidade de velocidade, e sim de facilidade e clareza na sua configuração. A única forma do administrador de segurança ter acesso ao sistema é através deste configurador. A figura 7.2 apresenta o diagrama de casos de uso para este módulo. Uma vez que esse software realiza o interfaceamento entre o administrador de segurança e o sistema detector de intrusão, nesta figura podem ser observados todos casos de usos do IDS criado.

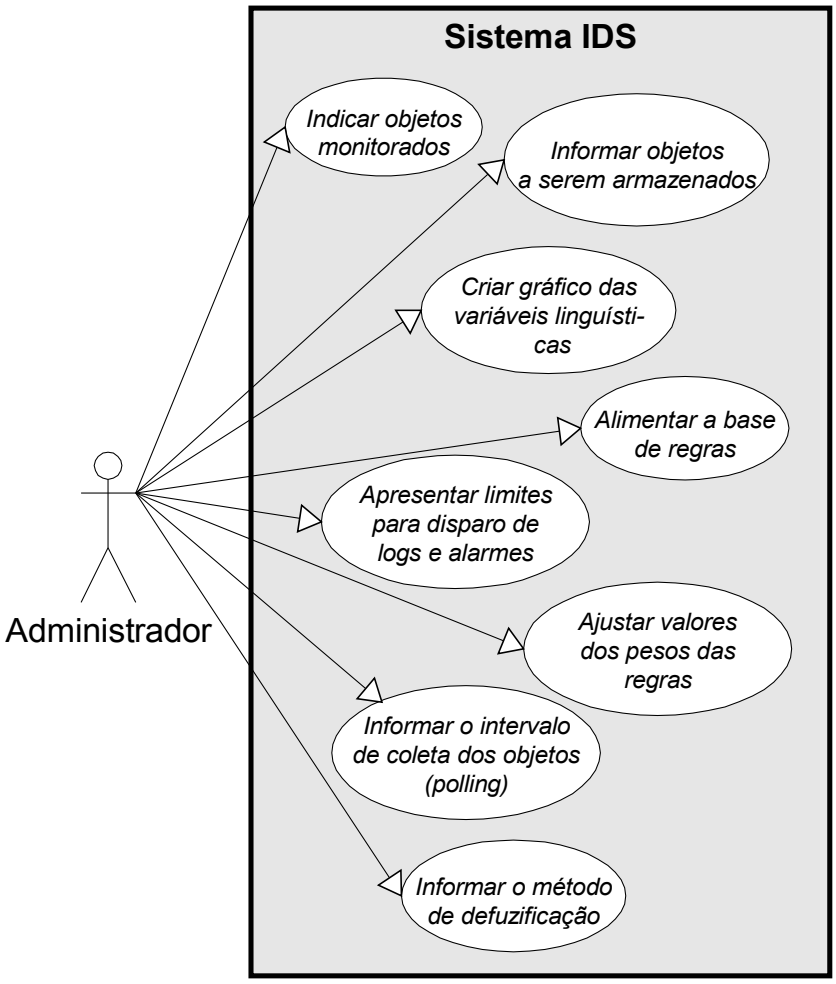

Figura 7.2: Diagrama de casos de uso do IDS

Abaixo seguem informações detalhadas sobre todos os casos de usos do Configurador.

- *Indicar objetos monitorados:* o administrador precisa informar ao sistema quais os objetos da MIB RMON2 a serem buscados e em quais agentes esses objetos serão capturados. É necessário também fornecer a comunidade SNMP de leitura desses objetos. A seleção dos objetos visados se faz necessária para diminuir a banda de rede utilizada pelo IDS para a sua operação.
- *Informar os objetos a serem armazenados:* o administrador pode indicar quais dos objetos informados no caso de uso anterior precisam ser armazenados para a posterior aplicação do algoritmo *fuzzy* c-means. É necessário informar ao sistema o tempo que ele precisará ficar armazenando

os dados dos objetos capturados para, depois desse período, ser possível a geração do gráfico das variáveis lingüísticas.

- *Criar gráfico das variáveis lingüísticas:* é possível gerar manualmente os gráficos das variáveis lingüísticas para que ocorra a *fuzzificação* provinda dos objetos da MIB. Essa operação é necessária para aquelas situações em que o administrador não dispõe do tempo para a criação automática dos gráficos das variáveis lingüísticas ou precisa indicar ao sistema algum gráfico que não segue o padrão proposto pela geração via *c-means*. Um exemplo dessa aplicação é a situação em que há o monitoramento do tráfego destinado a alguma porta TCP que não tenha serviços disponibilizados. Essa operação, de forma semelhante aos *honeypots* antes apresentados, se aplicada a sistemas internos ao *firewall* privado, pode identificar ataques provenientes de computadores presentes na mesma rede. A criação manual do gráfico das variáveis lingüísticas relacionadas a esse tipo de observação é necessária. É possível, por exemplo, gerar um gráfico com o valor constante em 1 para a pertinência associada ao objeto capturado. Assim, se uma regra for construída envolvendo esses acessos às portas sem serviço, o IDS proposto detectará ataque caso ocorram acessos ilegais a essas portas.

- *Alimentar a base de regras:* de acordo com o seu conhecimento da estrutura e da atuação da rede monitorada pelo sistema, o administrador poderá criar regras que informem o comportamento normal ou situações anormais indicativas de atividade ilegal. Para isso, devem ser geradas regras que envolvam os valores das variáveis lingüísticas automática ou manualmente criadas no configurador.

- *Apresentar limites para o disparo de logs e alarmes:* todo o ciclo de coleta (*pooling*) do IDS resulta em um valor entre zero e um. Neste caso de uso o administrador deve informar quais os limites para que sejam escritos os históricos (*log*) da operação do IDS e os limites para que sejam disparados alarmes via e-mail e via SMS (*short message service*). Caso se necessite históricos de toda a operação do sistema detector de intrusão, basta indicar o valor zero no limite para a geração de logs.

- *Ajustar valores dos pesos das regras:* a toda regra criada do configurador é associado um peso equivalente a 1. No entanto, é possível alterar esses pesos para que as regras que gerarem falsos positivos ou falsos negativos sejam reajustadas ao ambiente monitorado.

- *Informar o intervalo de coleta dos objetos:* é necessário informar o tempo de espera até que o sistema repita todo o ciclo de coleta e análise. Dependendo das características das redes monitoradas o administrador pode escolher entre um tempo pequeno, possibilidade em que aumentará o consumo de banda e recursos destinados ao IDS, e um tempo longo, onde o consumo de recursos diminui junto com a eficácia do detector de intrusão. A escolha por um intervalo de *polling* demasiado grande faz o sistema perder a capacidade de alertar rapidamente os administradores em caso de alguma anormalidade detectada.

- *Informar o método de defuzificação:* este caso de uso é necessário para que seja indicada ao Fids a forma utilizada na etapa de *defuzificação* do processo de inferência. O administrador pode escolher entre o método dos primeiros

máximos e o método do centro de área. O método dos primeiros máximos é indicado para abordagens onde são criadas apenas regras indicativas de anormalidade. O método do centro de área pode ser usado para configurações em que também são criadas regras para os casos de normalidade. O capítulo 8 fornece exemplos dessas duas abordagens.

A operação do configurador é feita por um algoritmo no qual são realizados todos os casos de uso apresentados. A utilização de um diagrama de atividade tem a capacidade de descrever a seqüência de operações necessárias à configuração do IDS. Os diagramas de atividade se diferem dos fluxogramas, pois estes são limitados exclusivamente a processos seqüenciais, enquanto aqueles fazem da representação paralela a sua principal característica (FOWLER, 2005). A figura 7.3 apresenta o diagrama de atividades para o configurador.

O primeiro laço apresentado no diagrama 7.3 indica a possibilidade de leitura das configurações já informadas ao sistema. A leitura é feita do mesmo arquivo de configuração consultado pelo IDS, para impedir que o Configurador apresente informações diversas daquelas em operação.

No segundo laço há possibilidade de se acrescentar ou retirar agentes ou objetos monitorados. Para essas modificações é necessário indicar os respectivos endereços IP, os objetos (OIDs) da MIB RMON2 e as comunidades necessárias às leituras SNMP.

A etapa de apresentação das variáveis lingüísticas se caracteriza pela possibilidade de escolha entre o método automático (*c-means*) e o método manual. Se for escolhido o método manual o administrador precisa informar o formato e os limites associados aos gráficos das variáveis lingüísticas criados.

A alimentação das regras deve ser feita através da escolha das variáveis lingüísticas e dos valores que essas deverão assumir. É possível criar regras indicativas de comportamento anormal e de comportamento normal. A aplicação do método de Mamdani traz ao sistema o poder de aplicar os valores obtidos nos objetos capturados a todas as regras criadas pelos administradores de segurança. Caso o administrador gere apenas regras indicativas de anormalidade o sistema implicitamente conclui que é normal todo o comportamento que não está descrito nas regras criadas.

É necessário indicar quais os limites para a geração de alarmes. Os históricos detalhados (logs) podem ser configurados para sempre escreverem em disco suas conclusões. Assim, mesmo que não ocorra um alarme, o administrador pode observar os resultados da aplicação das regras, no intuito de ajustar algum falso-negativo. Existem dois tipos de alarmes, um para o envio de mensagens via e-mail e outro para o envio de mensagens pelo celular (SMS). Essa divisão se deve ao fato de, caso o resultado não seja demasiado alto, mas, mesmo assim, expressivo, o administrador pode optar por apenas enviar um e-mail para um endereço configurado. No entanto, se o resultado deixar poucas dúvidas sobre o comportamento anômalo da rede, a mensagem SMS tem a capacidade de notificar o administrador mesmo que este não esteja em seu ambiente de trabalho.

O sistema propicia a realimentação através do ajuste dos pesos associados às regras criadas. Se ocorrer algum falso-positivo ou falso-negativo, o administrador tem a possibilidade de ajustar os pesos das regras que concluíram erroneamente o estado do sistema durante a operação do sistema detector de intrusão.

