UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE ELETRICIDADE

RAIMUNDO PEREIRA DA CUNHA NETO

Sistema de Detecção de Intrusos em Ataques Oriundos de *Botnets* Utilizando Método de Detecção Híbrido

São Luís
2011

RAIMUNDO PEREIRA DA CUNHA NETO

SISTEMA DE DETECÇÃO DE INTRUSOS EM ATAQUES ORIUNDOS DE *BOTNETS* UTILIZANDO MÉTODO DE DETECÇÃO HÍBRIDO

Dissertação de Mestrado submetida à Coordenação do Programa de Pós-Graduação em Engenharia de Eletricidade da UFMA como parte dos requisitos para obtenção do título de Mestre em Engenharia de Eletricidade, área Ciência da Computação.

Orientador: Prof. Dr. Zair Abdelouahab

São Luís

2011

RAIMUNDO PEREIRA DA CUNHA NETO

SISTEMA DE DETECÇÃO DE INTRUSOS EM ATAQUES ORIUNDOS DE *BOTNETS* UTILIZANDO MÉTODO DE DETECÇÃO HÍBRIDO

Dissertação de Mestrado submetida à Coordenação do Programa de Pós-Graduação em Engenharia de Eletricidade da UFMA como parte dos requisitos para obtenção do título de Mestre em Engenharia de Eletricidade, área Ciência da Computação.

Aprovada em ____ / ____ / 2011

_____

Prof. PhD. Zair Abdelouahab

Orientador

_____

Profa. Dra. Daniela Barreiro Claro

Membro da Banca Examinadora

_____

Prof. Dr. Francisco José da Silva e Silva

Membro da Banca Examinadora

_____

Prof. Dr. Denivaldo Cicero Pavão Lopes

Membro da Banca Examinadora

Cunha Neto, Raimundo Pereira da

Sistema de Detecção de Intrusos em Ataques oriundos de *Botnets* utilizando Método de Detecção Híbrido / Raimundo Pereira da Cunha Neto. – São Luís, 2011.

99 f.

Orientador: PhD. Zair Abdelouahab

Dissertação (Mestrado) – Programa de Pós-Graduação em Engenharia de Eletricidade, Universidade Federal do Maranhão.

1. Segurança de Redes. 2. *Botnets*. 3. IDS. 4. PSO. 5. Redes Neurais I. Título.

CDU XXX.XXX.XX

AGRADECIMENTOS

A Deus, ser maior e supremo, que me amparou nos momentos mais difíceis desta jornada.

À minha filha Nyédja Laysa, razão da minha vida, pelo carinho, amor e força de toda determinação para continuar este caminho.

Ao meu eterno mestre, meu pai, Walter Cunha e a minha Avó Raimunda, que partiram desta vida durante a construção deste projeto.

A CNPq pelo apoio e contribuição financeira dada durante todo o desenvolvimento do trabalho aqui apresentado.

Ao meu Orientador Prof. Dr. Zair Abdelouahab, pela imensa paciência, compreensão e dedicação dispensadas à realização deste trabalho.

À Profa. Dra. Maria da Guia, por não se esquecer de cobrar, a todo o momento, a conclusão deste trabalho.

Aos amigos do Mestrado Lianna, Frederico, Valéria, Eduardo, César, Vladimir, Marcondes, Amélia, Fernando Sérvulo, Ariel, Berto, Flávio e Bruno, pelos momentos de força.

A minha Mãe Janete Duarte pelo amor de me colocar no mundo para o aprendizado da vida.

Aos meus irmãos Daniela, Júnior, Germana e Vitor Manuel, aos meus sobrinhos Camille e Rodrigo, e as Minhas Tias Valquiria e Valmira.

Agradeço às minha família pelo apoio incondicional em todos os momentos difíceis que fizeram parte dessa jornada.

Aos meus amigos Mara Cristina, Conceição de Maria, Graça, Francisca, Kleber, Marcos Iran, Deborah, Dina, Marcus, João, Ronny, Waltemberg, Daniel Coelho e Isaac que estiveram sempre torcendo por esta conquista.

A todas as outras pessoas que contribuíram diretamente e indiretamente e que, apesar de não terem sido citadas, não foram e nem serão esquecidas.

Com Deus aprendi que com o bem possamos ir sempre além. Se cair, é hora de levantar e seguir. E nunca desistir. Foi assim, que hoje cheguei aqui.

RESUMO

A ampliação dos mecanismos de defesas no uso do combate de ataques ocasionou a evolução dos *malwares*, que se tornaram cada vez mais estruturados para o rompimento destas novas barreiras de segurança. Dentre os inúmeros *malwares*, a *Botnet* tornou-se uma grande ameaça cibernética, pela capacidade de controle e da potencialidade de ataques distribuídos e da estrutura de controle existente. A detecção e a prevenção de intrusão desempenham um papel cada vez mais importante na segurança de redes de computadores. Em um sistema de detecção de intrusão, as informações sobre a situação atual e os conhecimentos sobre os ataques tornam mais eficazes o processo de segurança diante desta nova ameaça cibernética. A solução proposta apresenta um modelo de Sistema de Detecção de Intrusos (IDS) que visa na ampliação de detectores de *Botnet* através da utilização de sistemas objetos ativos, propondo uma tecnologia de coleta por sensores, filtro de pré-processamento e detecção baseada em assinatura e anomalia, auxiliado pelo método de inteligência artificial Otimização de Enxame da Partícula (PSO) e Redes Neurais Artificiais.

Palavras-chave: Segurança de Redes, *Botnets*, IDS, PSO, Redes Neurais Artificiais

ABSTRACT

The defense mechanisms expansion for cyber-attacks combat led to the malware evolution, which have become more structured to break these new safety barriers. Among the numerous malware, Botnet has become the biggest cyber threat due to its ability of controlling, the potentiality of making distributed attacks and because of the existing structure of control. The intrusion detection and prevention has had an increasingly important role in network computer security. In an intrusion detection system, information about the current situation and knowledge about the attacks contribute to the effectiveness of security process against this new cyber threat. The proposed solution presents an Intrusion Detection System (IDS) model which aims to expand Botnet detectors through active objects system by proposing a technology with collect by sensors, preprocessing filter and detection based on signature and anomaly, supported by the artificial intelligence method Particle Swarm Optimization (PSO) and Artificial Neural Networks.

Keywords: Network Security, Botnet , IDS, PSO, Artificial Neural Networks

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SIGLAS

AP – *Access Point*

BDWIDS – *Wireless Intrusion Detection System DataBase*

CERT.br – Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil

C&C – Comando e Controle

CIDF – *Common Intrusion Detection Framework*

DB – *DataBase*

DDoS – *Distributed Denial of Service*

DEE – Departamento de Engenharia de Eletricidade

DFDB – *Standard of Intruders and Intrusion DataBase*

DNS – *Domain Name System*

DoS – *Denial of Service*

FTP – *File Transfer Protocol*

GT – *Global Threat*

HIDS – *Host Intrusion Detection System*

HTTP – *Hypertext Transfer Protocol*

IDS – *Intrusion Detection System*

IDS-H – *Intrusion Detection System Hybrid*

IIDB – *Incidents of Intrusion and Forensic Information DataBase*

IP – *Internet Protocol*

IRC – *Internet Relay Chat*

LAN – *Local Area Network*

LABSAC – Laboratório de Sistemas e Arquiteturas Computacionais

MAC – *Media Access Control*

MABDS – *Model of Multi-Agent Bots Detection System*

MD5 – *Message-Digest algorithm 5*

MITM – *Man in the Middle*

MLP – *MultiLayer Perceptron*

NIC – *Network Interface Card*

NIDIA – *Network Intrusion Detection System based on Intelligent Agents*

NIDS – *Network Intrusion Detection System*

P2P – *Peer to Peer*

PSO – *Particle Swarm Optimizer*

RADB – *Reaction DataBase*

RBF – *Radial Basis Function*

RNA – Redes Neurais Artificiais

SCA – *System Controller Agent*

SEA – *Security Evaluation Agent*

SHA-1 – *Secure Hash Algorithm*

SMA – *System Monitoring Agent*

SOM – *Self-Organizing Maps*

STDB – *Strategy DataBase*

SUA – *System Updating Agent*

UFMA – Universidade Federal do Maranhão

WLAN – *Wireless Local Area Network*

WIDS – *Wireless Intrusion Detection System*

SUMÁRIO

1 INTRODUÇÃO

A tecnologia das redes vem se tornando ao longo dos anos muito utilizadas em corporações, ambientes domésticos e redes públicas, como shopping, aeroportos e restaurantes, devido às inúmeras vantagens apresentadas pela sua utilização, proporcionando a humanidade o rompimento de barreiras, diminuição de distância, além da ampliação do conhecimento. A implantação generalizada de redes trouxe novos desafios à segurança e a privacidade, visto que, sua estrutura organizacional muitas vezes favorece a invasão por inúmeras formas de ataque, como negação de serviço, *sniffing*, MITM, entre outros. A vulnerabilidade das redes de computadores devido a ataques é um grande problema em toda a organização do negócio porque novos ataques são frequentemente descobertos.

Dentre os crescentes ataques, as *Botnets*[1] vêm se destacando pela capacidade de realizar os ataques de forma distribuída e pelos impactos provocados em grandes redes pelo mundo [1, 2, 3], como invasão a sistemas financeiros para roubar informações de cartões e senhas bancárias e além de parar o funcionamento de redes, através de ataques de DDoS. Nos dispositivos móveis e em seus serviços implementados, este tipo de ataque provocaria grandes estragos, pela facilidade de infecção e pelo capacidade de agir durante a intrusão na rede.

A utilização de defesa contra ataques de *Botnets*, algumas ferramentas são utilizadas como Antivírus, Anti-spam, F*irewall*, entre outras. Entre as ferramentas de defesas atuais, destacam-se os Sistemas de Detecção de Intrusos (IDS) como mecanismos que focam na detecção das atividades intrusivas em redes [4]. Muitas das tecnologias de IDS propostas se complementam, pois, para diferentes tipos de ambientes algumas abordagens possuem melhor desempenho.

O Sistema de Detecção de Intruso proposto visa à coleta de dados e análise de pacotes transmitidos dentro da rede, apoiada por uso de componentes para ampliar a detecção de intrusos, com ênfase a ataques provenientes de *Botnets*. O protótipo é aplicado no processo de detecção o uso de técnica híbrida, utilizando as técnicas de detecção baseadas por assinaturas e anomalias. Na aplicação da detecção por anomalia são utilizadas as técnica de inteligência artificial de Otimização por Enxame de Partículas (PSO) e Redes Neurais Artificiais (RNA), onde são identificados aspectos importantes entre elas.

A arquitetura do modelo propõe a integração dos seus componentes aos agentes do projeto NIDIA (*Network Intrusion Detection System based on Intelligent Agents*), que trabalha no uso de agentes para a detecção de intrusos.

1.1 Cenário e Definição do Problema

O crescimento das tecnologias através da proliferação das conexões de banda larga e a abertura do mercado há uma variedade de novas aplicações baseadas na *web* tem ocasionado a um conjunto difuso de ponto de injeção de tráfego de rede malicioso, o que torna as ameaças às redes de computadores numerosas e potencialmente devastadoras.

Redes computadores é uma tecnologia abrangente e traz inúmeros tipos de problemas, como as ações de usuários mal-intencionados que buscam ler ou modificar secretamente mensagens enviadas a outros destinatários, além do acesso não autorizado a serviços remotos, visando obter benefícios, chamar atenção ou prejudicar organizações ou usuários comuns. Muitas informações podem ser coletadas pelo atacante, como informações confidenciais, a exemplo do número do cartão de crédito e senhas. Estas informações são uma das principais finalidades dos ataques de *Botnets* pelo mundo, que atualmente estão disponíveis para aluguel, por parte de seus criadores.

As *Botnets* são definidas como redes de computadores infectados que recebem comandos para a realização de ataques [1,2], a principal forma de infecção é o envio de *spam*.

No ano de 2010, a taxa global de spam foi de 89,1% dos e-mails enviados pela Internet. Este número representou um aumento de 1,4% quando comparado ao ano de 2009. Deste total de *spam*, cerca de 88,2% foram enviados a partir de *Botnets* [5].

No Brasil, o número de *spam* detectado sofreu grandes alterações entre os anos de 2009 e 2010, apresentados no gráfico 1.1. Durante o ano de 2010, foram registradas altas taxas de detecção de *spam*, principalmente durante os meses de maio a julho, com variações de 5.000.000 a 7.000.000 [6], período este de realização da Copa do Mundo de Futebol, que proporciona ao crescimento de ações de ataques, devido ao evento ser considerado um dos maiores no meio esportivo mundialmente. Nos meses de novembro e dezembro do mesmo ano, estas taxas sofreram uma grande redução, chegando a ser menos de 10% dos números registrados em 2009.

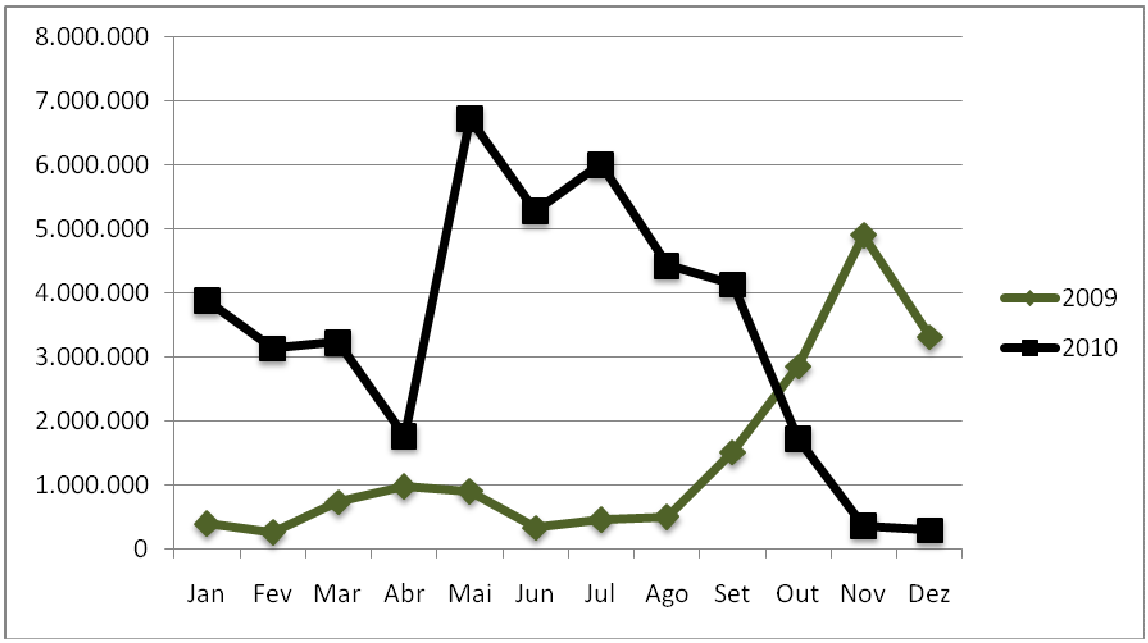


Gráfico 1.1 Comparativo mensal de envios de Spam no Brasil entre os anos de 2009 e 2010.