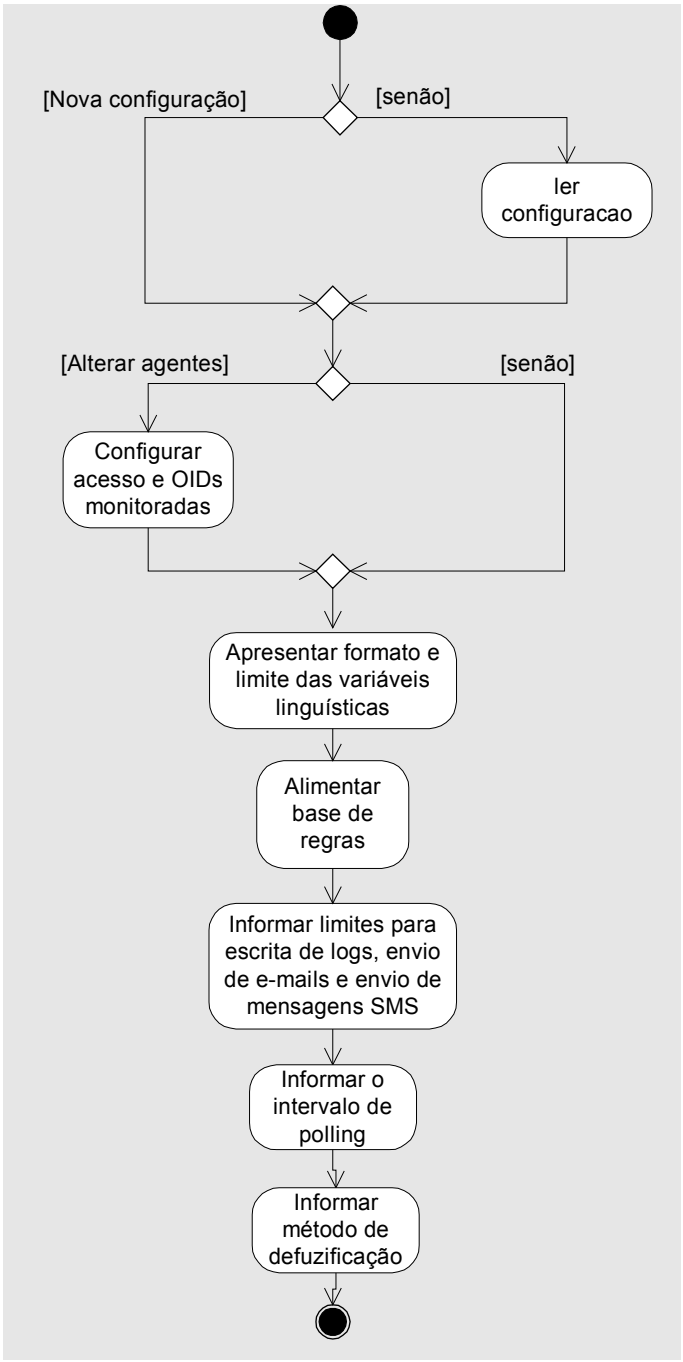

Figura 7.3: Diagrama de atividades do IDS

1.24 Modelagem do Fids

A modelagem do Fids foi proposta com o objetivo de deixar esta parte do sistema detector de intrusão o mais eficiente possível. Os próximos diagramas UML elucidam a operação deste componente.

O diagrama de estados apresenta as fases em que o Fids divide sua operação. Assim como apresentado na figura 7.4, o Fids trabalha sobre os quatro estados abaixo apresentados.

- O estado *OID Search* é aquele em que o sistema vai consultar os agentes para a obtenção dos objetos da MIB RMON2. Se um agente não conseguir

responder no tempo estipulado, o IDS ignora as regras que contenham os objetos desse agente.

- No estado *Rules Test* ocorre o processamento dos valores resultantes da captura dos objetos referenciados nas regras criadas pelo administrador. Todo esse processamento é feito de forma otimizada, pois é utilizado a linguagem c++, que possui boa performance.
- O estado *Attack* é disparado pelo sinal *HighFuzzyResult* (alto resultado do sistema difuso). Nesse estado o sistema enviará e-mails para os endereços configurados e, se necessário, também mensagens para celular. Essas emissões de mensagens ocorrem pelo uso do próprio serviço de entrega de e-mails do servidor onde o IDS é instalado.
- O estado *Idle* é disparado pelo sinal *LowFuzzyResult* (baixo resultado do sistema difuso), ocorrido sempre que o estado *Rules Test* devolver um valor abaixo do limite para ser considerado ataque. Neste estado todo o sistema fica aguardando o tempo de fazer a próxima consulta. Quando ocorrer o sinal de timer o sistema fará novamente a busca pelos OIDs, reiniciando o ciclo.

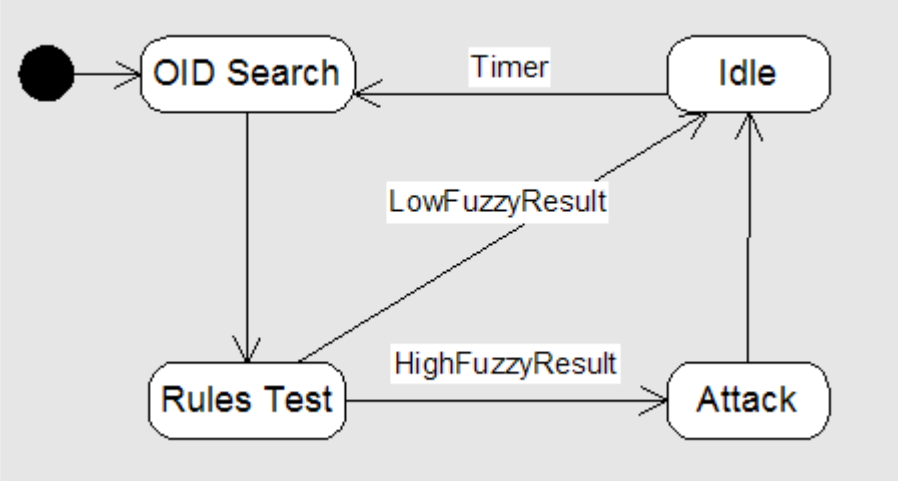

Figura 7.4: Diagrama de estados do IDS

O diagrama de classes descreve os tipos de objetos e classes do sistema e as várias formas de relacionamento estático existentes entre eles. Sua análise criteriosa pode detectar problemas de modelagem e possíveis soluções de menor custo operacional (FOWLER, 2005). A análise do diagrama de classes (figura 7.5) permite observar os detalhes utilizados na modelagem do Fids.

As nove classes apresentadas no diagrama 7.5 foram criadas de modo a facilitar a implementação deste IDS, além de propiciar o desenvolvimento independente de cada uma. As classes geradas possuem as funções abaixo apresentadas:

- *Principal:* é a classe por onde se inicia todos os estados demonstrados no diagrama de estados. Ela controla e instancia os objetos dentro do IDS.
- *Oid:* armazena o resultado de cada objeto buscado da MIB RMON2 de um dado agente.

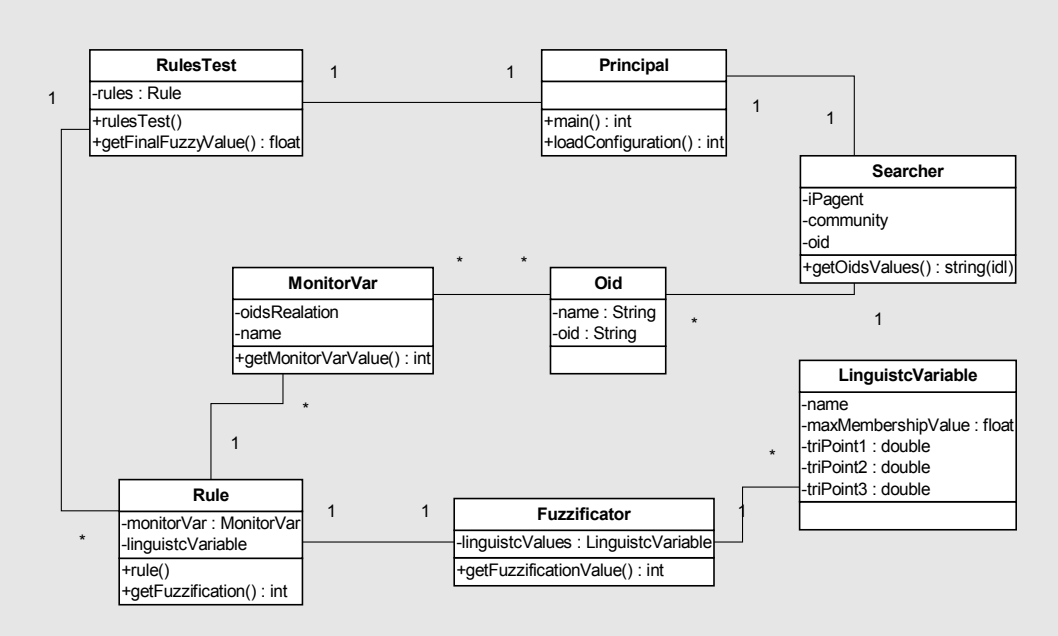

Figura 7.5: Diagrama de classes do IDS

- *Searcher:* classe que faz a busca pelos objetos da MIB RMON2 nos diferentes agentes configurados no sistema.
- *MonitorVar:* responsável por aplicar as operações de conjunção nas premissas de cada regra.
- *RulesTest:* dá início ao processo de teste de todas as regras. Ela as reconhece como agrupamentos de objetos Rule.
- *Rule:* é responsável por buscar em *MonitorVar* os valores resultantes das avaliações de cada premissa que compõe uma regra.
- *Fuzzificator:* desempenha papel importante na *fuzzificação* dos valores de entrada do IDS. Essa classe, quando indagada pela *Rule* consulta as *LinguistcVariables* para saber qual resultado atribuir à variável lingüística recebida.
- *LinguistcVariable:* armazena o conteúdo de cada uma das variáveis lingüísticas informadas ao IDS pelo configurador.

Segundo Fowler (2005), uma eficiente análise da relação existente entre objetos, assim como da forma como eles interligam-se para a tomada das ações desempenhadas em um sistema, pode ser obtida pela elaboração de diagramas de colaboração. A observação das relações existentes entre os diferentes objetos permite que ocorram correções na modelagem, tornando menos custosa a implementação do software (FOWLER, 2005). O diagrama de colaboração para o Fids é apresentado na figura 7.6.

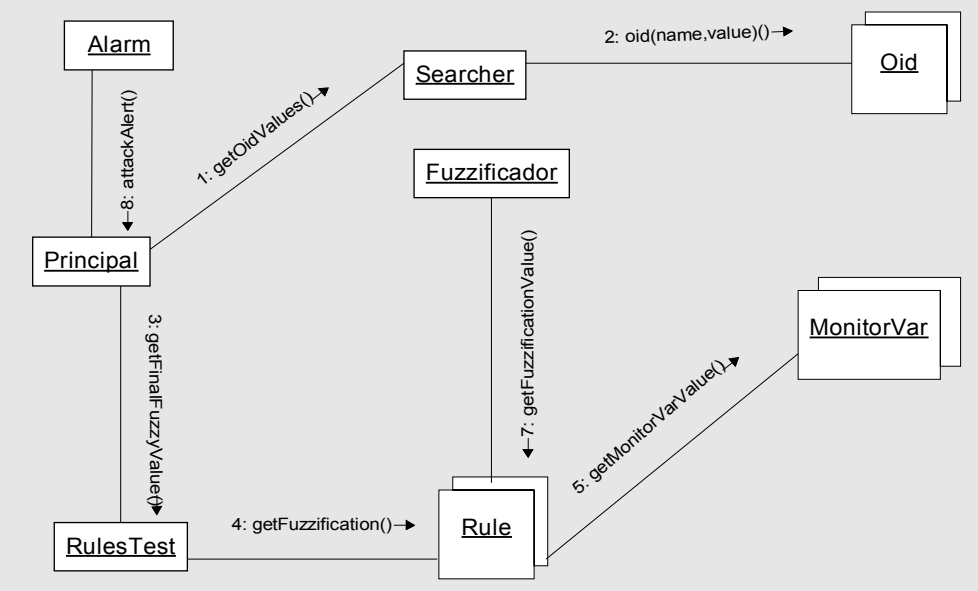

Figura 7.6: Diagrama de colaboração do IDS

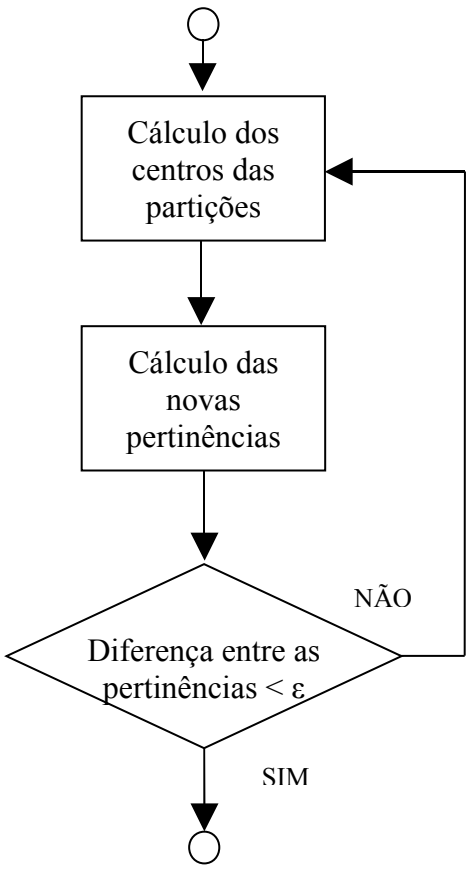

Figura 7.7: Fluxograma do Clusterizador

Pela análise da figura 7.6, observa-se a classe Principal informando ao *Searcher* de que é necessário fazer uma nova busca pelos valores dos *Oids*. De posse desses valores a *Principal* incumbe à classe *rulesTest* a tarefa de avaliar as diferentes regras configuradas no sistema. Cada regra é uma instância de *Rule*. Essa instância tem a

capacidade de, consultando *MonitorVar* e *Fuzzificator* atribuir um determinado valor a regra que representa. Esse valor é lido pela *RulesTest* que analisa a avalia os resultados da etapa de *defuzificação.*

1.25 Modelagem do Clusterizador

A modelagem do Clusterizador seguiu basicamente os três passos apresentados no algoritmo *fuzzy c-means* (capítulo 6). Os parâmetros de entrada desse elemento do IDS provêm do Configurador. Este, com uma interface amigável, recebe as definições criadas pelo administrador do sistema e as repassa ao Clusterizador. A ordem de iniciar o processo descrito pelo algoritmo de geração dos conjuntos difusos (figura 7.7) também é dada pelo Configurador.

TESTES

Os testes para o sistema detector de intrusão deste trabalho foram executados nas dependências do Ponto de Presença da Rede Nacional de Pesquisa no Rio Grande do Sul (POP-RS). Foi monitorada uma rede de um cliente deste ponto de presença. A estrutura utilizada para os testes apresentados neste capítulo é exposta na figura 8.1.

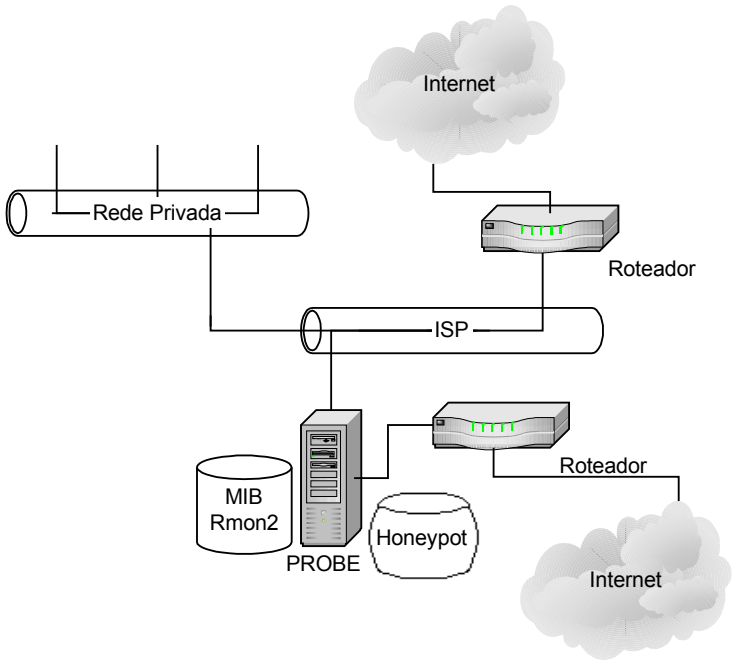

Figura 8.1: Estrutura de testes

Na estrutura apresentada na figura 8.1, houve cuidado em separar a rede utilizada para as consultas SNMP e a rede supervisionada. Dessa forma, o tráfego necessário às monitorações não influenciou nos dados dos objetos RMON2 capturados durante a operação deste IDS.

Nestes testes, o intervalo de *pooling* foi configurado para 1 minuto. Assim como exposto no capítulo 8, um intervalo menor, embora traga maior consumo de banda de rede destinado a operação do IDS, fornece maior precisão e velocidade nas decisões proferidas.

Os testes em uma rede em produção trouxeram a certeza de que o IDS proposto pode ser utilizado como um mecanismo de segurança. As análises aqui apresentadas foram feitas em concordância com a gerência de rede da instituição monitorada.

O capítulo 4 forneceu alguns argumentos sobre as vantagens de se utilizar *honeypots* integrados a sistemas detectores de intrusão. Embora nestes testes não tenha sido usada a ferramenta honeyd, os princípios dos *honeypots* foram empregados através do

monitoramento de portas com serviços não disponibilizados. Ainda que a ferramenta Honeyd traga maiores recursos na criação e configuração dos *honeypots*, ela exige a alocação de alguns endereços IP da rede monitorada, o que, para os testes aqui apresentados, exigiria maior envolvimento da equipe de gerência da rede em operação.

A figura 8.2 demonstra o volume de tráfego observado pelo IDS durante o período de testes (uma semana). O tráfego desta instituição tem média de pouco mais de 50 Kbit/s destinados à Internet e pouco menos de 300 Kbit/s destinados à própria rede. Seu volume de tráfego diminui consideravelmente durante os finais de semana, uma vez que, grande parte dos usuários não desempenha suas funções durante esse período.

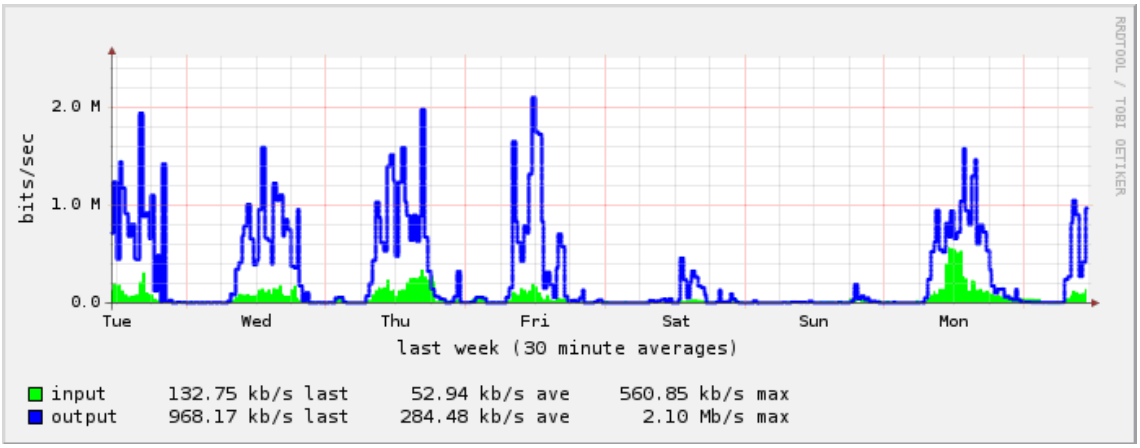

Figura 8.2: Tráfego da instituição monitorada

A estratégia de testes foi feita pela definição das variáveis lingüísticas e das regras para atuação deste IDS. Como forma de verificar o comportamento e a eficácia do sistema em questão, tomando o cuidado para não comprometer os serviços disponibilizados, simulações de ataques foram disparadas contra a rede e os servidores monitorados. Assim, os testes feitos nesta rede, demonstram de forma apropriada, os resultados deste IDS diante de situações semelhantes.

A construção das regras para a alimentação da base de conhecimentos deste IDS pode ser feita seguindo diversos métodos. Quanto à abrangência do tráfego monitorado, para os testes aqui demonstrados, foram analisadas duas abordagens: análise do tipo de tráfego total passante na rede e análise do tráfego de entrada e saída em um determinado computador.