Fonte [6]

Apesar de sofrer variações ao longo dos anos, o número de spam, no Brasil, teve um aumento relevante nos dois últimos anos. Segundo os dados coletados pelo CERT.br, entre os anos de 2003 a 2010, foram detectados um total de 78.076.775 spams [6], conforme gráfico 1.2. Deste total, os anos de 2009 e 2010 representam 74,65% dos spams enviados pela Internet pelo país.

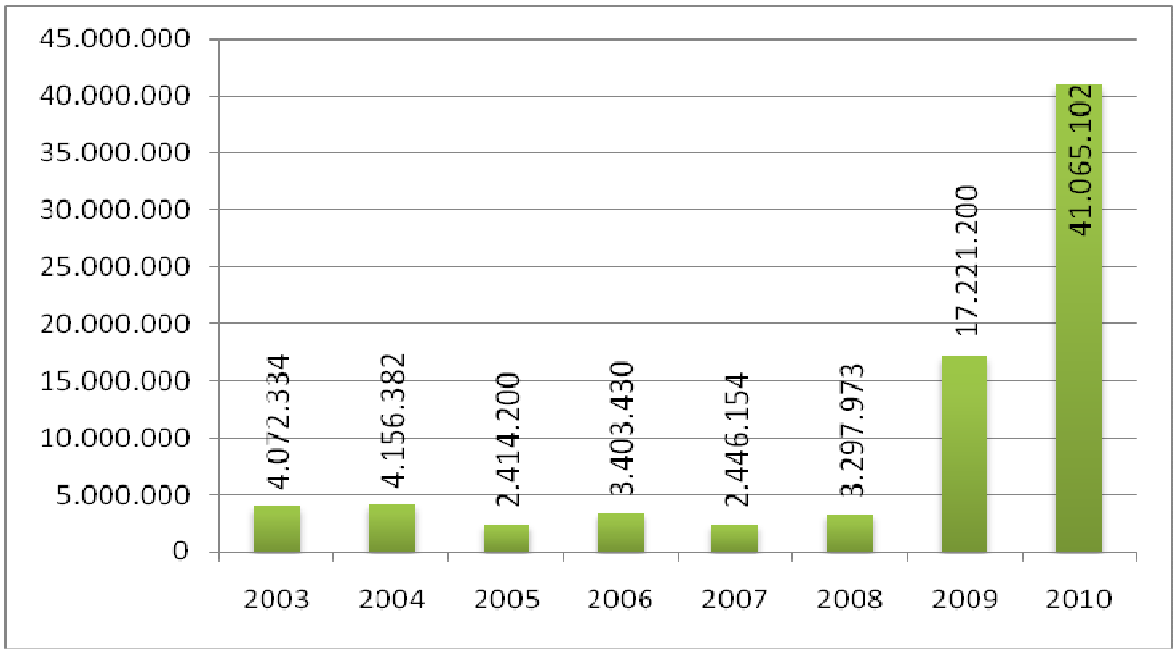


Gráfico 1.2 Comparativo anual de envios de Spam no Brasil entre os anos de 2003 e 2010.

Fonte [6]

A taxa de crescimento no Brasil, no ano de 2010, foi de 138,46% em relação a 2009, muito além do índice mundial de 1,4%, indicado anteriormente. O que demonstra grandes consequências ao país, visto que, no mundo a grande maioria dos spams é proveniente de *Botnets*. Sendo assim a necessidade da criação de ferramentas necessárias para prevenção e proteção contra essa crescente ameaça no país.

1.2 Objetivo Geral e Específicos

Esta dissertação propõe, de forma geral, o desenvolvimento de um Modelo de Arquitetura de Sistema de Detecção de Intruso em Ataques oriundos de *Botnets* utilizando o método de detecção híbrido. De modo mais específico, espera-se:

- Desenvolver um modelo de híbrido de detecção de intruso;
- Propor a arquitetura geral do sistema e interfaces dos serviços do IDS;
- Avaliar o uso das técnicas de inteligência artificial Otimização por Enxame de Partícula e Redes Neurais Artificiais na Detecção de Intruso baseada em Anomalia;
- Avaliar a ferramenta proposta através de testes e estudos de caso;
- Propor integração do modelo proposto ao modelo do NIDIA (*Network Intrusion Detection System based on Intelligent Agents*), que é um sistema de detecção de intrusos baseado em agentes inteligentes que está sendo desenvolvido na Universidade Federal do Maranhão.

1.3 Organização do Trabalho

Este trabalho de dissertação está estruturado em 6 (seis) capítulos como seguem:

No Capítulo 1 encontra-se o cenário que motivou a realização deste trabalho, o problema a ser resolvido e os objetivos.

O Capítulo 2 apresenta uma visão geral em torno do histórico, conceitos, arquitetura, ciclo de vida e classificação das *Botnets*.

No Capítulo 3 encontram-se a descrição sobre Sistemas de Detecção de Intrusos, apresentando conceitos, classificações, tipos de coletas e analise e trabalhos relacionados.

O Capitulo 4 apresenta a proposta de uma arquitetura para um sistema de detecção de intrusos oriundos de ataques de *Botnets*, que tem como base a aplicação da tecnologia de Otimização por Enxame de Partículas e Redes Neurais Artificiais, bem como a proposta de integração ao Projeto NIDIA.

O Capítulo 5 mostra um estudo de Modelo de IDS demonstrando detalhes das implementações de um protótipo da arquitetura, o ambiente de testes e os resultados destacando-se o protótipo da ferramenta.

Por fim, no Capítulo 6, as considerações finais são apresentadas, ressaltando-se as contribuições deste trabalho e as sinalizações para a continuidade deste trabalho.

2 BOTNET

2.1 Introdução

Apresenta-se, neste capítulo, uma visão geral sobre *Botnet* como rede de computadores comprometidos, enfatizando-se sua história, conceitos, arquitetura, topologias, ciclo de vida, atividades e uso do DNS nas aplicações das *Botnets*.

2.2 Visão Geral sobre *Botnet*

A *Botnet* é uma rede de computadores comprometidos, denominados de *Bot*, sob o controle remoto de um operador humano, o "*Botmaster*". O termo "*Bot*" é derivado da palavra "*Robot*". Os *Bots* são dispositivos hospedeiros projetados para executar algumas funções pré-definidas de forma automatizada, que permite ao *Botmaster* controlar remotamente as ações de ataque [1, 2, 3,7].

Diferentemente de outros *malwares*, como vírus e *worms*, que tem como principal foco é atacar o hospedeiro infectado, as *Botnets* possuem uma estrutura de comando e controle, através de Servidores C&C (Comandos e Controle) que recebem comandos do *Botmaster* e repassam para os *Bots*, para que possam executar as instruções recebidas, utilizando uma plataforma de ataque distribuído [3 ,8, 9, 10].

A *Botnet*, como em alguns outros serviços na Internet, surgiu como uma ferramenta útil. Os *Bots* foram desenvolvidos inicialmente como um indivíduo virtual, utilizando um canal IRC (*Internet Relay Chat*), para realizar atividades de seu proprietário, enquanto o mesmo estaria ocupado em outro lugar.

O primeiro *Bot* foi desenvolvido em 1989, por Greg Lindahl, um operador de servidor IRC, com finalidade de jogar o jogo *Hunt the Wumpus* com usuários IRC. Dez anos depois, Pretty Park descobriu o primeiro *worm* que utilizava um servidor de IRC como meio de controle remoto. No mesmo ano foi lançado o *SubSeven Trojan/Bot*, era uma cavalo de Tróia controlado remotamente, com recursos para roubo de senhas[3].

Em 2000, foi criado o *Global Threat* (GT) *Bot*, era uma *Botnet* que utilizava canais de comunicação IRC, através do *software mIRC*, realizava varreduras de portas abertas, ataques de negação de serviço distribuído (DDoS), além de acesso a um servidor de controle, porém não possuía mecanismo de espalhamento direto por seus *Bots* clientes.

No início de 2002, surge o *Sdbot*, escrito em C++, por um programador russo conhecido como SD. Este Bot foi mais importante para a evolução das tecnologias da *Botnet*, pois teve seu código-fonte liberado pelo autor. Muitos passaram a utilizar seu conceito ou código, para a construção de futuros *Bots*, sendo que, em 2006, empresas com a Panda e a *Microsoft*, reportaram variações deste *malware* [3].

Nos últimos anos, muitas tecnologias de *Botnet* foram aprimoradas e passaram a ser além de ferramentas de ataques, com servidor centralizado, para múltiplos servidores com estrutura descentralizada.

2.3 Arquitetura

A organização arquitetural da *Botnet* é composta por 3 (três) elementos básicos: O *Botmaster*, o Servidor de Comando e Controle, e o Cliente *Bot*. Esta arquitetura global da *Botnet* é apresentado na Figura 2.1.

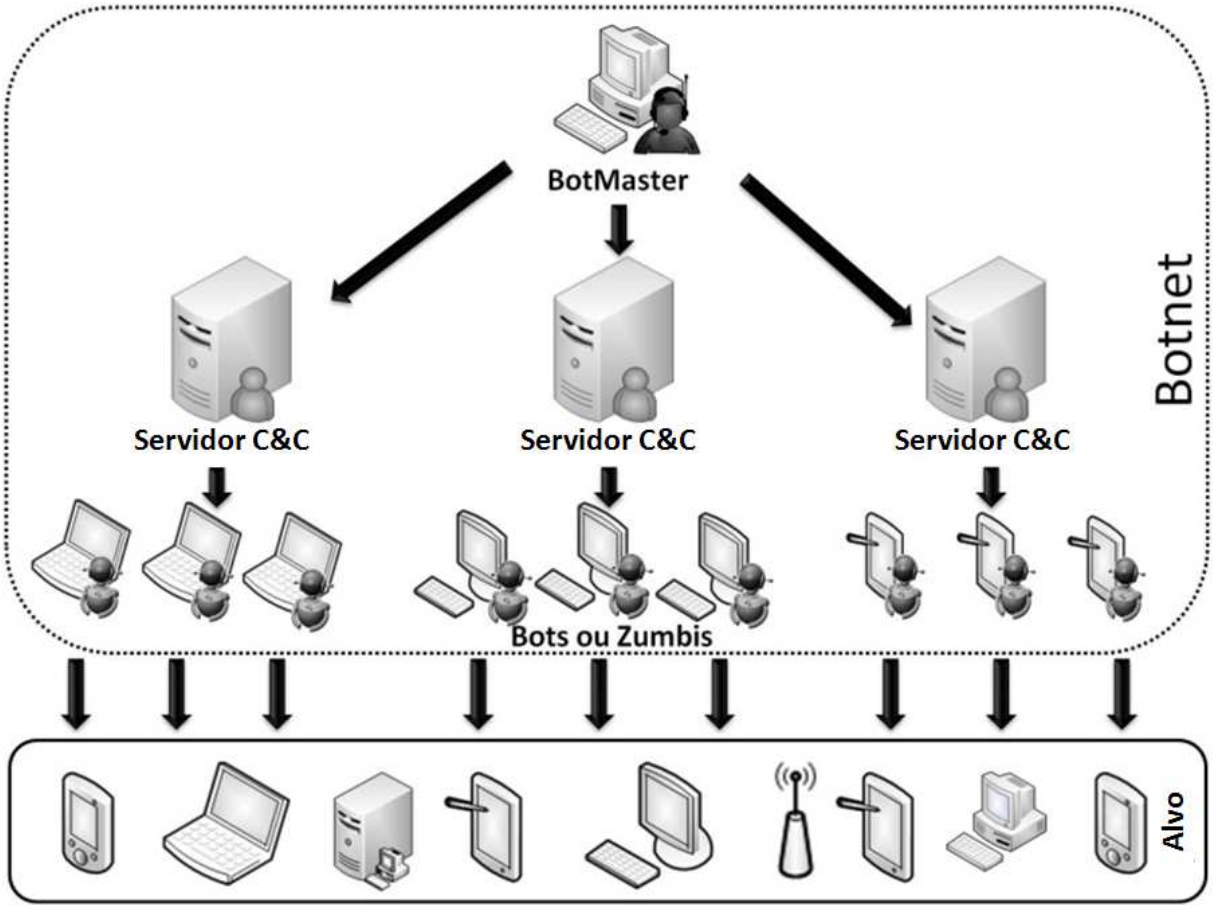


Figura 2.1 Arquitetura da *Botnet*

Fonte [7]

2.3.1 Botmaster

Denominamos de *Botmaster* o controlador humano da *Botnet*. Ele opera controlando remotamente os *Bots*, através de comandos enviados ao Servidor C&C, que fazem a comunicação com os *Bots* [1, 2, 3, 7, 8]. Dentre esses comandos utilizados podemos destacar alguns realizados nos canais IRC:

- .*recconect* – faz a *Bot* reconectar ao canal;
- .*execute* – executa algum processo em modo visível ou invisível;
- .*repeat* – repete determinada ação em uma dada quantidade de vezes;
- .*opencmd* – abre o *prompt* de comando remotamente;
- .*visit* – faz a *Bot* visitar determinada *url*;
- .*scan* – realiza um *scan* em um ip juntamente com uma porta estabelecida;
- .*pingflood* – realiza um *ping* em um ip
- .*udpflood* – realiza um *udpflood* em determinado ip.

Geralmente o *Botnet* é controlado por seu criador, mas muitas *Botnets* são criadas para a comercialização, alugadas para ações criminosas.

2.3.2 Servidor C&C

A parte principal da *Botnet* é servidor de C&C, responsável pela comunicação entre o *Botmaster* e os *Bots*, através do encaminhamento de comandos para a execução de ações. De acordo com sua estrutura de comando e controle (C&C) as *Botnets* podem ser baseado em IRC, baseado em HTTP, ou baseado em *Peer to Peer* (P2P) [1, 2, 3, 8, 9]. *Botnets* P2P usam o protocolo P2P para evitar ponto único de falha. Além disso, *Botnes* P2P são mais difíceis de localizar e de realizar o desligamento de seu servidor C&C.