Como amostra para testes, seguindo a abordagem da verificação do tráfego total monitorado, analisou-se o volume deste tráfego relativo ao protocolo SMTP encontrado pelo *probe* RMON2. A análise deste tipo de dado pode demonstrar atividade de envio de spam com origem dos computadores presentes dentro da própria instituição ou como tentativas provindas da Internet que intentam encontrar computadores vulneráveis ao repasse de e-mails (*open proxy*). Como visto no primeiro capítulo deste trabalho, o envio de spams está também relacionado à operação de *malwares* que tentam se replicar a outros computadores na Internet ou a sistemas localizados na própria rede privada. Essa atividade pode ainda estar relacionada às ações de fraude na Internet conhecidas por *fishing* onde os infratores enviam e-mails solicitando que usuários façam recadastramentos através do fornecimento de senhas e contas em instituições financeiras.

Os dados do tráfego obtidos através do monitoramento do número de pacotes SMTP foram aplicados ao módulo Clusterizador na intenção de obter 3 conjuntos difusos

distintos. Conforme visto no capítulo 6, a aplicação do algoritmo *fuzzy* c-means faz as partições geradas serem classificadas de acordo com suas distâncias relativas aos centros calculados pelo algoritmo. Nos dados relativos ao tráfego SMTP da rede supervisionada, o algoritmo determinou as pertinências de cada ponto obtido pelo monitoramento dos objetos RMON2. As figuras 8.3 e 8.4 apresentam o resultado da aplicação do Clusterizador sobre esses dados.

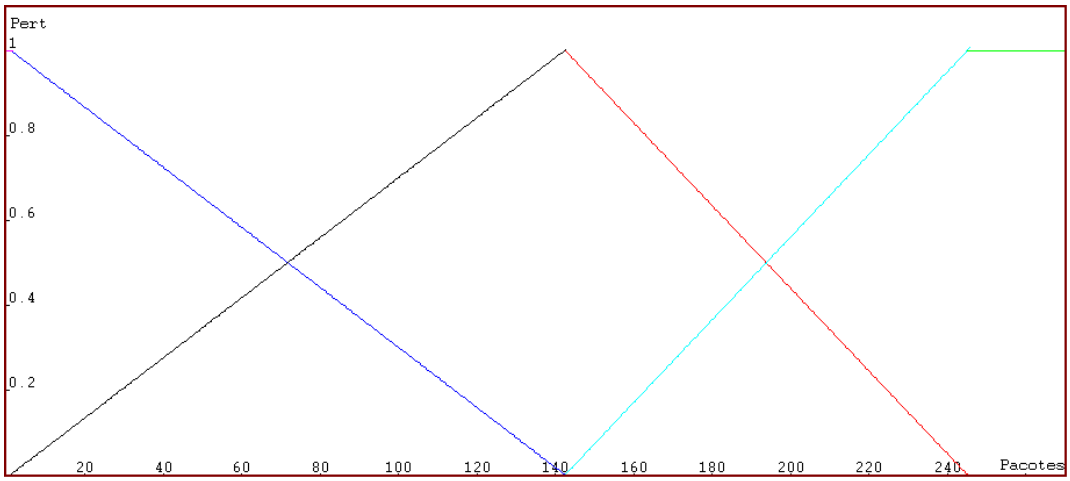

Figura 8.3: Conjuntos difusos do tráfego SMTP na rede monitorada (pacotes)

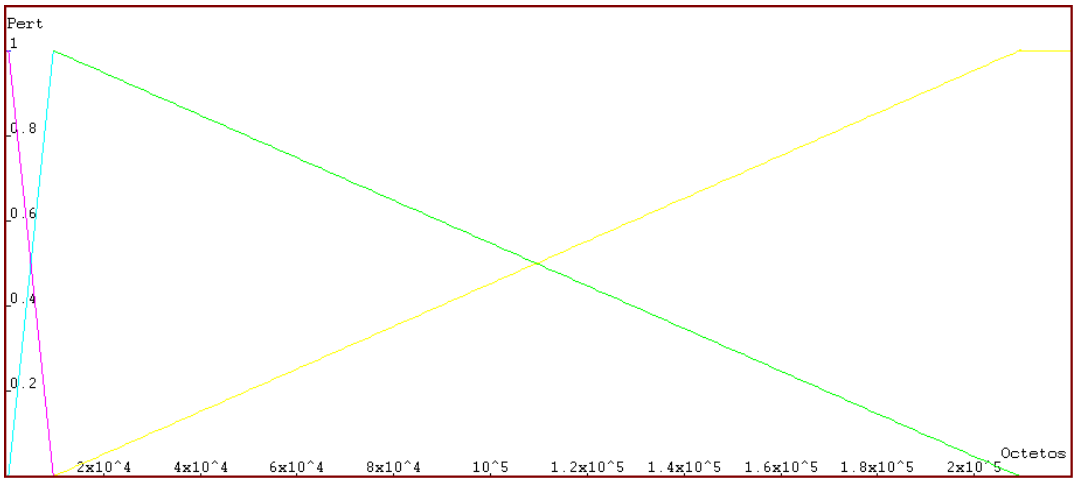

Figura 8.4: Conjuntos difusos do tráfego SMTP na rede monitorada (octetos)

O clusterizador determina os melhores centros das partições geradas bem como as pertinências associadas a cada objeto capturado. A cargo do adminsitrador de segurança, o clusterizador pode ser configurado para gerar um maior número de partições. Poder-se-ia, por exemplo, criar os conjuntos baixo, médio-baixo, médio, médio-alto e alto, por exemplo.

A geração das regras para a atuação da máquina de inferência descrita no capítulo 6 pode ser realizada pela determinação apenas dos comportamentos tidos como anômalos ou pela criação de regras também para outros comportamentos, como médios e normais, por exemplo. Em ambas as abordagens o método de Mamdani é aplicado para a avaliação das regras na etapa de inferência. Os conectores *e* entre as regras resultam no mínimo valor adquirido na *fuzzificação* de cada premissa. Os conectores *ou* são implicitamente associados às interligações entre as regras. Assim sendo, caso seja utilizada a abordagem em que apenas são criadas as regras delimitadoras do comportamento anormal, a etapa de *defuzzificação* que utiliza o algoritmo dos primeiros

máximos, traduzirá a saída de maior valor entre as avaliações das regras criadas. Caso for escolhida a abordagem que cria regras informativas também para outros comportamentos, é o método do centro de área que deve ser utilizado pelos Fids para a etapa de *defuzzyficação*, pois nele há um maior envolvimento das respostas obtidas nas avaliações das regras sintomáticas dos diferentes comportamentos modelados.

As regras geradas para a determinação de alguma atividade ilegal relacionada ao tráfego SMTP estão abaixo apresentadas. Cabe ressaltar que, dependendo da experiência e o conhecimento das peculiaridades de cada rede monitorada, os administradores de segurança podem optar por confeccionarem diferentes regras que possam melhor delimitar o comportamento dos seus ambientes supervisionados.

Se num_pacotes_smtp é alto e num_octetos_smtp é alto então alerta é alto (1)

Se num_pacotes_smtp é alto e num_octetos_smtp é baixo então alerta é alto (2)

As regra *1* foi criada com objetivo de identificar ataques de DoS contra o serviço de e-mails da instituição. A regra *2* tem a intenção de identificar ataques de varreduras, feitas através de pequenos pacotes, direcionadas ao serviço SMTP de computadores dentro e fora da rede corporativa.

Para o teste destas duas regras foi constituído um script em um servidor Linux que abria e fechava 8 conexões por segundo para a porta 25 TCP em um servidor de e-mail da instituição monitorada. Essa simulação de ataque foi feita durante a madrugada para evitar que pudesse ocasionar alguma dificuldade à rede em produção. O script permaneceu operando por 30 segundos. O Fids, com as regras (1) e (2) aplicadas às variáveis lingüísticas dos gráficos 8.3 e 8.4, resultou em um valor de 0,*84*. Esse resultado foi obtido em menos de 2 segundos, já adicionado os tempos de busca dos objetos. Como o sistema foi configurado para alertar o administrador em todo o ciclo que a resposta da máquina de inferência fosse superior a *0.75*, neste caso o sistema conseguiu distinguir a atividade ilegal.

O resultado de 0,84 veio da obtenção do valor 2701 para o objeto *num_pacotes_smtp* e 186775 para *num_octetos_smtp.* A avaliação da primeira premissa da regra *1* resultou no valor 1 e, para a segunda premissa*,* 0,84. Já a avaliação da regra *2* obteve o valor 0 pois a variável *num_octetos_smtp* não se enquadra do conjunto baixo. Assim, o processo de *defuzificação* do método de Mamdani de inferência resultou no valor de 0,84 (figura 8.5).

Para elucidar o exemplo da abordagem da análise do tráfego envolvido na comunicação de um determinado computador, foram monitorados os pacotes HTTP em octetos e em número de pacotes destinados a um servidor *web* presente na rede monitorada. Os dois gráficos resultantes da aplicação do método *c-means* sobre os dados capturados estão apresentados nas figuras 8.6 e 8.7.

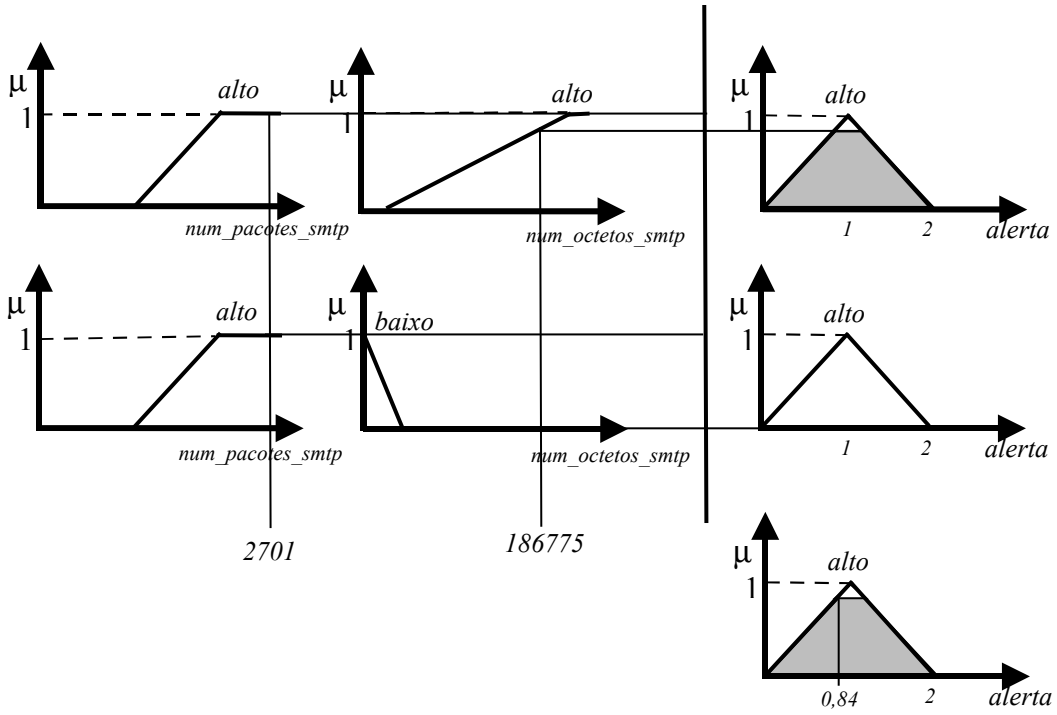

Figura 8.5: Processo de inferência aplicado às duas regras para o tráfego SMTP.

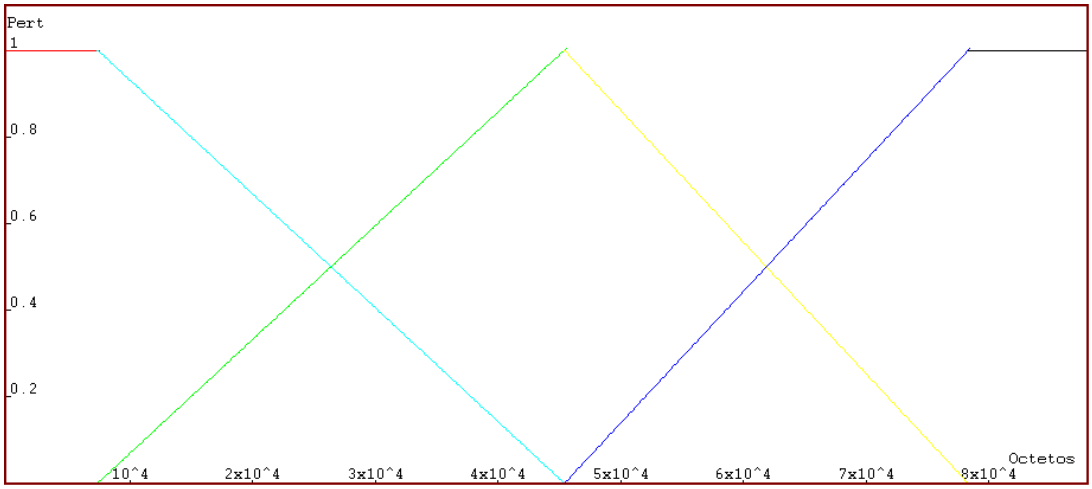

Figura 8.6: Conjuntos difusos para os octetos http destinados ao servidor.

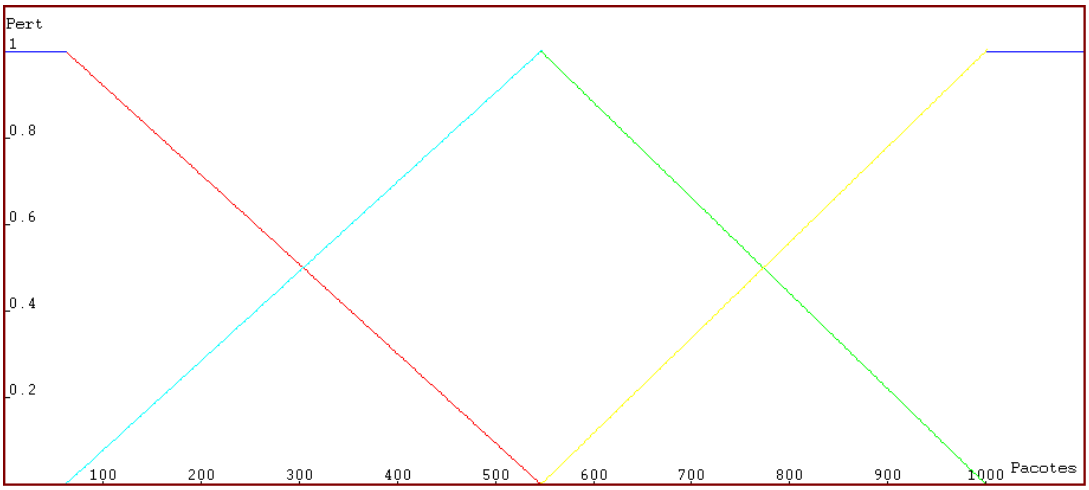

Figura 8.7: Conjuntos difusos para os pacotes http destinados ao servidor.

Nos testes sobre o servidor web monitorado, foram utilizados os conceitos dos *honeypots*. Para isso, todos os pacotes originados dentro da própria rede supervisionada e destinados a alguma porta sem serviço disponível no servidor monitorado seriam contados e aplicados sobre o gráfico construído manualmente no configurador (figura 8.8).

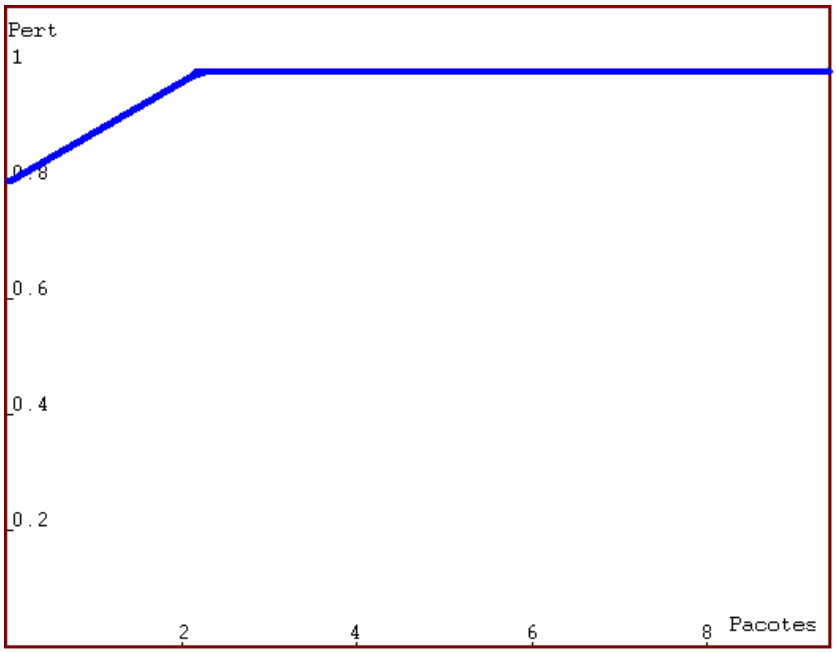

Figura 8.8: Conjunto difuso alto para os pacotes destinados a serviços não disponíveis no servidor monitorado.

De acordo com as definições expostas no capítulo 4 todos os pacotes destinados aos *honeypots* são maliciosos. O conjunto *fuzzy* alto apresentado na figura 8.8, dependendo do quanto o administrador de segurança irá tolerar esses acessos ilegais, pode ter um comportamento constante em 1 ou ser uma reta ascendente em direção a essa constante. Dessa forma, toda a tentativa de conexão a portas não disponíveis nesse servidor gerará um alerta de anormalidade. Essa abordagem tem sua importância ressaltada no fato de, mesmo sem consumir recursos de processamento e banda no tal servidor web, faz o mesmo ter função de honeypot. Embora a localização do probe RMON2 situado fora do ambiente da *firewall* privada da instituição monitorada impeça que pacotes originados e

destinados a computadores internos à rede privada sejam monitorados, a geração e implementação dessa regra envolvendo conceitos de *honeypots* tem a intenção de demonstrar a utilização desses mecanismos no contexto do Fids.

As regras dedicadas à verificação dos indicativos de anormalidades sintomáticas de atividade maliciosa na rede monitorada estão abaixo apresentadas.

Se num_pacotes_honeypot é alto então alerta é alto (1)

Se num_octetos_http_entrada é alto e num_pacotes_http_entrada é alto e então alerta é alto (2)

Se num_octetos_http_entrada é médio e num_pacotes_http_entrada é médio então alerta é médio (3)

Se num_octetos_http_entrada é baixo e num_pacotes_http_entrada é baixo então alerta é baixo (4)

A regra *1* traduz o comportamento da cooperação entre os *honeypots* e este IDS. Dependendo da tolerância do administrador de segurança, todos os acessos direcionados às portas não disponibilizadas no servidor web podem fazer o IDS disparar alarmes indicativos de anomalias.

A regra *2* foi criada para determinar ataques de DoS. Conforme apresentado no capítulo 7, a toda regra é associado um peso para a regulação do quanto ela influencia no resultado final do Fids. Inicialmente, se não houver nenhuma mudança a cargo do administrador de segurança, esses pesos podem ser desconsiderados uma vez que possuem o valor *1*. Se, por algum comportamento peculiar ao servidor monitorado, essa regra for responsável por um falso positivo, o administrador tem a possibilidade diminuir o peso a ela associado. No entanto, conforme visto no capítulo 6, a utilização da regra *2* se justifica pelo processo de Mamdani que sempre escolhe a premissa de menor valor.

As regras *3* e *4* unem à abordagem do monitoramento individual de computadores com à abordagem que também realiza a criação de regras demonstrativas dos comportamento aceitáveis. Neste tipo de metodologia há uma maior complexidade na criação das regras, pois, de forma diferente à apresentada na ilustração 8.5, onde o relacionamento entre regras embasava-se apenas na captura da regra com maior resposta, aqui, os resultados da etapa de *defuzzificação*, aplicando-se o método do centro de área, são influenciados também pelos valores provenientes das avaliações das regras indicativas dos comportamentos tolerados.

Os testes dessas regras foram feitos com a utilização da ferramenta *echoping* (ECHOPING, 2007), que possibilita criar verificações de conectividade utilizando o protocolo TCP. Com ela, foram disparados, durante a madrugada, 180 conexões ao servidor web monitorado, 3 a cada segundo. Mesmo com os tempos de busca dos OIDs o sistema levou menos de 2 segundos para chegar ao resultado apresentado na figura 8.9. O valor das variáveis lingüísticas obtidas pela leitura dos objetos SNMP estão abaixo apresentadas.