Botnets mais prevalentes são baseadas em protocolo *Internet Relay Chat* (IRC) [1,3], com comando centralizado e mecanismo de controle. Há duas topologias de redes Bots [8]:

- A rede centralizada com base no IRC ou o protocolo HTTP;
- A rede Descentralizada com base em protocolos P2P.

2.3.2.1 Topologia Centralizada

Nesta topologia, um ponto central é responsável pela troca de comandos entre o *Botmaster* e os Clientes *Bots*. Neste modelo, o *Botmaster* escolhe um anfitrião, normalmente um computador com acesso a banda larga, para ser o ponto central, ou seja, o Servidor de C&C, que monitora o status dos *Bots* e envia as instruções dadas pelo proprietário. Este Servidor C&C executa determinados serviços de rede como IRC ou HTTP [8,11, 12]. Esta topologia é a mais comum, devido à estrutura simples, facilidade de gerenciamento e alta velocidade. A principal vantagem desta topologia é a baixa latência de mensagens utilizadas pelo *Botmaster* para a realização de ataques pelos *Bots*, uma vez que todas as conexões acontecem através do Servidor C&C. A desvantagem deste modelo é que o Servidor C&C é o ponto crítico, se alguém conseguir descobrir sua ação, pode eliminá-lo e assim a *Botnet* inteira será inútil e ineficaz. A Figura 2.2 mostra a arquitetura de comunicação de base para um modelo centralizado.

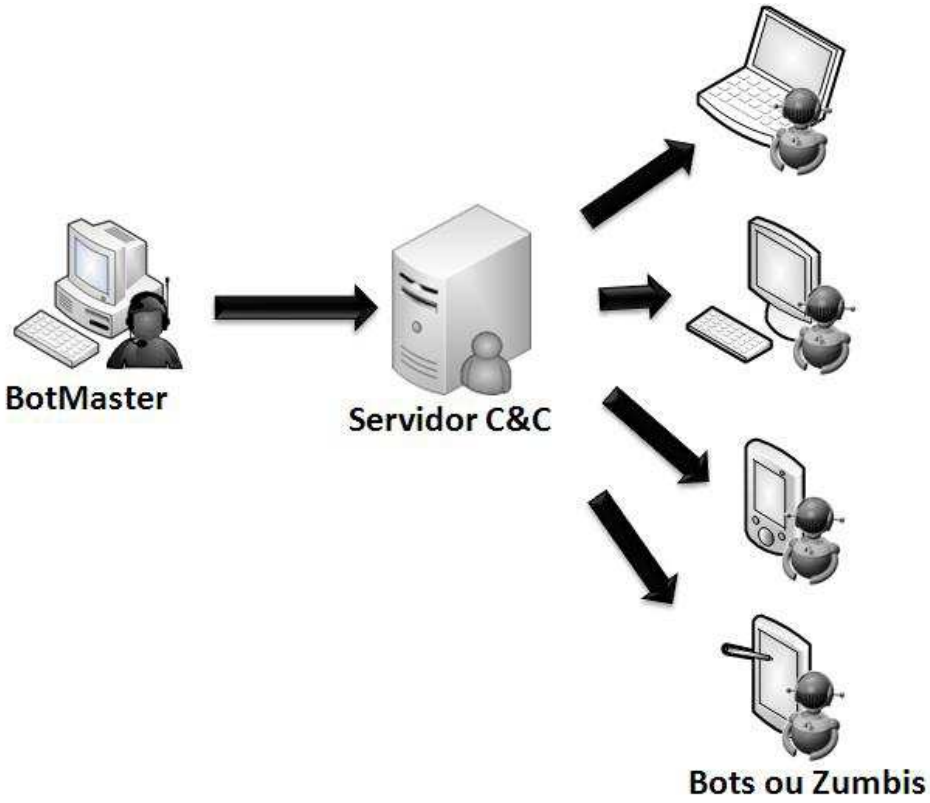


Figura 2.2 Topologia Centralizada

Fonte [7, 8, 9]

Os protocolos IRC e HTTP são dois tipos comuns utilizados por Servidores C&C para comunicação, que consideramos *Botnets* neste modelo baseado em IRC e HTTP, respectivamente.

A *Botnet* baseada em IRC utiliza o protocolo de troca de mensagem em tempo real baseado no modelo cliente-servidor, que pode ser usado em muitos computadores em redes distribuídas. Algumas vantagens que fizeram o protocolo IRC ser amplamente utilizado na comunicação remota para *Botnets* são [8, 12, 13]:

- Baixa latência de comunicação,
- Comunicação em tempo real anônimo,
- Capacidade do grupo (muitos para muitos) e privados (um a um) de comunicação;
- Instalação e comandos simples e;
- Grande flexibilidade na comunicação.

Os comandos básicos são conectar a servidores, entrar em canais e postar mensagens nos canais.

Neste modelo, o *Botmaster* pode comandar seu *Bots* como um todo ou o comando de alguns dos *Bots* seletivamente usando comunicação um-para-um. O servidor de C&C executa serviços de IRC que é o mesmo com outros serviços de IRC padrão. *Botmaster* geralmente cria um canal designado aos servidores C&C onde todos os Bots vão conectar, aguardando comandos no canal que irá instruir cada *Bot* ligado para fazer a ligação com *Botmaster*.

As *Botnets* baseadas em HTTP tornaram-se conhecidas devido aos pesquisadores de segurança focarem seus trabalhos no acompanhamento do tráfego de IRC para detectar *Botnets* [8]. Por conseguinte, os atacantes começaram a utilizar o protocolo HTTP como um canal de comunicação de C&C para fazer as *Botnets* se tornam mais difíceis de detectar. A principal vantagem de utilizar o protocolo HTTP está em esconder o tráfego dos *Bots* na web, passando por um tráfego normal, então ele pode facilmente passar por *Firewalls* com porta baseado em mecanismos de filtragem e evitar a detecção IDS [8,14]. Normalmente os *Firewalls* bloqueiam o tráfego de entrada/saída para as portas indesejadas, que incluem muitas vezes a porta do canal IRC.

Para melhorar a eficácia das *Botnets* de topologia centralizada, muitos controladores utilizam vários servidores para a realização dos ataques, assim a *Botnet* terá

maior abrangência e diminuição da carga de controle dos Servidores C&C. Esta arquitetura de múltiplos servidores é descrita na Figura 2.3.

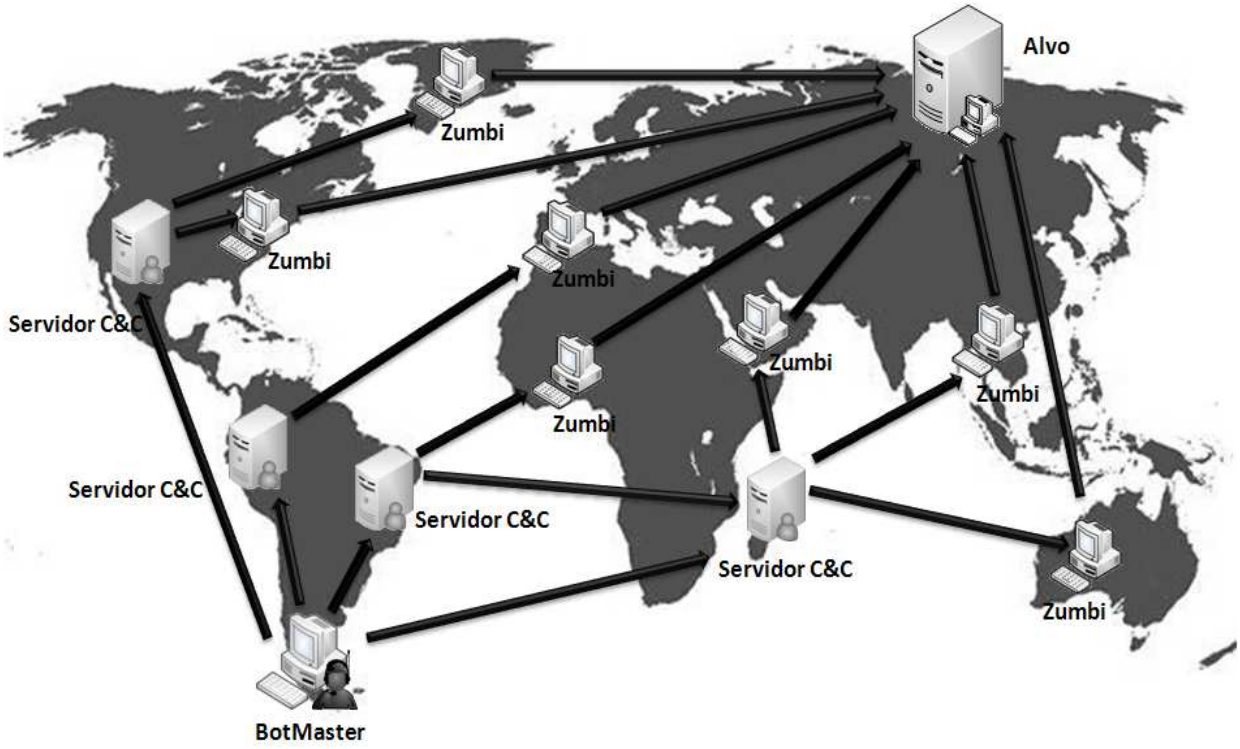


Figura 2.3 Topologia Centralizada com Múltiplos Servidores

Fonte [8]

2.3.2.2 Topologia Descentralizada ou Distribuída

Na topologia descentralizada, como mostrado na Figura 2.4, os *Bots* não se comunicam com o Servidor C&C central, mas com outros *Bots* [15,16, 17]. Esta abordagem aumenta a confiabilidade da *Botnet*, mas complica a gestão e reduz a velocidade de ação, devido a inúmeras trocas de mensagens pelos *Bots* para a realização dos ataques.

A principal idéia é que cada *Bot* tenha uma "lista de amigos" ou encaminhe a lista de nós com os endereços de IP de outros clientes *Bots*. A lista de nós é construída quando um Cliente *Bot* infecta uma nova vítima, sua lista de nós é passada para a vítima. Se o novo Cliente *Bo*t já tenha sido infectado antes, atualizações da lista ocorrem através das informações enviadas pelo *Bot* que executou a nova infecção.

Esta topologia tem como principal desvantagem a dificuldade de implementação de técnicas de gerenciamento e troca de mensagens entre os *Bots* para seu funcionamento, devido sua estrutura sem um ponto central de controle. Atualmente, eles são relativamente

pequenos, mas, no futuro, eles podem se tornar maiores e representar um grande desafio para a detecção de *Botnets* [11].

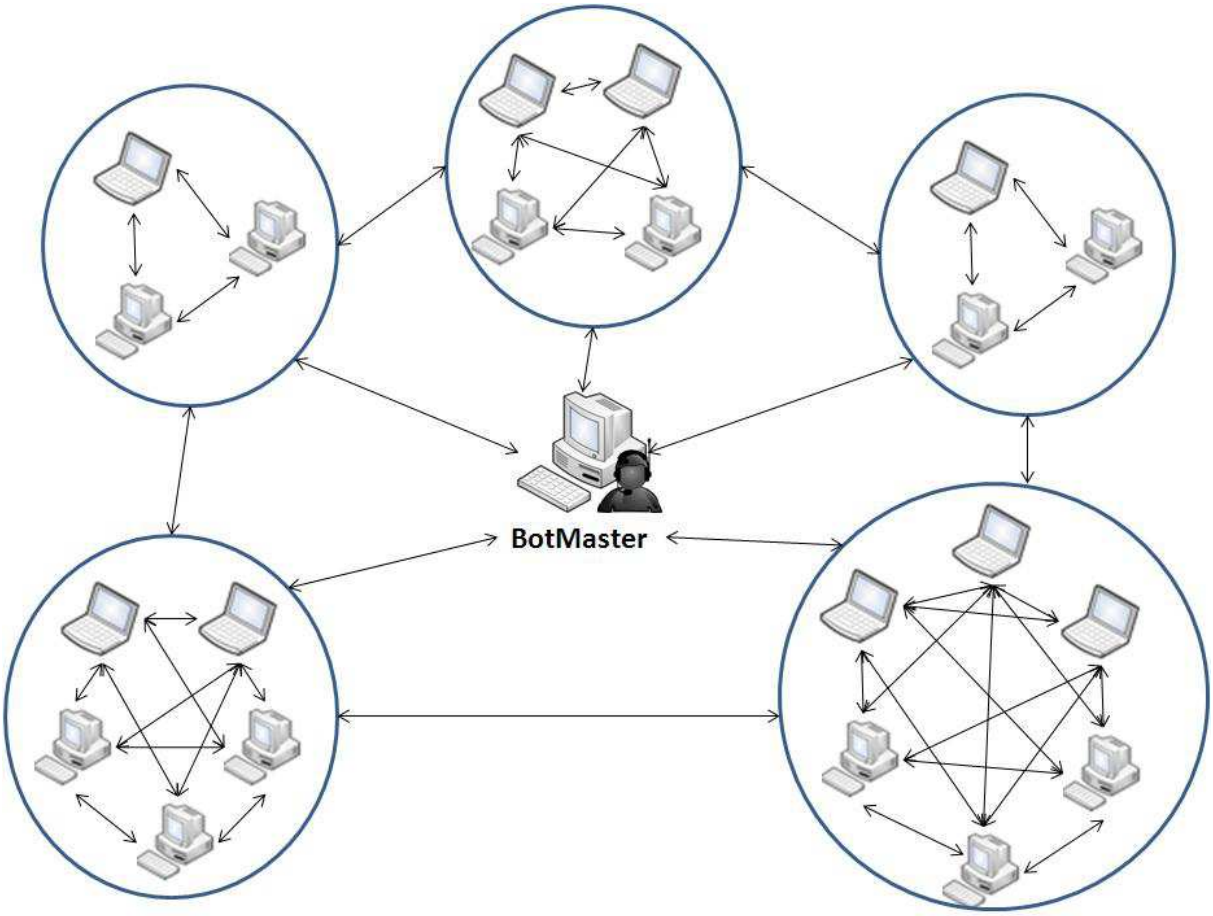


Figura 2.4 Topologia Distribuída

Fonte [7, 8, 9]

2.3.3 *Bot* ou *Zumbie*

É o computador ou dispositivo comprometido, controlado remotamente por um *Botmaster* para a execução de algumas ordens através dos comandos recebidos. Depois que o código do *malware* foi instalado no computador infectado, o computador se torna um *Bot* ou *Zumbie* [1, 2, 3, 7]. Os *Bots* normalmente se distribuem através da Internet, a procura de computadores desprotegidos e vulneráveis para infectar. Quando eles encontram um computador desprotegido, eles o infectam e depois enviam um relatório ao *Botmaster* [3].

2.4 Ciclo de Vida

Uma *Botnet* típica pode ser criada e mantida em quatro fases [2], conforme Figura2.5:

a. **Infecção Inicial:** os *Bots* são infectados de diversas maneiras, como por exemplo, sendo ativamente explorada vulnerabilidade existente no dispositivo, *malware* baixado automaticamente durante a visualização de páginas web ou automaticamente baixada e executada através de abrir um anexo de e-mail. Em redes sem fio esta infecção pode ser facilita pela estrutura deste tipo de rede, o ataque pode ser lançando direto no dispositivo, sem a necessidade de utilizar a estrutura da rede.

b. **Injeção Secundária de Ligação:** após serem infectados os *Bots* executam o código de *Bot* e começa o *rally*, processo de busca da ligação com o Servidor C&C, através do seu nome de domínio. Este procedimento ocorre em grupo, visto que, os *Bots* realizam suas tarefas em conjunto. Após esta etapa, os Clientes *Bots* aguardam os comandos que serão enviados pelo *Botmaster* ao Servidor.

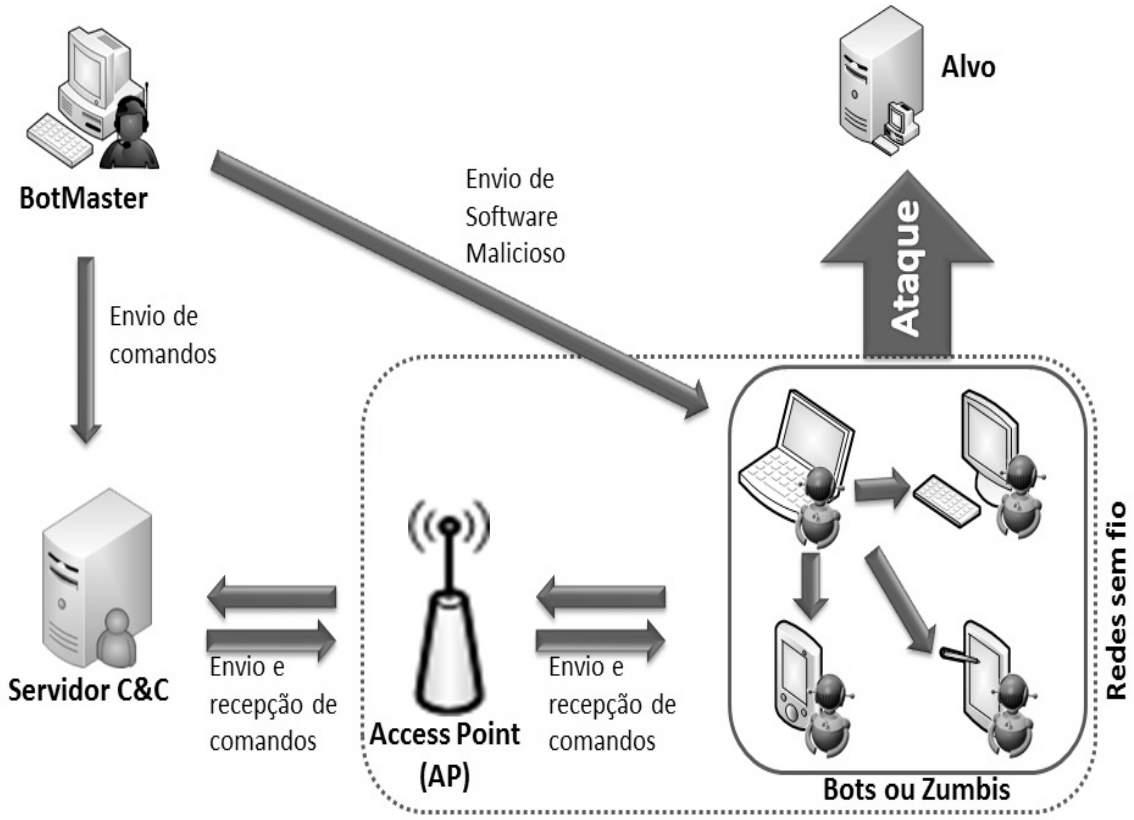


Figura 2.5 Ciclo de vida da *Botnet* em Redes sem Fio

Fonte [1]

c. **Atividades Maliciosas:** nesta etapa o *Bot* comunica-se com o servidor C&C que envia as instruções para a realização de ações de ataque como envio de *spam*, DDoS, entre outros.

d. **Atualização e Manutenção:** devido à grande disponibilidade de informações a ser trocada em uma *Botnet*, como a captura de informações pelos *Bots*, se faz necessário à atualização de novos comandos, além da mudança de servidor C&C ocasionada pela queda do serviço do servidor anterior. Na migração de Servidores C&C a *Botnet* ao migrar para outro Servidor C&C servidor, é preciso realizar a consulta do DNS. Este processo de consulta ocorre também na mudança do endereço IP do Servidor C&C, se um Servidor C&C utiliza IP dinâmico, o endereço IP correspondente pode ser alterado a qualquer momento, além de que o *Botmaster* também pode alterar o endereço IP do Servidor C&C intencionalmente, para dificultar a detecção. Com a mudança do endereço IP, os *Bots* não conseguir ligar ao antigo endereço IP, sendo assim necessita enviar a consulta de DNS para acesso ao novo servidor C&C.

2.5 Atividades

As *Botnets*, como enunciado anteriormente, possuem uma estrutura de controle para a realização de várias atividades [11], estas ações são elencadas abaixo:

- O ataque de *Distributed Denial of Services* (DDoS) é realizado através da ação conjunta dos *Bots*, que visa em um ataque a um sistema computacional ou rede provocando a perca de serviços.
- O *Key Logging* com a ajuda de um *key logger*, programa cuja finalidade é registrar tudo o que foi digitado, o *Bot* obtém informações sigilosas, ele recebe o comando e executa a ação de recolha de informações como captura de senhas, números de cartão de créditos e afins.
- No *Sniffing Traffic* os *Bots* também utilizam um *sniffer* para observar os dados de texto, com finalidade obter informações confidenciais, como nomes de usuários, senhas, dados bancários, entre outros.

- Na Manipulação de Enquetes online os *Bots*, que possuem endereço de IP diferente, realizam a votação da enquete, validado como realizado por pessoas reais.
- Distribuição de um novo *malware* é realizada pelos Bots através dos mecanismos de *download* e execução de um arquivo via HTTP ou FTP.

2.6 Considerações Finais

Neste Capítulo, percebemos o quanto a evolução das *Botnets*, diferentemente dos outros *malwares* pode ser gerenciado pelo seu controlador, tornando ainda maior seu poder de invasão. Entre as topologias, o modelo centralizado destacou as *Botnets* baseadas em IRC e HTTP, enquanto no modelo descentralizado verificamos que os Clientes *Bots* se comportam tanto como Clientes ou Servidores, nas *Botnets* baseada em P2P.

A comunidade de segurança necessita de soluções para deter o crescimento desse novo invasor, com o objetivo de tornar as redes das organizações mais seguras e preparadas para essa ameaça virtual.

Assim, o Capítulo a seguir apresentará a descrição sobre Sistemas de Detecção de Intrusos como tecnologia de combate ao avanço das *Botnets*.

3 SISTEMA DE DETECÇÃO DE INTRUSO

3.1 Introdução

Neste capítulo, apresentamos uma descrição dos Sistemas de Detecção de Intrusos, os tipos de coletas e quais as técnicas de análise empregadas, dando ênfase à detecção de *Botnets*. Abordarmos os trabalhos relacionados com a arquitetura proposta neste trabalho.

3.2 Visão do Sistema de Detecção de Intrusos

A segurança na maioria dos ambientes empresariais, atualmente, é baseada no conceito de defesa em profundidade, onde várias camadas de defesas são usadas para impedir os adversários de violarem as políticas de segurança da empresa. A defesa em profundidade é baseada na premissa de que, mesmo se um adversário penetra em uma das camadas de defesa, ele não será capaz de causar muito dano, porque as outras camadas irão fornecer um adequado nível de proteção.

Mesmo que os mecanismos preventivos como *Firewall*, criptografia e autenticação, apresentam a primeira linha de defesa para os usuários mal-intencionados, uma camada adicional de defesa chamado de detecção de intrusão é frequentemente usada para proteger as redes.

Sistemas de Detecção de Intrusos (IDS) são ferramentas de segurança que, assim como outras medidas, como antivírus, *Firewalls* e sistemas de controle de acesso, se destinam a reforçar a segurança dos sistemas de informação e comunicação [4,19]. IDS são considerados a segunda força da segurança, pois visa coletar e avaliar os dados de um sistema e tomar medidas de prevenção e proteção [20,21]. A fim de detectar tal comportamento, sistemas de detecção de intrusão normalmente contêm dois tipos de componentes [22]:

- Componentes de coleta de dados;
- Componentes de análise de dados.

Existem várias vantagens para a implementação de um Sistema de Detecção de Intrusão em uma rede. O IDS atua como um impedimento para os atacantes. Este IDS também pode detectar ataques que ignoram outras medidas de segurança. Além disso, fornece informações sobre invasões que podem ser usados para responder a tais ataques. Um IDS

também é útil na realização de controle de qualidade de projeto de segurança e administração da rede.

3.3 Classificação de IDS por Tipo de Coleta de Dados

Componentes de coleta de dados são compostos de entidades que são responsáveis pelo acompanhamento e coleta de dados sobre as atividades do usuário e da aplicação. Os dados coletados são então usados por um segundo tipo de componentes, chamado de análise de componentes [20, 21,22]. Duas abordagens principais para a coleta de dados têm sido tradicionalmente utilizadas, que classificamos em dois tipos de sistemas de detecção de intrusão [23]:

- IDS baseado em Host **(*Host Intrusion Detection System* – HIDS)** funciona em um Host e centra-se na recolha de seus dados, geralmente através de logs de auditoria do sistema operacional.
- IDS baseado em Rede **(*Network Intrusion Detection System* – NIDS)** tem funcionamento em rede e centra-se na recolha de dados por acompanhamento do tráfego que flui através de uma rede.

3.3.1 IDS Baseada em Host

O IDS baseado em *Host*, ou HIDS, protege somente o sistema no qual ele reside, ou seja, apenas o *Host* onde se encontra instalado e não a sub-rede inteira [24], de acordo com a Figura 3.1. Esta ferramenta utiliza a placa de rede da máquina onde está inserida, funcionando normalmente e operando em modo não promiscuo. Esta é uma vantagem do HIDS, visto que, em alguns casos os NIC´s não são capazes de operar em modo indiscriminado, embora a maioria dos NIC´s modernos execute esta funcionalidade.

Outra vantagem do HIDS é a capacidade de adequar seu conjunto de regras para o sistema *Host* em particular. A capacidade de ajustar suas regras irá melhorar seu desempenho e diminuir os falsos positivos.

A grande vantagem de um HIDS, no entanto, reside na sua capacidade de detectar alterações específicas em arquivos no sistema operacional do seu hospedeiro. Ele pode controlar o tamanho dos arquivos e verificar os *checksums* para garantir que os arquivos essenciais do sistema não sejam modificados de forma maliciosa, sem

alguém perceber. Além disso, pode ver o tráfego dentro de um sistema que nunca passa a rede e, portanto, nunca seria visto pelo NIDS.

Existem algumas desvantagens para a eleição de usar um HIDS. Você terá que escolher aquele que é adaptado ao seu sistema operacional. Um HIDS irá adicionar carga para o *Host* em que está configurado, pois o processo HIDS irá consumir recursos.

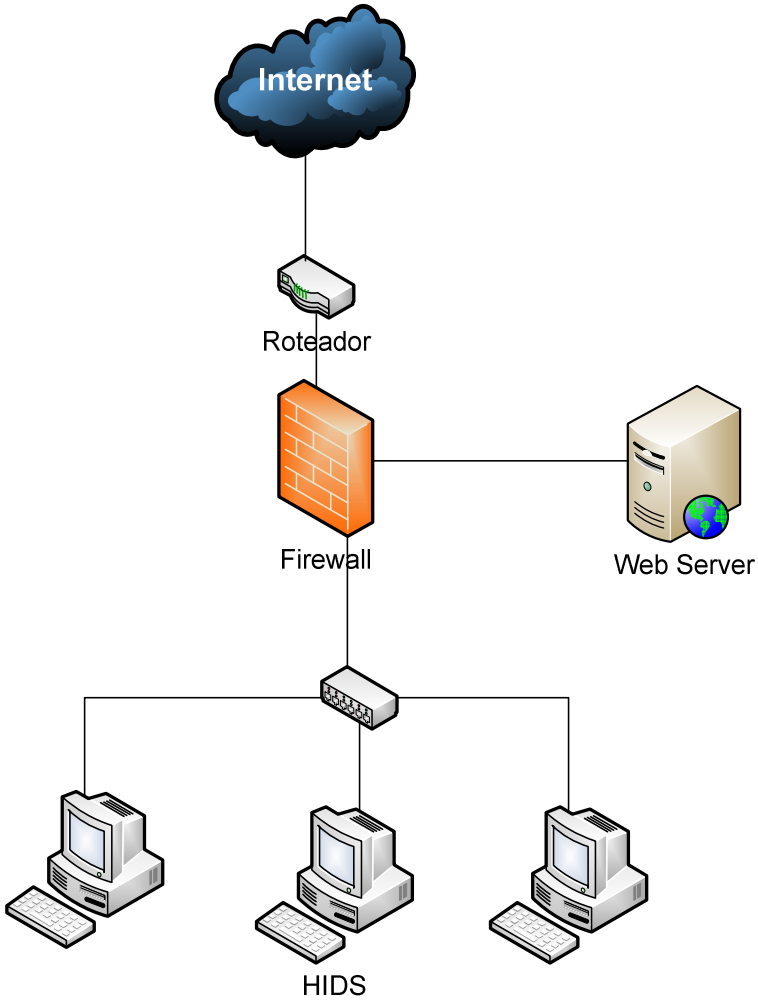


Figura 3.1 Funcionamento da HIDS

Fonte[24]

3.3.2 IDS Baseada em Rede - NIDS

O NIDS deriva seu nome pelo fato de monitorar um segmento inteiro da rede, ou da sub-rede, conforme Figura 3.2. Isto é feito alterando o modo de funcionamento da placa de interface de rede (NIC) [24]. Normalmente, uma placa de rede opera em modo não promíscuo, ouvindo apenas pacotes destinados a seu próprio endereço MAC (Controle de

Acesso ao Meio). Outros pacotes que não são encaminhados a pilha são ignorados. Para monitorar todo o tráfego no segmento da rede e não apenas os pacotes endereçados a própria máquina, o NIDS deve aceitar todos os pacotes e enviá-los até o banco de dados de tráfego coletado. Esta ação é conhecida como modo promíscuo. Neste modo, a ferramenta de detecção pode escutar todas as comunicações do segmento da rede. No entanto, isso não é tudo o que é necessário para garantir que o NIDS seja capaz de escutar todo o tráfego na sub-rede.