- *num_pacotes_http_honeypot=0.*
- *num_pacotes_http_entrando=909;*
- *num_octetos_http_entrando=79448;*

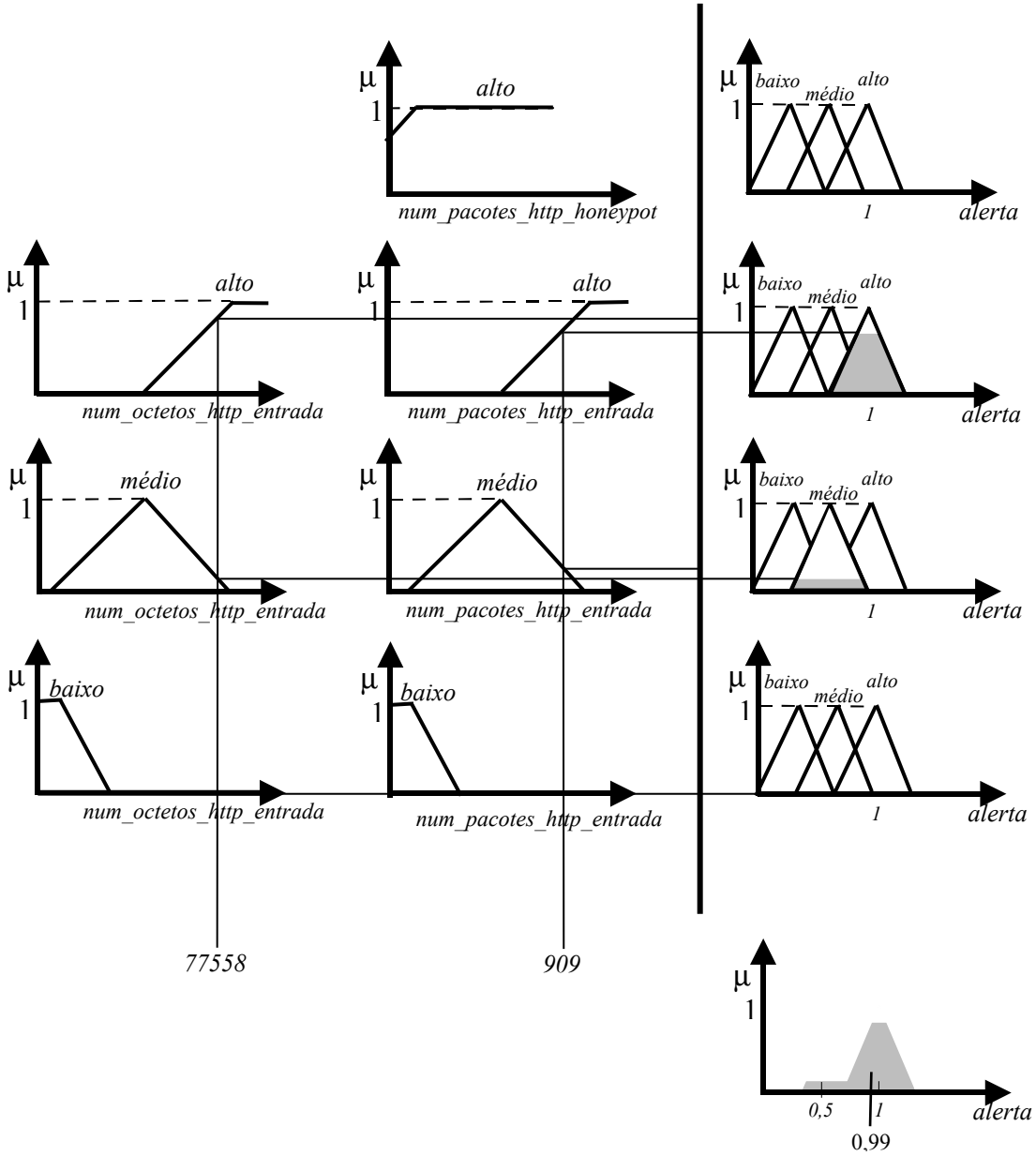

Figura 8.9: Processo de inferência aplicado às quatro regras para o monitoramento das atividades do servidor web.

Na figura 8.9 é demonstrada a aplicação do processo de inferência de Mamdani à abordagem da geração de regras para diferentes comportamentos associados aos estados dos sistemas monitorados. O Fids corretamente conseguiu detectar a anomalia provinda do ataque disparado contra o servidor web da instituição. No entanto, para este tipo de abordagem, o administrador de segurança deve tomar cuidado em definir regras delimitadoras da normalidade para todas as regras que modelem algum tipo de anormalidade. Se, utilizando o método de *defuzificação* do centro de área o administrador criar regras para o comportamento normal sem aquelas correspondentes ao anormal, o sistema poderá gerar falsos positivos à menor saída proveniente da avaliação da regra sintomática de anormalidade.

Há a possibilidade de serem criadas muitas regras distintas para analisar a atividade dos computadores presentes na rede supervisionada. Dependendo da característica de

cada rede, as regras devem determinar as atividades que podem ser presságios de incidentes de segurança em andamento.

Este capítulo de testes demonstrou duas possíveis utilizações do IDS aqui proposto. Dependendo do conhecimento da rede monitorada, os administradores de segurança podem criar centenas de regras para a melhora na eficácia do Fids. Quanto maior o tempo de coleta dos dados para a geração dos conjuntos difusos e quanto maior for a adequação das regras criadas, mais eficiente será o IDS aqui proposto.

CONCLUSÕES

A atual situação da segurança exige esforços integrados da comunidade científica. Este trabalho tratou da utilização da inteligência artificial associada à gerência de redes e à segurança de sistemas para a obtenção de um IDS eficaz. Este esforço multidisciplinar teve a finalidade de utilizar essas diferentes tecnologias em prol da melhora na segurança das redes onde for instalado o sistema detector de intrusão proposto.

A integração de ferramentas de gerência e de segurança tem a finalidade de concentrar esforços e aproveitar recursos muitas vezes já disponíveis nas redes privadas. Os agentes RMON2 podem ser facilmente instalados e configurados para que monitorem os segmentos de redes nos quais se deseja analisar o comportamento. Como muitas redes atualmente operam sobre ambientes comutados, há a possibilidade de fazer o espalhamento de portas nesses comutadores para que agentes RMON2 possam analisar todo o tráfego passante.

O IDS proposto por esse trabalho não teve o objetivo de atuar como único mecanismo de segurança da rede monitorada. A intenção foi obter um modelo de software que possa, junto com *firewalls* e honeypots, por exemplo, agregar esforços na tentativa de melhorar a segurança das redes privadas. A inclusão de *honeypots* na estrutura deste IDS permitiu obter todas as vantagens relacionadas à operação desse mecanismo. O capítulo 8 mostrou que além da possibilidade de se criar *honeypots* utilizando a ferramenta Honeyd, pode-se ainda, através do monitoramento dos pacotes destinados a portas sem serviço, usar seus conceitos para que todos os computadores monitorados tenham a capacidade de comportarem-se como um desses mecanismos sem ser preciso criar novos recursos ou aumentar capacidades de processamento. No que se refere aos *firewall*s, o Fids proposto nesse trabalho, tem melhor desempenho se aplicado a sistemas localizados em redes privadas protegidas por *firewall*s, uma vez que, conforme apresentado em trabalhos como Virti (2006-a), o volume de tráfego malicioso presente na Internet é surpreendente, e, portanto, monitorar o tráfego que já é barrado por outros sistemas de defesa pode representar desperdício de recursos. Além disso, o trabalho Virti (2006-b) verificou que, muitas vezes, quando sistemas de redes privadas são comprometidos com algum tipo de *malware* há uma grande chance desse software malicioso criar varreduras destinadas à computadores com endereços IP logicamente próximos àqueles atacados. Dessa forma, a utilização de *firewall*s associados a *honeypots* que, por sua vez, estão interligados a IDSs é bastante promissora.

O capítulo 6 descreveu as vantagens e os detalhes da utilização da inteligência artificial em sistemas detectores de intrusão. A lógica difusa foi empregada como forma de construir um sistema especialista que pudesse se adaptar às peculiaridades das redes supervisionadas. A utilização do algoritmo *fuzzy* c-means ajudou a tornar mais amigável a etapa de configuração do Fids. Esse algoritmo representa a memória do sistema

embasado no monitoramento de objetos RMON2 anteriormente capturados e armazenados sobre a forma de diferentes variáveis lingüísticas.

Embora a criação das regras ainda exija conhecimento da rede onde o IDS é instalado, o capítulo 8 mostrou que, usando as diferentes abordagens demonstradas na criação dessas regras, é possível configurar o Fids com certa facilidade. Os testes apresentados demonstraram a eficácia do sistema proposto na detecção de ataques como DoS e tentativas ilegais de acesso. Dependendo da habilidade do administrador de segurança que configurará o sistema, muitas outras anomalias sintomáticas de ataques podem ser detectadas.

Este trabalho não teve por objetivo esgotar todos os assuntos relacionados à utilização e integração das tecnologias aplicadas ao Fids. O que aqui buscou-se, no entanto, foi a elaboração de um modelo de integração entre ferramentas e protocolos de gerência de redes e de mecanismos de segurança para, com o auxílio da inteligência artificial, poder contribuir na melhora da segurança das redes onde este sistema detector de intrusão for instalado.

1.26 Trabalhos futuros.

Ao longo deste trabalho alguns pontos criaram temas que podem ser utilizados em trabalhos futuros. Um desses pontos é a possibilidade da geração automática de regras. Existem estudos que apontam para essa possibilidade (GOMEZ, 2002-a), muito embora ainda exijam que exemplos de anormalidade também sejam criados para o treinamento de redes neurais.

Outra possibilidade de trabalhos futuros é a implementação de um sistema que implemente regras predefinidas para determinados comportamentos de redes. Dependendo do perfil dessas redes, o administrador de segurança poderia escolher quais regras embasadas no RMON2 poderiam ser empregadas na tentativa de detectar atividades anômalas. É possível também desenvolver essas regras preconcebidas para determinados perfis de computadores. Assim, seria possível escolher, por exemplo, o perfil de regras direcionadas ao monitoramento de um servidor de e-mail ou de uma estação de trabalho.