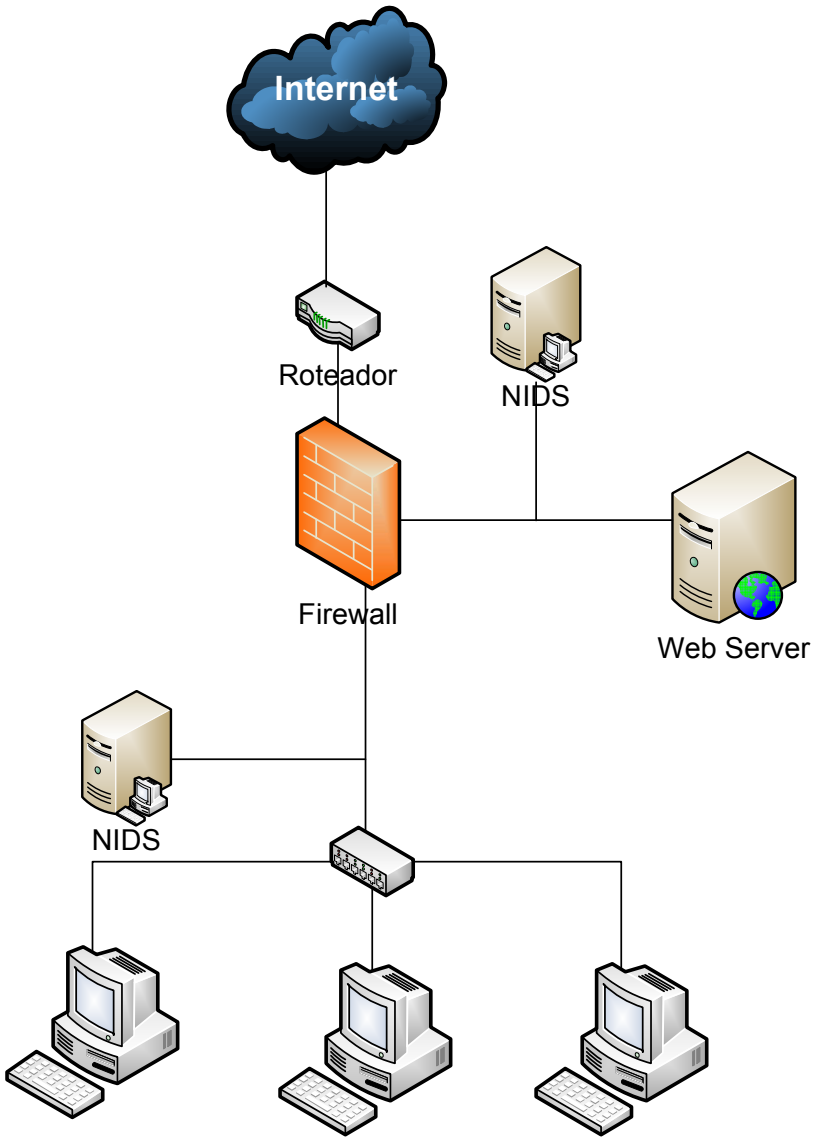


Figura 3.2 Funcionamento do NIDS em Rede

Fonte[24]

A vantagem de um NIDS é que ele não tem qualquer impacto sobre os sistemas ou redes que está monitorando. A ferramenta não adiciona nenhuma carga para os anfitriões,

sendo assim, quando um atacante vier a comprometer um dos sistemas que está sendo observado não pode identificar o NIDS. A desvantagem deste tipo de monitoramento é não examinar o tráfego total da rede, mas somente o segmento no qual o sensor está conectado, ou seja, parte da rede que não está ao alcance do sensor pode ficar sem ter pacotes coletados para análise.

3.4 Classificação de IDS por Tipo de Análise de Dados

Depois que os dados são coletados é necessário a análise dos dados para a detecção da atividade mal-intencionada. Um IDS normalmente incorpora mecanismos que analisam automaticamente os dados recolhidos por vários coletores para detecção de atividades maliciosas. A análise dos dados envolve a consolidação destes dados do IDS, possivelmente em um local central e identifica as atividades maliciosas [22]. Podemos destacar 03(três) tipos de técnicas de análise:

- IDS baseado em Assinatura
- IDS baseado em Anomalia
- IDS híbrida

3.4.1 IDS Baseado em Assinatura

Este tipo de IDS visa na detecção de intrusos através da utilização de assinaturas de ataques. Estas assinaturas são compostas por um conjunto de regras que caracteriza o intruso. Este processo facilita a detecção, quando uma assinatura para um ataque for identificada, é realizado um processo simples de comparação de um pacote contra a assinatura do intruso especificado [4, 20, 22].

A técnica de detecção baseada em assinatura está sendo amplamente utilizada hoje por sistemas comerciais, pois permite a detecção de ataques muito precisos, resultando em baixas taxas de falso positivo. No entanto, as técnicas de detecção baseada em assinaturas podem ser utilizadas para a detecção de *Botnets* conhecidas, visto que, suas características de ataque já estão previstas no banco de dados de assinatura, entretanto, esta solução não é útil para *Bots* desconhecidos, pois suas ações não tratadas na identificação deste tipo de IDS[20,22], sendo necessário a aplicação da detecção por anomalia.

3.4.2 IDS Baseado em Anomalia

Técnicas de detecção baseado em anomalias [19, 20, 22] visam detectar intrusos com base nas anomalias do tráfego de rede, tais como a alta latência da rede, elevados volumes de tráfego, comportamento anormal do sistema que poderiam indicar a presença de atividades maliciosas na rede e tráfego em portas incomuns [1, 4].

Este tipo de IDS utiliza técnicas de aprendizagem e baseiam-se em estabelecer um modelo explícito ou implícito que permite que os padrões analisados sejam categorizados. A técnica visa em construir perfis que constitui o tráfego normal e geram alertas se um fluxo de rede é significativamente diferente do perfil de rede padrão [25]. Uma característica singular dessa arquitetura é a necessidade de dados rotulados para treinar um padrão comportamental, ou seja, um procedimento que coloca exigências severas sobre os recursos [26]. Em muitos casos, a aplicabilidade dos princípios da aprendizagem de máquina coincide com técnicas estatísticas, embora a primeira seja focada na construção de um modelo que melhora o seu desempenho com base em resultados anteriores.

A abordagem baseada em anomalia assim como a abordagem por assinatura necessita ser configurados antes de sua implantação. Neste caso o IDS coleta o tráfego que estará sendo monitorado, e posteriormente compara esse tráfego de entrada com o padrão da rede, a fim de detectar anomalias. A grande vantagem da abordagem baseado em anomalia é a autoaprendizagem pode detectar novos ataques desconhecidos assim que eles acontecem [25]. Algumas técnicas utilizadas para a detecção de anomalia são citadas abaixo:

- Redes Neurais Artificiais;
- Otimização por Enxame de Partícula.

3.4.2.1 Redes Neurais Artificias

Uma rede neural artificial (RNA) é um modelo de processamento de informação que é inspirado nos sistemas biológicos nervoso, definida por uma rede de neurônios interconectados, fornecem uma funcionalidade de aprendizagem paralela e distribuída nas tomadas de decisão [27,28,29,30,31]. Cada neurônio, representado na Figura 3.3, atua como um elemento de processamento de informação, sua função básica é somar as entradas, ponderado por fatores denominados pesos sinápticos dos elementos do vetor de entrada e aplicar este resultado como entrada de uma função não linear denominada função de ativação [30].

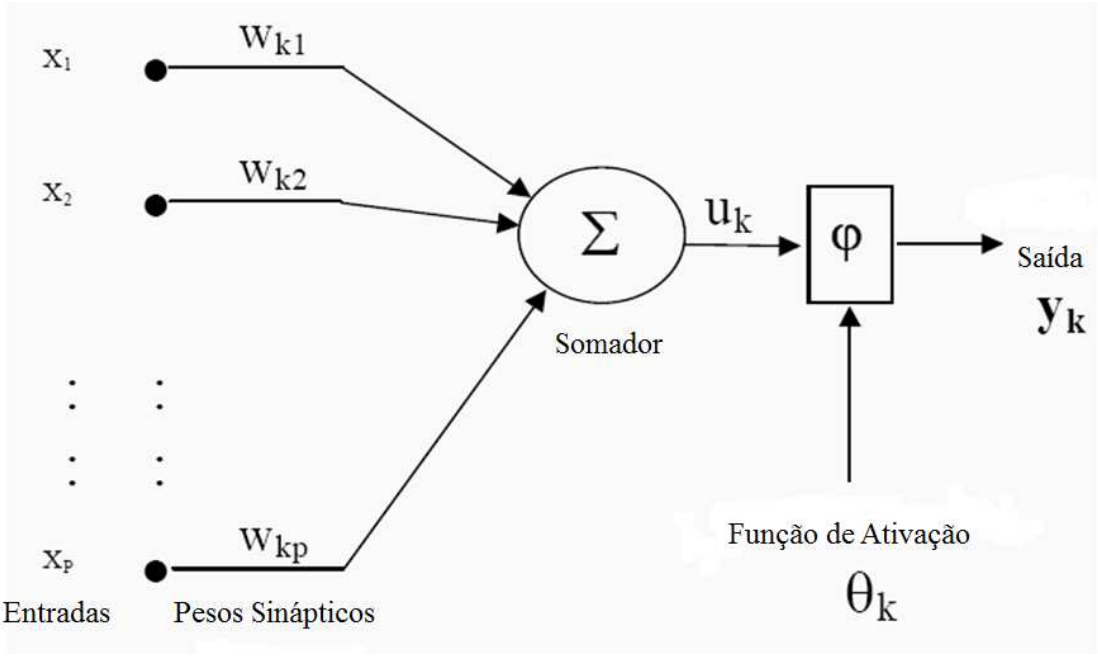


Figura 3.3 Modelo de Neurônio Artificial
Fonte[30]

Como no cérebro humano, as redes neurais também aprendem a executar uma tarefa específica ou aplicada para definir um parâmetro de peso, pois adquirem conhecimento através do processo de aprendizagem e armazenam o conhecimento adquirido [27,30]. A arquitetura de uma rede neural é definida pela forma na qual os neurônios estão organizados e interconectados, ou seja, o número de camadas, o número de neurônios por camada, tipos de conexão entre os neurônios e a topologia da rede, especificado na Figura 3.4.

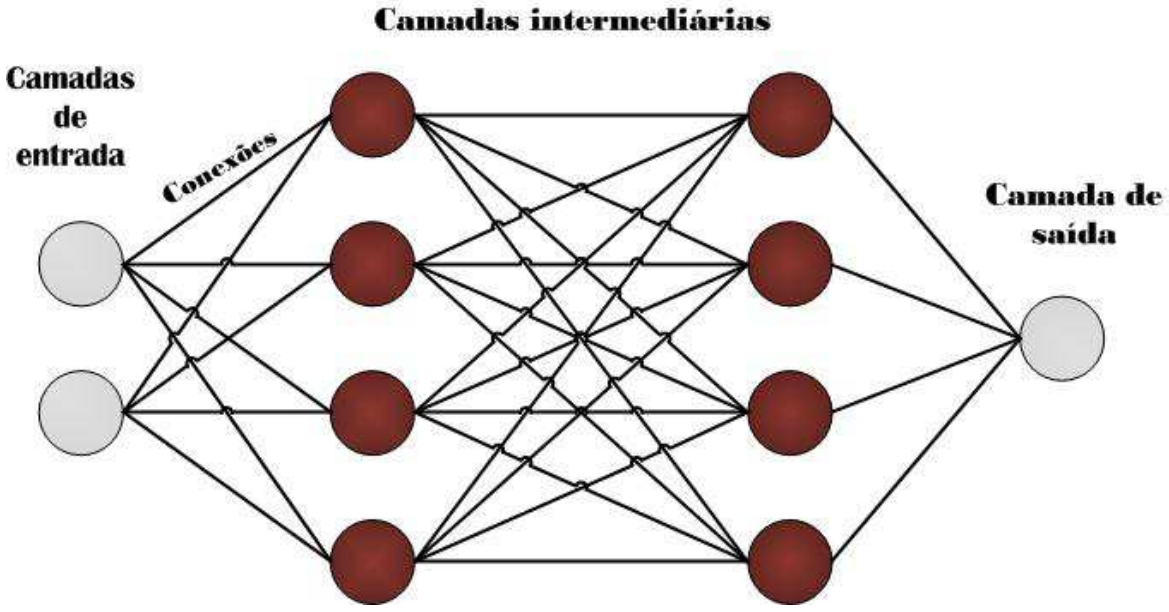


Figura 3.4 Arquitetura de uma rede neural
Fonte[30]

As RNAs realizam com muito sucesso o reconhecimento e a correspondência complicada ou padrões incompletos. O processo de aprendizagem é essencialmente um processo de otimização, no qual os parâmetros do melhor conjunto dos coeficientes são encontrados para resolução de um determinado problema [27]. A aplicação de maior sucesso das redes neurais é a classificação ou categorização, reconhecimento de padrões e respostas a um evento [27, 31].

Existem dois diferentes métodos de aprendizagem para as redes neurais: o supervisionado e não supervisionado[27, 30, 32]. No método de aprendizado supervisionado, a rede aprende a saída desejada para um dado de entrada ou padrão. A arquitetura bem conhecido da rede neural supervisionada é o *MultiLevel Perceptron* (MLP), ou Perceptron de Múltiplas Camadas, o MLP é empregado para problemas de Reconhecimento de Padrões, que apresentam uma ou mais camadas intermediárias, denominadas de camadas oculta ou escondidas [27, 30, 32]. Por outro lado, no método de aprendizagem não supervisionada, a rede aprende sem especificar saída desejada. *Self-Organizing Maps* (SOM) ou Mapa Auto-Organizável são redes neurais artificiais com apenas duas camadas: camada de entrada e a camada de saída [32]. SOM são empregados para problemas de classificação [27], destacam-se as potencialidades de visualização de dados multivariados, análise de agrupamentos, mineração de dados, descoberta de conhecimento e compressão de dados.

A técnica de RNA têm sido utilizada em várias aplicações de detecção de intrusão baseada em anomalia, principalmente por sua capacidade de processo paralelo de informações, flexibilidade e adaptabilidade às mudanças ambientais [27,31,33]. RNAs devidamente treinadas são capazes de fornecer resultados precisos para os padrões que não são muito diferentes dos padrões de treinamento [33].

3.4.2.2 Otimização por Enxame de Partícula

Particle Swarm Optimization (PSO), Otimização por Enxame de Partículas, é uma técnica de inteligência computacional evolutiva, desenvolvida por Eberhart e Kennedy, em 1995, inspirada no comportamento social através da inteligência de enxames biológicos [34,35].

O conceito de enxame de partículas originou-se como uma simulação do sistema social simplificado, a intenção inicial era graficamente simular a coreografia de um bloco de

aves em voo ou cardume de peixes. No entanto, verificou-se que o modelo enxame de partículas pode ser usado como um otimizador [35, 36].

No PSO os algoritmos são utilizados para encontrar uma solução para um problema de otimização em algum espaço de busca *n*-dimensional. Nesta técnica o sistema é inicializado com uma população candidatas a solução, denominadas de Partículas [34, 35, 37].

Seja dado o vetor $V_i(t)=\{v_{i,1}(t), v_{i,2}(t),... v_{i,m}(t)\}$ a representação da posição da partícula no tempo *t*, cada partícula mantém o registro de suas coordenadas no espaço do problema em que estão associadas e determina sua melhor posição que alcançou até o tempo *t*, esse valor é o que denominamos de *pbest*. Outro valor que é controlado pelo PSO é o melhor valor obtido pelo enxame até o tempo t, este valor é o melhor valor entre as partículas, chamado de *best* global ou *gbest* [34,35]. A

Para facilitar a exploração do espaço de busca, tipicamente cada partícula deve ter certo nível de aleatoriedade em seu movimento, para que o movimento do enxame tenha certa capacidade exploratória, ou seja, cada partícula deve ser influenciada pelo restante do enxame, mas também deve explorar independentemente sua posição [36,38]. A partícula representa um indivíduo com 02 (dois) tipos de informação:

- Experiência própria, ela sabe que escolhas foram as melhores no passado;
- Conhecimento de outros indivíduos, suas escolhas e sucesso associado.

A PSO é uma técnica de computação baseada em inteligência que não é bastante influenciado pelo tamanho do problema, pode convergir como solução ótima em muitos problemas, onde a maioria dos métodos analíticos não converge [35, 38,39]. Portanto, pode ser aplicada a diferentes problemas de otimização em sistemas computacionais. Além disso, a PSO tem algumas vantagens sobre outras técnicas de otimização:

- É mais fácil de implementar por existir menos parâmetros para ajustar;
- Cada partícula se lembra de seu próprio valor anterior, bem como o melhor bairro, por isso, tem uma capacidade de memória mais eficaz;
- É mais eficiente na manutenção da diversidade do enxame, uma vez que todas as partículas usam as informações relacionadas com a partícula de maior sucesso, a fim de melhorar a si mesma.

Aplicação da técnica de PSO vem sendo empregadas no uso de detecção de intrusos baseadas em anomalia [39, 40], devido sua facilidade de implementação e a capacidade de geração de valores otimizados, principalmente na definição de modelos

padrões [39] ou em colaboração com outras técnicas como Máquinas de Vetores de Suporte[40].

3.4.3 IDS Híbrida

Os IDS que utilizam a aplicação das técnicas de assinatura e anomalia em conjunto, são denominamos de IDS Híbrido ou IDS-H. Este IDS visa aumentar o poder de detecção de intrusos, pois detectam tanto ataque conhecidos, quanto desconhecidos, conforme descrito na Figura 3.4.

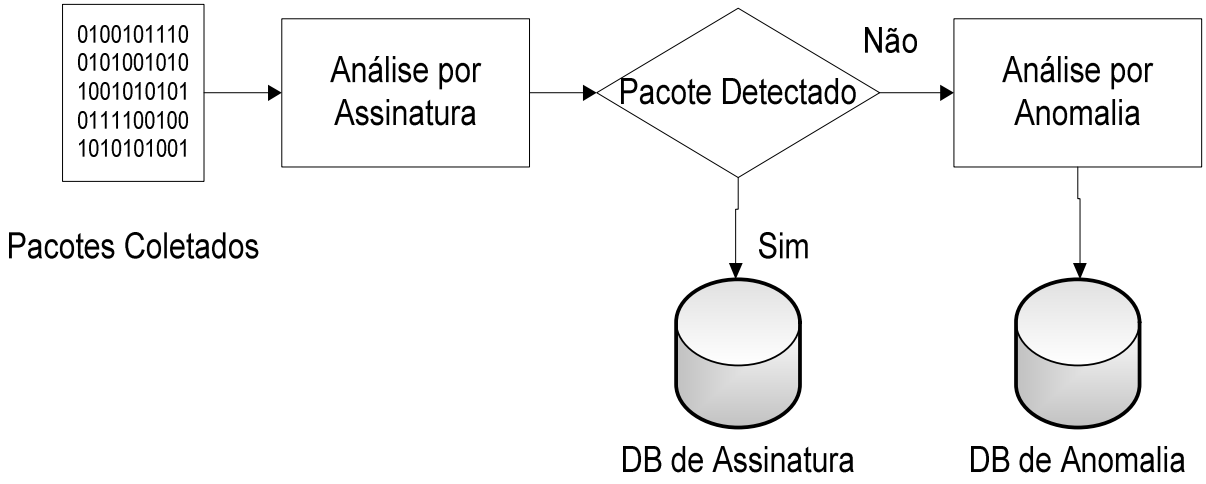


Figura 3.5 Detecção Híbrida de Intruso

Fonte[2,4]

Esta ferramenta faz o uso da análise em seu banco de assinatura, caso não encontre nenhuma regra, encaminha o pacote para ser analisado pela técnica de anomalia. Na detecção de *Botnets*, este tipo de análise se torna mais eficaz, devido aos Bots sofrem constantes mudanças nas formas de agir o que ocasionaria não serem detectados pelas assinaturas, sendo necessário assim a analise por anomalia. No entanto apenas o uso de anomalia não se torna eficaz na detecção dos Bots, pois as regras de assinatura trazem uma maior precisão na detecção, diminuindo assim os falsos positivos.

3.5 Trabalhos Relacionados

As *Botnets* possuem características dinâmicas, por realizarem inúmeras ações determinadas pelo seu controlador, como DDos, coleta de informações, varredura de tráfego, ou seja, os *Bots* podem realizar vários tipos de ataque, o que tornar difícil sua detecção por

IDS tradicionais. A variação de técnicas utilizadas pelos *Bots*, além da estrutura de controle exercido pelo *Botmaster*, possibilita a *Botnet* um alto poder de ataque e uma flexibilidade quanto a sua variação de atividades. Técnicas de detecção de *Botnet* vêm sendo desenvolvidas, destacam-se a seguir trabalhos relevantes no contexto deste projeto.

A ferramenta *Snort*, uma das mais utilizadas atualmente, é uma ferramenta código aberto de sistema de detecção de intrusão (IDS), que monitora o tráfego da rede para encontrar sinais de intrusos, configurado através de um conjunto de regras e assinaturas de log de tráfego, que é considerado suspeito [1, 9, 24, 25]. Sofrem de uma infinidade de problemas, primeiro, como o tráfego de rede e do tamanho de seu banco de dados de assinatura continuam a aumentar, o rendimento se torna um gargalo. Segundo, a detecção de ataques é baseada principalmente em assinaturas conhecidas, tornando esses sistemas inadequados para lidar com ataques desconhecidos, sendo assim, novas *Botnets* precisam da intervenção humana para a criação de assinaturas, para sua detecção. Finalmente, o número significativo de falsos positivos gerados limita seu uso para a prevenção [41].

MABDS [7] é um modelo de IDS que realiza a detecção de *Botnet* através de algoritmos baseados em assinaturas, derivadas da análise dos diferentes tipos de softwares maliciosos que criam *Bots*. Propõe a aplicação do método híbrido, através da análise de sistemas e *Honeypots*, onde realizam a análise de eventos no sistema operacional e no ambiente da rede, utilizando sistemas multi-agente. Assim como no *Snort*, este modelo não detecta os *Bots* que não se encontram no seu banco de assinatura, ou seja, não é aplicado a detecção baseada em anomalia.

Botsniffer [40,41] esta ferramenta utiliza uma metodologia para identificar controladores de *Botnets* baseada na similaridade de tráfego. Emprega um algoritmo de correlação espaço-temporal que atua em *Botnets* de estrutura centralizada e baseadas em protocolos específicos: IRC e HTTP. É aplicável a um conjunto restrito de *Botnets*, portanto, não se aplica a mais recente estrutura de *Botnets* descentralizadas, além de realizar a detecção por assinatura.

BotMiner [42] utiliza técnica de detecção baseada em anomalia através da mineração de dados para detecção de tráfego da *Botnet*. Esta IDS define um agrupamento de comunicações de tráfego em dois níveis: similaridades de tráfego de comunicação e tráfego possivelmente malicioso. Em seguida, ele executa correlação cruzada para identificar os Hosts que compartilham tanto os padrões de comunicação semelhantes e semelhantes padrões de atividade maliciosa. A *BotMiner* identifica características de *Botnets* independente do protocolo e da estrutura utilizada pela rede maliciosa. Esta ferramenta pode detectar *Botnets*

do mundo real, incluindo *Bots* baseada em IRC, em HTTP e P2P com uma baixa taxa de falsos positivos [9]. Esta arquitetura não utiliza a detecção por assinatura. A *BotMiner* utiliza uma *Whitelist* (uma lista branca, ou seja, lista de Ips "amigos" para tráfego livre na rede) para a redução do volume de tráfego monitorado, onde são registrados todos os IPs que possam trafegar normalmente por serem IPs seguros, no entanto, esta característica pode ser tornar um ponto falho da ferramenta, visto que, se tivermos um IP dentro desta lista infectado, seu pacote será ignorado da análise.

BotGAD (Detector de Atividade em Grupo de *Botnet*) [43] essa arquitetura de detecção de *Botnet* combina a consulta de *Bots* e migração de Servidores C&C, que exige a utilização de dados de tráfego DNS no processo de infecção lançamento de ataques e atualização de códigos de ataque. Neste modelo centra-se na propriedade do comportamento dos *Bots*, sendo ideal em grande escala dos dados de tráfego DNS, a partir de sensores implantados na entrada de dados de *Botnets*, geralmente dispersos em diferentes redes, ou seja, espalhados pela Internet. A detecção de *Botnet* baseada em DNS vem ser uma das mais promissoras, pois detecta redes de *Bots* independentemente de sua estrutura, centralizada ou distribuída, no entanto, esta arquitetura não detecta por assinatura.

A Tabela 3.1 apresenta um quadro comparativo com as funcionalidades dos principais IDS apresentados nesta seção, destacando suas principais funcionalidades.

Detecção	**Snort**	**MABDS**	**Botsniffer**	**BotMiner**	**BotGAD**
Baseado em Assinatura	**Sim**	**Sim**	Não	Não	Não
Correlação Horizontal	Não	Não	**Sim**	**Sim**	**Sim**
HIDS	**Sim**	**Sim**	Não	Não	Não
NIDS	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**
Bot criptografado	Não	Não	**Sim**	**Sim**	**Sim**
Botnet Centralizada	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**
Botnet Descentralizada	Não	Não	Não	**Sim**	**Sim**
Bot Individual	**Sim**	**Sim**	Não	Não	Não
Atividade em Grupo	Não	Não	**Sim**	**Sim**	**Sim**

Tabela 3.1 Tabela comparativa de funcionalidade das ferramentas IDS

3.6 Considerações Finais

Este capítulo apresentou uma visão geral Sistemas de Detecção e Intrusos envolvendo sua definição, características, classificação de acordo com critérios de coleta e

análise, destacando as técnicas de PSO e Redes Neurais na aplicação de detecção por anomalia, e recentes pesquisas envolvendo técnicas de detecção com ênfase a ataques de *Bots*. O próximo capítulo apresentará a proposta de uma arquitetura para um sistema de detecção de intrusos com integração ao Modelo NIDIA.

4 Arquitetura Proposta e Integração ao NIDIA

4.1 Introdução

Este capítulo apresenta a proposta de uma arquitetura para um Sistema de Detecção de Intrusos Híbrido, que tem como base a aplicação da tecnologia de Otimização por Enxame de Partícula e Redes Neurais Artificiais. Uma visão geral da arquitetura e do seu funcionamento será apresentada, bem como as principais interações entre os módulos. Ao final do capítulo, a integração da arquitetura proposta ao Sistema NIDIA será apresentada.

4.2 Visão Geral da Arquitetura

A solução proposta foi construída inicialmente com base em diversas arquiteturas disponíveis, citadas no capítulo anterior e posteriormente adaptada ao IDS NIDIA (*Network Intrusion Detection System based on Intelligent Agents*) [30], com a inovação de ampliar a ferramenta através de soluções para os problemas enfrentados no combate à detecção de *Botnets*. Esta arquitetura utiliza um modelo de detecção híbrido, ou seja, emprega as técnicas de assinatura e anomalia.

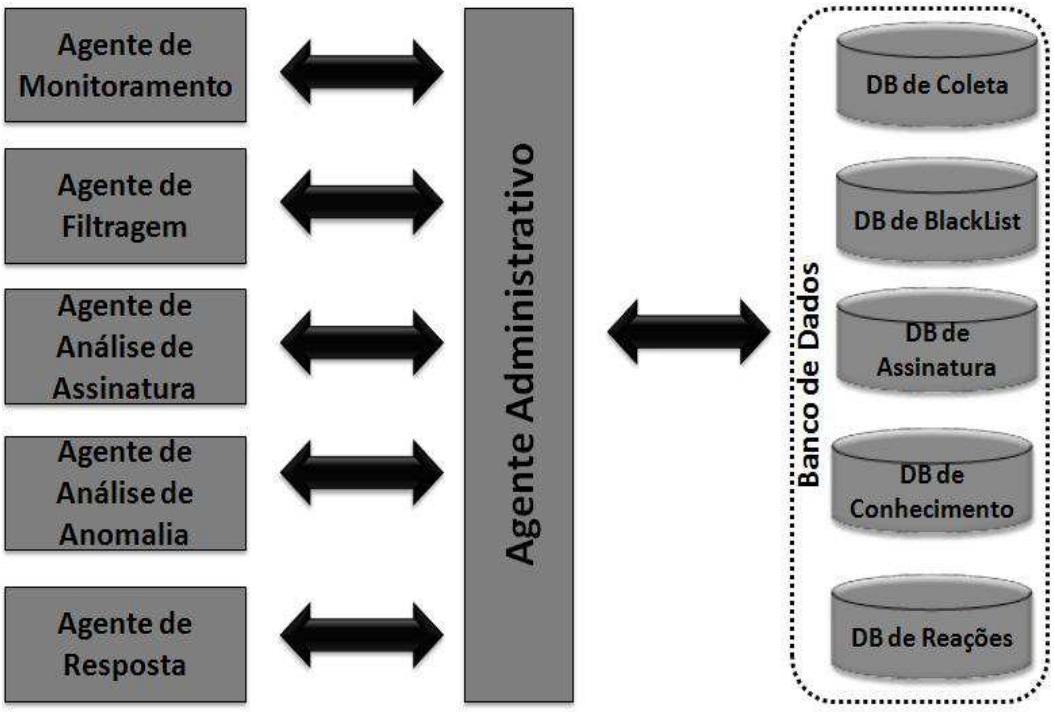


Figura 4.1 Arquitetura Geral do Modelo de um IDS-H

A arquitetura proposta é representada na Figura 4.1. Destaca-se neste trabalho a utilização de um IDS-H para identificar *Botnet* apoiada por um conjunto de componentes que interagem diretamente ou indiretamente para coleta, filtragem e análise de pacotes na rede. O modelo utiliza a filtragem de pacotes através das *Blacklists*, ou seja, faz a verificação de IPs em uma lista “negra”, além de realizar as análises de assinatura e anomalia, para minimizar o grau de falsos positivos.