A armazenagem da correspondência entre números MAC e endereços IP através do monitoramento do grupo *Address Maping* da MIB RMON2 permitiria a implementação de sistemas detectores de intrusão que pudessem coibir ataques relacionados ao forjamento de endereços IP. Embora, para isso o *probe* RMON2 necessitaria estar no mesmo segmento de rede dos computadores monitorados, os ganhos relativos às detecções de ataques como *mean in the meadle* e *teardrop* (capítulo 2) poderiam justificar os esforços. Se houvessem probes em todos os segmentos de rede, também a MIB RMON1 poderia ser utilizada para a verificação de anormalidades presentes nos níveis inferiores ao nível de rede (3 no modelo OSI).

REFERÊNCIAS

ARGAEZ, E. **Internet World STATS**, Bogota, 2002. Disponível em: <http://www.internetworldstats.com/stats.htm>. Acesso em: fev. 2007.

ASTITHAS, P.; KOUTEPAS, G.; MAGLARIS, B. Integrating Intrusion Detection and Network Management. In: NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM, NOMS, 2002. **Proceedings…** Washington: IEEE, 2002. p. 329-344.

ATAQUE hacker poderoso deixa Internet mais lenta. **Terra Tecnologia**, São Paulo. Disponível em: <http://tecnologia.terra.com.br/interna/0,,OI1397422-EI4805,00 .html> Acesso em fev. 2007.

AXELSSON, S. **Intrusion Detection System:** A Survey and Taxionomy. Department of Computer Engineering, Chalmers University of Technology, Göteborg, 2000. Technical Report.

BACE, R.; MELL, P. **Intrusion Detection Systems**. U.S.A: National Institute of Standards and Tecnology, Computer Division Security, 2001. Disponível em: <http://csrc.nist.gov/publications/nistpubs/>. Acesso em jul. de 2006.

BALASUBRAMANIYAN, J. S. et al. An Architecture for Intrusion Detection Using Autonomous Agents. In: ANNUAL COMPUTER SECURITY APPLICATIONS CONFERENCE, ACSAC, 14., 1998. **Proceedings…** Washington: IEEE, 1998. p. 13-23.

BEALE, J.; FOSTER, J. **Snort 2:** Sistema de Detecção de Intruso**.** Rio de Janeiro: Alta Books, 2003.

BENSO, A. C. de S. **Sistema de Detecção de Intrusão baseado em Métodos Estatísticos para Análise de Comportamento**. 2003. 121 f. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

BERNARDES, M. C.; MOREIRA E. S. Implementation of an Intrusion Detection System based on Mobile Agents. In: INTERNATIONAL SYMPOSIUM ON SOFTWARE ENGINEERING FOR PARALLEL AND DISTRIBUTED SYSTEMS, PDSE, 2000. **Proceedings…** Washington: IEEE, 2000. p. 158-168.

BERNARDES, M. C.; MOREIRA, E. S. Implementation of an Intrusion Detection System Base on Mobile Agents. In: INTERNATIONAL SYMPOSIUM ON SOFTWARE ENGINEERING FOR PARALLEL AND DISTRIBUTED SYSTEMS, 2000. **Proceedings…** Washington: IEEE, 2000. p. 158-165.

BRIDGES, S. M. et al. Intrusion Detection via Fuzzy Data Mining. In: ANNUAL CANADIAN INFORMATION TECNOLOGY SECURITY SYMPOSIUM, 12., 2000, Ottawa. **Proceedings…** Ottawa: [s.n.], 2000.

CABREIRA, J. D. B. et al. Proactive Intrusion Detection and SNMP-based Security Management: New Experiments and Validation. In: IFIP/IEEE INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT, 2003. **Proceedings…** Washington: IEEE, 2003. p. 93-96.

CAIS: Centro de Atendimento a Incidentes de Segurança. Rede Nacional de Pesquisa, Rio de Janeiro. Disponível em: <http://www.rnp.br/cais>. Acesso em: fev. 2007.

CARTER, L. W. Setting Up a Honeypot Using a Bait and Switch Router. In: SANS SECURITY ESSENTIALS AND THE CISSP DOMAINS, 10., 2005, Orlando. **Proceedings…** [S.l.]: SANS, 2004.

CASE, J. et al. **A Simple Network Protocol (SNMP)**: RFC 1157. [S.l.]: Internet Engineering Task Force, Network Working Group, 1990.

CASTELLS, M. **The Internet Galaxy:** Reflections on The Internet, Business and Society. Oxford: Oxford University Press, 2003

CERTBR: Centro de Estudos, Resposta e Treinamento e Tratamento de Incidentes de Segurança no Brasil, Registro br, São Paulo. Disponível em: <http://www.cert.br>. Acesso em: fev. 2007.

CONSORCIO Brasileiro de Honeypots, Projeto Honeypots Distribuídos. CERTBR E CENPRA, São Paulo. Disponível em: <http://www.honeypots-alliance.org.br/>. Acesso em jun. 2005.

CORCHADO, E. et al. Intrusion Detection System Based on a Cooperative Topology Preserving Method. In: ADAPTATIVE AND NATURAL COMPUTING ALGORITHMS, Viena, 2005. **Proceedings...** [S.l.:s.n.], 2005. p. 454-457.

CORCHADO, E.; HERRERO, A.; SAIZ, J. M. Detecting Compounded Anomalous SNMP Situations Using Cooperative Usupervised Pattern Recognition. In: ARTIFICIAL NEURAL NETWORKS: FORMAL MODELS AND THEIR APPLICATIONS, ICANN, 15., 2005, Warsaw, Poland. **Proceedings...** Berlin: Springer, 2005. p. 905-910.

CORDON, O.; JESUS. M. J.; HERRERRA, F. A Proposal on Reasoning Methods in Fuzzy Rule-Based Classification Systems. **International Journal of Approximate Reasoning**, New York, v. 20, p. 21-45, 1999.

CRIMES virtuais devem movimentar até 8 bilhões de dólares em 2007. **Terra Tecnologia**, São Paulo. Disponível em: <http://tecnologia.terra.com.br/interna /0,,OI1319667-EI4805,00.html> Acesso em fev. 2007.

DERAISON, R. **Nessus Vulnerability Scanner**. Disponível em: <http://www. nessus.org>. Acesso em: maio 2007.

DICKERSON J. E. et al. Fuzzy Intrusion Detection. In: JOINT IFSA WORLD CONGRESS, 9.; INTERNATIONAL CONFERENCE OF THE NORTH AMERICAN

FUZZY INFORMATION PROCESSING SOCIETY, NAFIPS, 20., 2001. **Proceedings…** [S.l.]: IEEE, 2001. v. 3, p. 25-28.

DICKERSON, J. E.; DICKERSON, J. A. Fuzzy Network Profiling for Intrusion Detection. In: INTERNATIONAL CONFERENCE OF THE NORTH AMERICAN FUZZY INFORMATION PROCESSING SOCIETY, NAFIPS, 19., 2000, Atlanta. **Proceedings...** [S.l.]: IEEE, 2000. p. 301-306.

DINGLEDINE, R. **Tor**: Anonymity Online. Disponível em: <http://tor.eff.org>. Acesso em: fev. 2007.

DNSSEC, o que é e por que precisamos dele. Rede Nacional de Ensino e Pesquisa, São Paulo. Disponível em: <http://www.rnp.br/newsgen/9801/dnssec.html>. Acesso em: dez. 2006.

DSHIELD: Distributed Intrusion Detection System. Disponível em: <http://www.dshield.org>. Acesso em: jun. 2005.

ERICKSON J. **Hacking:** The Art of Exploitation. San Francisco: No Starch Press, 2003.

FERREIRA, F. **Segurança da Informação**. Rio de Janeiro: Ciência Moderna, 2003.

FOWLER, M. et al. **UML Essencial.** Rio de Janeiro: Bookman, 2005.

FRANCO, L. H.; BARATO, L. G.; MONTES, A. Instalação e Uso de Honeypot de Baixa Interatividade. In: REUNIÃO DO GRUPO DE TRABALHO DE ENGENHARIA DE REDES, GTER; 17., 2004, São Paulo. **Reunião**. São Paulo: GTER, 2004. Disponível em: <http://eng.registro.br/gter17/videos>. Acesso em: jun. 2006.

FRAUDE Virtual deu Prejuízo de 300 milhões em 2006, estima IPDI**. IDG Now,** São Paulo. Disponível em: <http://idgnow.uol.com.br/seguranca/2007/02/02/idgnoticia. 2007-02-02.8169384159/IDGNoticia_view>. Acesso em: fev. 2007.

GASPARY, L. P. A SNMP-based Platform for Distributed Stateful Intrusion Detection in Enterprise Networks. **IEEE Journal on Selected Areas in Communications,** New York, v. 23, n. 10, p. 1973-1982, Oct. 2005.

GASPARY, L. P. **Gerenciamento Distribuído e Flexível de Protocolos de Alto Nível.** 2002. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

GOMEZ, J. et al. Complete Expression Trees for Evolving Fuzzy Classifier Systems with Genetic Algorithms to Network Intrusion Detection. In: FUZZY INFORMATION PROCESSING SOCIETY, NAFIPS, 19., 2000, Atlanta. **Proceedings...** [S.l.]: IEEE, 2000. p. 469-474.

GOMEZ, J.; DASGUPTA, D. Evolving Fuzzy Classifiers for Intrusion Detection. In: WORKSHOP ON INFORMATION ASSURANCE, 2001, New York. **Proceedings…** [S.l.]: IEEE, 2002. p. 68-75.

HATCH, B.; LEE, J.; KURTZ, G. **Hackers Linux Expostos:** Segredos e Soluções para a Segurança do Linux. São Paulo: Makron Books, 2002.

HERRERO, A.; CORCHADO, E.; SAIZ, J. M. A Cooperative Connectionist IDS model to Identify Independent Anomalous SNMP Situations. In: TALLER NACIONAL DE MINERÍA DE DATOS Y APRENDIZAJE, TAMIDA, 3., 2005, Granada. **Proceedings...** Granada: Thonson, 2005.

HOLZ, T. **New Fields of Application for Honeypots**. 2005. 50 f. Thesis (Department od Computer Science Diploma Thesis) – Department of Computer Science Rwth Aachen, Aachen.