4.2.1 Agente de Monitoramento

A função deste tipo de objeto é a captura de pacotes na rede. O agente de Monitoramento atua em pontos estratégicos da rede e funciona como sensor de rede passivo, trabalhando em modo promíscuo, desta forma não interferindo no desempenho e nem tráfego da rede. No trabalho proposto, a captura de pacotes se concentra em regiões críticas da rede, conforme demonstrado na Figura 4.2, para que possa identificar o maior volume de pacote e coletam informações importantes sobre seu tráfego. Os dados coletados são armazenados no Banco de Dados de Coleta.

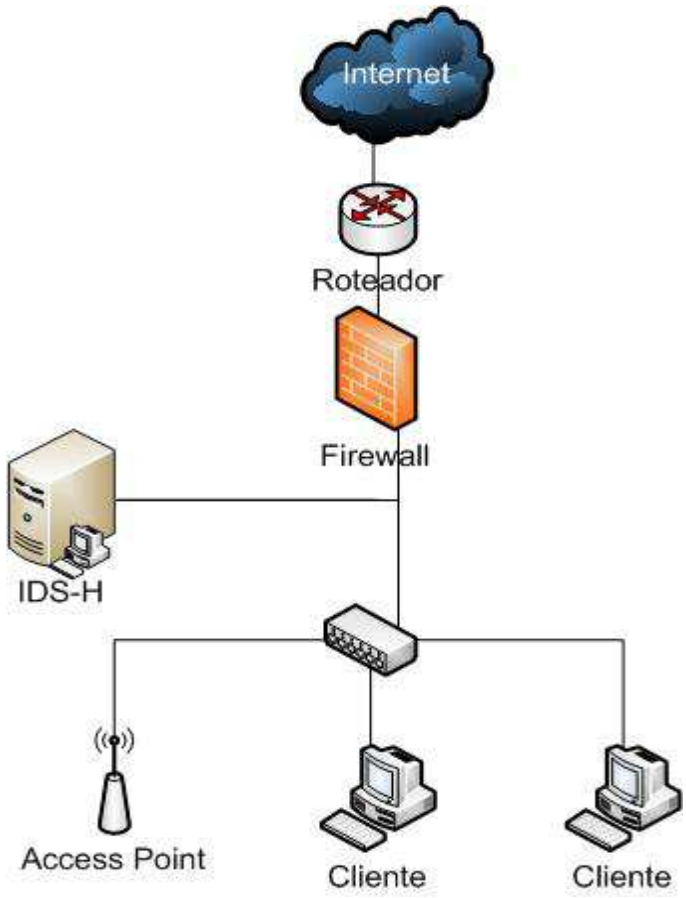


Figura 4.2 Atuação do IDS-H num ponto crítico da rede
Fonte[4]

4.2.2 Agente de Filtragem

Este objeto recebe os dados coletados, para a realização da filtragem de pacotes. Este procedimento reduz o número de pacotes a serem analisados pelos Agentes de Análise, pois elimina pacotes detectados no filtro. O filtro consiste por um banco de dados da lista de IPs considerados suspeitos, denominada *Blacklist* [44].

Uma *Blacklist* é uma base de dados, que contêm os endereços de IP, nomes de domínio, e outros dados que permitem reconhecer se esse endereço é malicioso ou não. E com isso, negar o seu acesso, caso seja identificado como suspeito. É possível também, detectar a origem do pacote e bloqueia-lo, para impedir o seu tráfego na rede [45,46].

Há um grande número de listas disponíveis e sendo compartilhado por diversas comunidades on-line que contém IPs suspeitos. Estas listas são geralmente conhecidas como lista “negra” [47]. Muitas dessas listas são usadas para ajudar a bloquear spam, ataques mal-intencionados, ou os usuários do incômodo. Algumas listas negras são excelentes fontes de informação quando os dados são usados corretamente, mas alguns são tão pobres que qualquer uso deles seria prejudicial à utilização da ferramenta.

Essa *Blacklist* funciona com a consulta DNS a uma base de dados. Uma vez que, o número de IP esteja em uma lista negra, qualquer filtro que venha a ser utilizado nessa lista irá banir o endereço do servidor. Sua principal função é proibir o acesso de alguns servidores da Internet que tenham sido identificados como maliciosos [48]. Então, todo endereço proveniente de um IP na lista negra, é rejeitado logo que a conexão é estabelecida.

Para que possa ter um bom funcionamento uma *Blacklist* deve ter as seguintes características:

- Integridade - A *Blacklist* deve conter informações atualizadas sobre número de endereços IP, para que não haja problemas de falsos positivos na verificação da lista.
- Receptividade - A *Blacklist* deve ter um baixo tempo de resposta para que os outros destinatários possam posteriormente ter acesso aos endereços IP. [2]

Essas listas podem ser construídas com a utilização de técnicas ou algoritmos de análise e rastreamento, pois existem diferentes implementações de listas de bloqueio que podem ser usadas para fazer filtragens específicas. O problema de uma *Blacklist* é mantê-la atualizada, já que umas grandes quantidades de registros de domínios suspeitos surgem a cada momento.

Nesta etapa do IDS-H os pacotes capturados são verificados no filtro onde são comparados os endereços IPs da lista negra. Caso seja detectado é encaminhado à detecção do intruso para o Agente de Resposta, ou seja, são encaminhados os pacotes contendo as informações do intruso encontradas. Os pacotes não identificados na filtragem são encaminhados para futura análise nos Agentes de Análise por Assinatura.

4.2.3 Agente de Análise de Assinatura

Nesta etapa o agente de analise de assinatura é responsável pela análise dos pacotes recebidos e não detectado como intruso após o processo de filtragem. Os pacotes coletados são formatados de maneira que padrões de ataques possam ser identificados e posteriormente confirmar o ataque. Para isso, utilizam-se a base de dados de assinaturas de intrusões. Estas assinaturas são compostas de regras para detecção de ataques de *Botnet*.

As *Botnets* mais prevalentes são baseadas em protocolo *Internet Relay Chat* (IRC) [1, 2, 3, 7, 9,10], como mecanismo de comando e controle. O Protocolo IRC foi originalmente concebido para as grandes salas de chat social para permitir várias formas de comunicação e divulgação de dados entre grande número de *Hosts* [3]. A grande prevalência de *Botnets* baseado IRC é devido à inerente flexibilidade e escalabilidade do protocolo, este tipo de protocolo geralmente utiliza portas 6667, 6668, 6669 e 7000. Estas características contribuem para a construção de um banco de assinatura, neste banco estão contidas regras coletadas de informações relevantes de cada tipo de *Botnet*, pelas quais são comparadas com características dos pacotes para a identificação do ataque ocorrente.

Outra fonte de assinatura, que auxiliaram a construção do banco de dados, foi à utilização de *Sandboxes*, ferramentas utilizadas para análise dinâmica de arquivos binários [49], caracterizada pela execução de um arquivo num ambiente controlado registrando em forma de relatório as ações realizadas pelos *malware* [50].

As principais funcionalidades observadas pelos *Sandboxes* descrevem características como: arquivos criados ou modificados durante a execução do arquivo, monitoração de acesso ou modificações à chave de registros do sistema, conexões de rede; dados transmitidos, entre outros [41].

Entre as ferramentas *Sandboxes*, disponíveis no mercado, grande parte disponibiliza uma interface sem restrição de acesso para a avaliação de resultados. Essas ferramentas apresentam um relatório com as ações desempenhas e mapeadas pelo sistema em

questão em relação ao arquivo submetido, o que possibilita a coleta de informações sobre as *Botnets* e outros *malwares*.

Após a execução da análise por assinatura os pacotes detectados são bloqueados e notificados para o Agente de Reação, conforme Figura 4.1. Os pacotes são encaminhados para uma posterior análise pelo Agente de Análise de Anomalia, visto que, faz-se necessário o uso de todos os pacotes na técnica de anomalia, para verificação maior sobre a intrusão.

4.2.4 Agente de Análise de Anomalia

O agente de análise de anomalia recebe os pacotes não filtrados pelo Agente de Filtragem, estes pacotes serão analisados para detecção de intrusos segundo comportamento anômalo. Com base nos pacotes coletados, a execução da análise de anomalia é determinada pela comparação do perfil padrão dos *Hosts* obtido em trafego normal e do perfil sob ataque.

Para que o sistema possa ser capaz de determinar com precisão o diagnóstico de detecção de *Botnets*, é utilizado o processo de treinamento para verificar as grandezas dos perfis dos *Hosts*, ou seja, é definido um perfil padrão dos *Hosts* em um funcionamento normal da rede e em funcionamento na ocorrência de um ataque, essas grandezas serão utilizadas para definir o estado de anomalia. Para determinar características em comum de dois ou mais *Host*, durante um intervalo de tempo, visto que, Clientes *Bots*, geralmente realizam atividades de grupo, o treinamento é executado utilizando as técnicas por Otimização por Enxame de Partículas e Redes Neurais Artificiais. Quando os perfis são definidos, a avaliação do agente de anomalia ocorre comparando os tráfegos da rede com as situações de tráfego normal ou sob ataque.

Após a detecção de intrusos são encaminhadas notificações para o Agente de Resposta, para serem tomadas medidas de prevenção e proteção da rede.

4.2.5 Agente de Resposta

O presente objeto é responsável por tomar contramedidas caso um incidente de segurança seja detectado. Com base na avaliação da *Blacklist*, do Agente de Análise de Assinatura e do Agente de Análise de Anomalia.

As medidas tomadas podem reagir contra ataques passivamente, ou seja, têm reação passiva e simplesmente informa ao administrador de um evento malicioso, sem nenhuma represália. Neste tipo de reação, a questão mais importante é a velocidade de

notificação quando os ataques ocorrem na rede [4]. O IDS-H também gera uma reação ativa quando ocorrem ataques e resposta a situações críticas. As contramedidas ativas ocorrem através do bloqueio do sinal de um invasor ou na criação de arquivo log para o *Firewall*, elas são realizadas após a notificação de um intruso.

4.2.6 Agente Administrativo

O agente administrativo integra todos os agentes do IDS-H. Ele é responsável pela atualização das bases de informações.

As consultas poderão ser feitas diretamente de qualquer camada, porém inserções devem ser feitas somente através desta camada. Ela tem a responsabilidade de manter a integridade e consistência das informações armazenadas no Banco de Dados.

4.2.7 Banco de Dados

O banco é responsável por manter de forma persistente informações provenientes de cada agente. Neste localizam-se as bases de dados utilizadas pelo IDS-H. Segue uma descrição das mesmas:

- Banco de Dados de Coleta: registra os pacotes coletados durante a captura de tráfego da rede;
- Banco de Dados de *Blacklist*: nele está contido os dados de IPs suspeitos. Os seus dados são exportados de *Blacklists* existentes;
- Banco de Dados de Assinatura: é à base de dados responsável por armazenar todas as informações referentes às assinaturas de *Botnets*;
- Banco de Dados de Conhecimento: registra as informações sobre os *Hosts* existentes na rede, detectado na fase de treinamento para a geração de um perfil padrão, a fim de identificar que dispositivos estão sob suspeita de ataques ou se tornaram possíveis *Bots*;
- Banco de Dados de Reações: nesta estão contidas as informações referentes às ações que devem ser tomadas de acordo com as intrusões detectadas. É adaptada de acordo com a política de cada organização.

4.3 Aplicabilidade da Arquitetura ao NIDIA

O Projeto NIDIA (*Network Intrusion Detection System based on Intelligent Agents*) foi criado em 2001 [52], na Universidade Federal do Maranhão – UFMA tem o objetivo de fornecer uma contribuição para a melhoria das técnicas de detecção de intrusos em redes de computadores, permitindo que vários pesquisadores, incluindo-se professores, alunos da graduação e pós-graduação, tenham a oportunidade de conhecer, trabalhar e disseminar as novas tecnologias das áreas de segurança de redes de computadores, inteligência artificial e sistemas multiagentes.