HOLZ, T.; RAYAL, F. Detecting Honeypots and Other Suspicious Environments. In: WORKSHOP ON INFORMATION ASSURANCE AND SECURITY, New York, 2005. **Preceedings…** [S.l.]: IEEE, 2005. p. 29-36

HONEYNET Research Alliance. Disponível em: <http://www.honeynet.org/alliance> Acesso em: jun. 2005.

HONEYNET. **Know Your Enemy**: Tracking Botnets, 2005. The Honeypots Project. Disponível em: <http://www.honeynet.org/papers/bots/>. Acesso em: jan. de 2007.

HOWARD, M.; LEBLANC, D. **Writing Secure Code.** Washington: Microsoft Press, 2003.

IDRIS, N. B.; SHANMUGAM, B. Artificial Intelligence Techniques Applied to Intrusion Detection. In: ANNUAL IEEE INDICON, 2005, Chennai. **Proceedings…** [S.l.]: IEEE, 2005. p. 52-55.

IDRIS, N. B.; SHANMUGAM, B. Novel Attack Detection Using Fuzzy Logic and Data Mining. In: INTERNATIONAL CONFERENCE ON SECURITY & MANAGEMENT, SAM, 2006, Nevada. **Proceedings…** [S.l.]: CSREA Press, 2006. p. 26-31.

JANG, R.; GULLEY, N. **Matlab Fuzzy Logic Toolbox**. Natick: The MathWorks, 1995.

JANIKOW, C. J. Fuzzy Decision Trees: Issues and Methods. **IEEE Transactions on Systems, Man, and Cybernetcs, Part B**, Washington, v. 28, n. 1, p. 1-14, Feb. 1998.

KABIRI, P.; GHORBANI, A. A. Research on Intrusion Detection and Response: A Survey. **International Journal of Network Security**, [S.l.], p. 84-102, June 2005.

KREIBICH, C.; CROWCROFT, J. Honeycomb: Creating Intrusion Detection Signatures Using Honeypots. **ACM SIGCOMM Computer Communication Review,** New York, p. 51-56, Jan. 2004.

KUWATLY, I. et al. Dynamic Honeypot Design for Intrusion Detection. In: INTERNATIONAL CONFERENCE ON PERVASIVE SERVICES, ICPS, 2004. **Proceedings…** Washington: IEEE, 2004. p. 95-104.

LEE, K. C.; MIKHAILOV, L. Intelligent Intrusion detection System. In: IEEE INTERNATIONAL CONFERENCE ON INTELLIGENT SYSTEMS, 2004. **Proceedings…** Washington: IEEE, 2004. v.2, p. 497-502.

LOPES, R.; SAUVÊ, J.; NICOLLETTI, P. **Melhores Práticas para a Gerência de Redes de Computadores.** Rio de Janeiro: Campus, 2003.

MANIC, M.; WILAMOWSKI, B. Fuzzy Preference Approach for Computer Network Attack Detection. INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, 2001. **Proceedings…** Washington: IEEE, 2001. v.2, p. 1345-1349.

MAURO, D.; SCHMIDT, K. **SNMP Essencial.** Rio de Janeiro: Campus, 2001.

MCCLURE, S.; SCAMBRAY, J.; KURTZ, G. **Hackers Expostos:** Segredos e Soluções para a Segurança de Redes. São Paulo: Makron Books, 2000.

MCNEILL, D.; FREIBERGER, P. **Fuzzy Logic.** New York: Touchstone, 1993.

MENEGHETTI, E. A. **Uma Proposta de Uso da Arquitetura Trace como um Sistema de Detecção de Intrusão**. 2002. 105 f. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

MILL, J.; INOUE, A. Support Vector Classifiers and Network Intrusion Detection. In: IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, Budapest, 2004. **Proceedings…** Washington: IEEE, 2004. p. 25-29.

ORFILA, A; CARBO J.; RIBAGORDA, A. Fuzzy Logic on Decision Model for IDS. THE IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, 2003, Madrid. **Proceedings…** Washington: IEEE, 2003. p. 1237-1242.

PEIKARI, C.; CHUVAKIN, A. **Security Warrior.** Beijing: O'Reilly, 2004.

PREATONI, R. **Zone-h**: The Internet Thermometer. Disponível em: <http://www.zone-h.org>. Acesso em: dez. 2006.

PROVOST, F.; FAWCETT, T. Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, KDD, 1997, Huntington Beach. **Proceedings…** Menlo Park, CA: American Association for Artificial Inteligence, 1997. p. 43-48.

PROVOST, F.; FAWEETT, T.; KOHAVI, R. The Case Against Accuracy Estimation for Comparing Induction Algorithms. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 16., 2006. **Proceedings…** San Francisco: Morgan Kaufmann, 2006. p. 445-453.

QIN, X. et al. Integrating Intrusion Detection and Network Management. In: NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM, NOMS, 2002. **Proceedings…** Washington: IEEE, 2002. p 329-344.

RAZZARI, M. R.; BATHAEIAN, N. Multi Agents in Mid Involvement Deception System. In: INTEGRATION OF KNOWLEDGE INTENSIVE MULTI-AGENT SYSTEMS, Boston, 2003. **Proceedings…** Washington: IEEE, 2003. p 88-93.

REGISTRO.BR, Registro de Domínios para a Internet no Brasil, São Paulo. Disponível em: <http://registro.br/estatisticas.html>. Acesso em: fev. 2007.

ROSE, M. **The Simple Book:** An Introduction to Management of TCP/IP-based Internets. New Jerssey: Prentice-Hall,1991.

RUSSEL, D.; GAMGEMI, G. T. **Computer Security Basics.** Sebastopol: O'Reilly & Associates, 1991.

SAHAI, A.; WATERS, B. Fuzzy Identity-Based Encryption. In: ADVANCES CRYPTOLOGY, EUROCRYPT, 2005. **Proceedings…** Berlin: Springer, 2005. p. 457-473.

SANDRI, S.; CORREA, C. Lógica Nebulosa. In: ESCOLA DE REDES NEURAIS, 5., 1999, São José dos Campos. **Anais...** São José dos Campos: Conselho Nacional de Redes Neurais: ITA, 1999.

SCHNEIER, B. **Segurança .Com. Segredos e Mentiras sobre a Proteção na Vida Digital.** Rio de Janeiro: Campus, 2000.

SHAH, H.; UNDERCOFFER, J.; JOSHI, A. Fuzzy Clustering for Intrusion Detection. In: IEEE INTERNATIONAL CONFERENCE ON FUZZY SYSTEMS, 12., 2003. **Proceedings…** Washington: IEEE, 2003. v. 2, p. 1274-1278.

SHERIF, J.S.; DEARMOND, T.G. Intrusion detection: system and models. In: IEEE INTERNATIONAL WORKSHOPS ON ENABLING TECNOLOGIES: INFRASTRUCTURE FOR COLLABORATIVE ENTERPRISES, WETICE, 11., 2002. **Proceedings…** Washington: IEEE, 2002. p. 115-133.

SILVEIRA, E. R.; DANTAS, M. A. R. Uma Abordagem de Monitoração de Tráfego de Rede Utilizando Lógica Difusa. **Infocomp: The Journal of Computer Science**, Lavras, v. 4, n. 3, p. 32-41, Sept. 2005.

SINGARAJU, G.; TEO, L.; ZHENG, Y. A Testbed for Quatitative Assessment of Intrusion Detection Systems using Fuzzy Logic. In: IEEE INTERNATIONAL INFORMATION ASSURANCE WORKSHOP, IWIA, 2004, Charlotte. **Proceedings…** [S.l.]: IEEE, 2004. p. 79-93.

SPITZNER, L. **Honeypots:** Tracking Hackers. [S.l.]: Addilson-Wesley Professional, 2002.

STALLINGS, W. **SNMP, SNMPV2 and RMON, Practical Network Management**. Massachussetts: Addison-Wesley,1996.

STANGER, J.; LANE, P. T.; DANIELYAN, E. **Rede Segura Linux.** Rio de Janeiro: Alta Books, 2002.

STOLL, C. **The Cuckoo's Egg.** New York: Pocket Books, 1990.

TANAKA, K. **An Introduction to Fuzzy Logic for Practical Applications**. New York: Springer, 1997.

TAROUCO, L. M. R. **Inteligência Artificial Aplicada ao Gerenciamento de Redes de Computadores**. 1990. 188 f. Tese (Doutorado em Engenharia) – Departamento de Engenharia, USP, São Paulo.

TILLAPART, P., THUMTHAWATWORM, T.; SANTIPRABHOB, P. Fuzzy Intrusion Detection System. **AU Journal of Tecnology**, [S.l.], p. 109-115, Oct. 2002.

VIECCO, C. Sebek Homepage. **Honeynet Project**. Disponível em <http://www.honeynet.org>. Acesso em dezembro de 2007.

VIRTI, E. S. **Diagnóstico das Tentativas de Ataque na Atualidade**. 2005. 30 f. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

VIRTI, E. S.; BERTHOLDO, L. M.; TAROUCO, L. M. R.; GRANVILLE, L. Utilizando Honeypots para a Medição de Atividade de Rede Não Usual na Internet. In: SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES, SBRC, 24., 2006, Curitiba. **Anais...** Curitiba: [s.n.], 2006. p. 377-389.

VIRTI, E. S.; TAROUCO, L. M. R.; BERTHOLDO, L. M.; CERON, J. M.; GRANVILLE, L. Honeypots as a Security Mechanism. In: IEEE WORKSHOP ON MONITORING, ATTACK DETECTION AND MITIGATION, MonAm, 1., 2006, Tuebingen. **Proceedings…** Tüebingen: IEEE, 2006. p. 55-58.

WALDBUSSER, S. **Remote Network Monitoring Management Information Base Version 2.**: RFC 2021. [S.l.]: Internet Engineering Task Force, Network Working Group, 1997.

WALDBUSSER, S. **Remote Network Monitoring Management Information Base:** RFC 1757. [S.l.]: Internet Engineering Task Force, Network Working Group, 1995.

ZADEH, L. A. Fuzzy Sets. **Information and Control**, [S.l.], p. 338-353, 1965.

ZHANG, X. et al. Intrusion Prevention System Design. COMPUTER AND INFORMATION TECHNOLOGY, 4., 2004. **Proceedings...** Washington: IEEE, 2004. p. 386-390.

ZHANG, X.; LI, C.; HU, Q. The Network Management Design Integrated With The Intrusion Detection System. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AN CYBERNETICS, 2004, Shanghai. **Proceedings...** Washington: IEEE, 2004. p. 257-262.