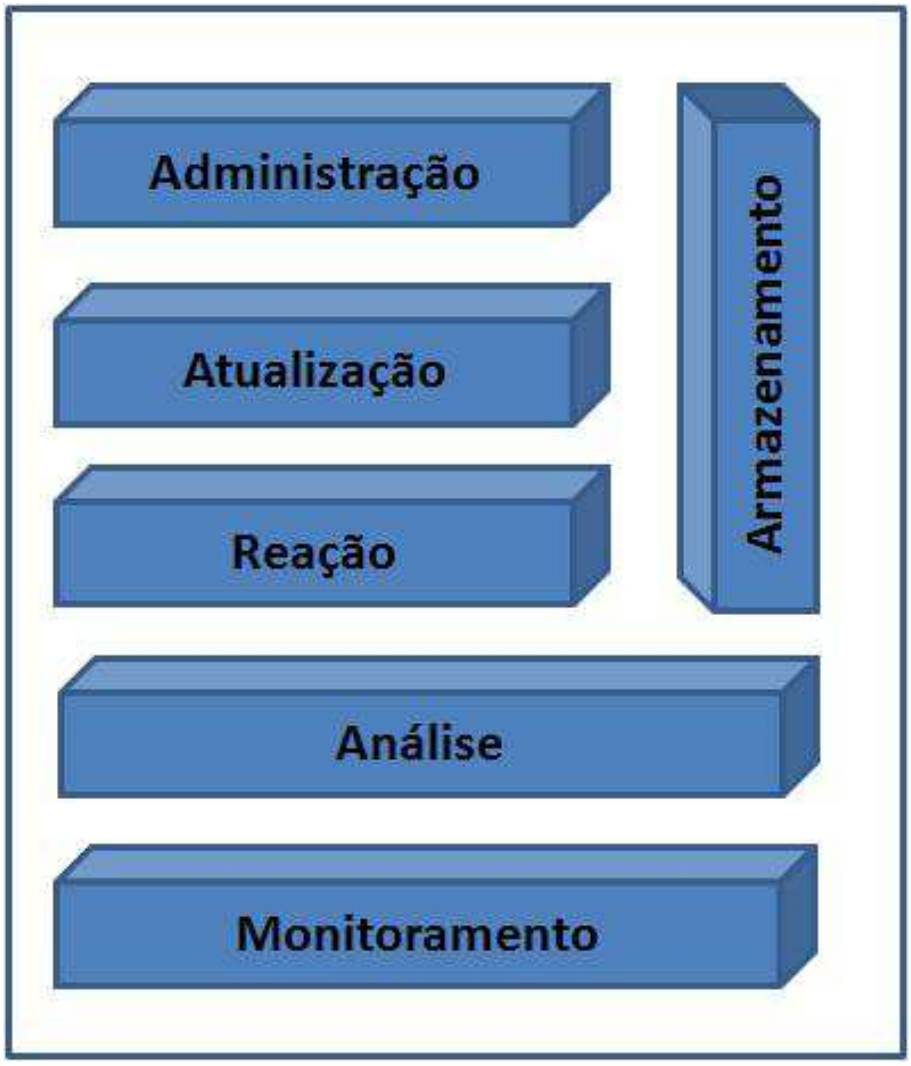


Figura 4.3 Modelo em camadas do NIDIA
Fonte[55]

Este projeto é inspirado no modelo CIDF (*Common Intrusion Detection Framework*) [53], possuindo para esta finalidade agentes com a função de geradores de eventos (agentes sensores), mecanismos de análise dos dados (agentes de monitoramento e de

avaliação de segurança), mecanismos de armazenamento histórico (base de dados) e um módulo para realização de contramedidas (agente controlador de ações). Além disso, existem agentes responsáveis pela integridade do sistema e pela coordenação das atividades do IDS como um todo.

A arquitetura do NIDIA é composta por camadas [55], apresentado na Figura 4.3, sendo que, cada camada possui atividades a desempenhar, onde estas atividades são executadas através do comportamento dos agentes que a compõe. É através destes agentes também que as camadas se comunicam trocando informações importantes para desempenhar suas atividades.

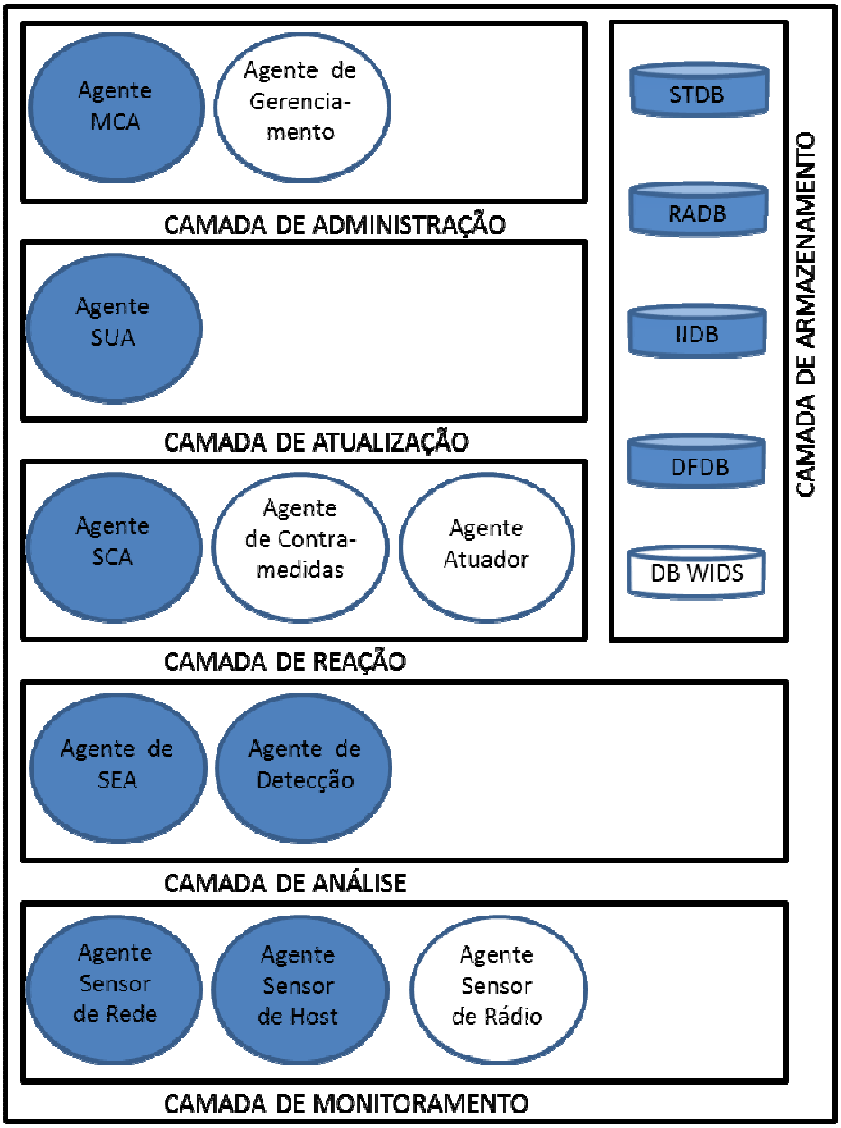


Figura 4.4 Arquitetura do NIDIA com integração para Detecção de Intrusos em Ambiente Wireless Fonte[30]

A utilização da arquitetura de multiagentes [54] para o IDS-NIDIA foi definida pelos seguintes fatores:

- Agentes podem ser adicionados ou removidos para/do sistema sem modificar outros componentes do sistema;
- Agentes podem ser reconfigurados ou atualizados sem causar problemas ao restante do sistema;
- Um agente ou um grupo de agentes pode realizar diferentes funções simples. Visto que os agentes podem trocar informações entre si, podem derivar resultados mais complexos. O objetivo específico de cada agente que compõe o IDS-NIDIA é demonstrado na Figura 4.3 através da sua arquitetura em camadas.

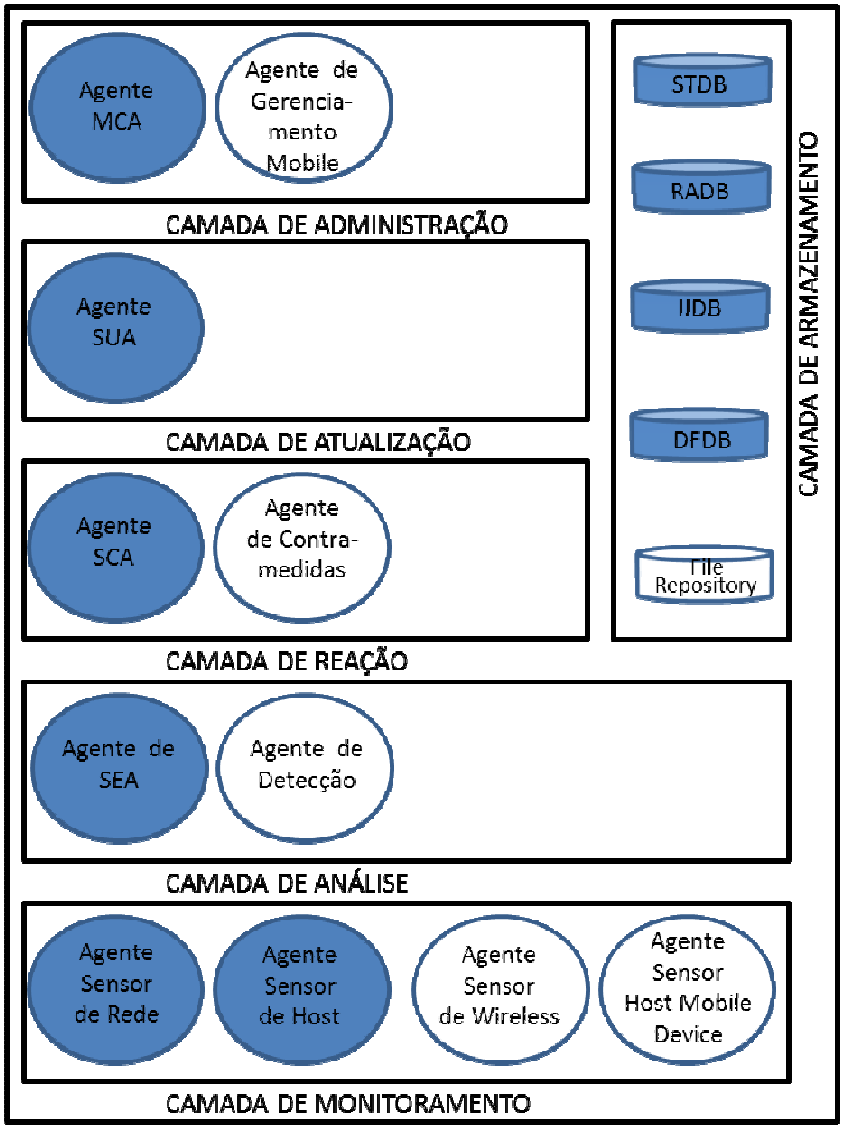


Figura 4.5 Arquitetura do NIDIA com integração para Detecção em Dispositivos Móveis Fonte[56]

O NIDIA vem sofrendo constantes mudanças ao longo dos anos, através de inovações integradas ao seu projeto. No ano de 2007, foram desenvolvidos agentes de

Detecção de Intrusos no Ambiente *Wireless* para integração ao projeto de pesquisa, ampliando significativamente as detecções em redes, visto que, até então o projeto não realizava detecção em redes sem fios[30], descrito na Figura 4.4. Nesta etapa do NIDIA foram implementado o Agente de Gerenciamento, Agente de Contramedidas, Agente Atuador, Agente Sensor de Rádio e Banco de Dados WIDS. As camadas e agentes serão discutidos a partir do item 4.3.1, neste capítulo.

Em 2008, foi incorporado ao NIDIA aplicações de Detecção em Dispositivos Móveis [56], demonstrado na Figura 4.5, nesta arquitetura foram integrados os seguintes componentes: Agente Gerenciamento *Mobile*, Agente de Contramedidas, Agente de Detecção, Agente Sensor de *Wireless*, Agente Sensor *Host Mobile Device* e *File Repository*.

A proposta deste trabalho apresenta novas funcionalidades ao conjunto de agentes do NIDIA, fornecendo um modelo de detecção de *Botnet*, em tempo real, baseado na noção de sociedade de agentes capaz de detectar novos ataques, além de definir uma nova arquitetura integrando os agentes criados por trabalhos anteriores, com os novos agentes.

4.3.1 Camada de Monitoramento

Esta camada é responsável por capturar a ocorrência de eventos no meio exterior e fornecer informações sobre o mesmo para o resto do sistema. Nesta camada, os agentes SMA (*System Monitoring Agent*) estão localizados. Estes agentes funcionam como “sentidos receptores” do sistema. Os dados obtidos recebem uma pré-formatação sendo em seguida repassado para o agente de avaliação de segurança. Os agentes SMAs dividem-se em três categorias:

- Agentes sensores de rede: responsáveis por capturar os pacotes que estão trafegando na rede. Estes atuam em pontos estratégicos da rede e funcionam como monitores de rede passivo, trabalhando em modo promíscuo, ou seja, sem que influencie nas informações que estão sendo transmitidas, desta forma não interferindo no desempenho e nem no tráfego da rede;
- Agentes sensores de *Host*: trabalham coletando informações em tempo real de um *Host* em particular (geralmente servidores) e disponibilizando-as para análise;
- Agentes sensores de rádio: opera com a captura no ambiente de comunicação sem fio, este sensor trabalha em modo promíscuo [30];

- Agente Sensor *Host Mobile Device*: realiza a captura das informações dos dispositivos móveis [56].

4.3.2 Camada de Análise

Nesta camada é realizado à análise dos eventos recebidos da camada de monitoramento. Os eventos coletados são formatados de maneira que padrões de ataques possam ser identificados e posteriormente a confirmação de um ataque. Para isso, utilizam-se bases de conhecimento, como a base de dados de padrões de intrusões (IIDB, *Incidents of Intrusion and Forensic Information DataBase*), a base de dados de incidentes de intrusão (DFDB, *Standard of Intruders and Intrusion DataBase*), a base dados de estratégias (STDB, *Strategy DataBase*), a base de dados de redes sem fios (BDWIDS, *Wireless Intrusion Detection System DataBase*) e a base de dados de dispositivos móveis (*File Repository*).

Na camada de análise, localizam-se os agentes SEA (*Security Evaluation Agent*) são responsáveis por realizar a análise dos eventos coletados e emitir uma notificação de intruso sobre os eventos que foram previamente formatados. Nesta camada, o agente de Detecção, que realiza a detecção de intrusões ocorridas no meio wireless, dos dispositivos móveis e redes cabeadas, através dos comportamentos anormais dos dispositivos monitorados.

4.3.3 Camada de Reação

A camada de reação tem como finalidade tomar contramedidas caso um incidente de segurança seja detectado. Com base no parecer do SEA, esta camada deve realizar uma contramedida de acordo com as bases de dados de estratégia (STDB) e de ações (RADB, *Reaction DataBase*).

Nesta camada localizam-se os seguintes agentes:

- Agente SCA (*System Controller Agent*) que tem por finalidade realizar ações sobre os eventos que foram previamente notificados. Para isso, utiliza bases de conhecimento, como a base de dados de intrusos e intrusões (IIDB), a base de dados de incidentes de intrusão e informação forense (DFDB) e a base de estratégias (STDB);
- Agente de Contra-Medidas (CMA, *Counter-Measure Agent*) tem a função de decidir que contramedidas devem ser tomadas, no intuito de conter em tempo

real qualquer intrusão ocorrida no ambiente monitorada. Essas contramedidas são mapeadas em ações, que são enviadas para os Módulos Atuadores [30];

- Agente Atuador (AA, *Actuator Agent*) é responsável por executar as ações determinadas pelo Módulo de Contramedidas, no intuito de conter as atividades de intrusão acontecendo na rede monitorada. A execução dessas ações se dá, na maioria dos casos, com o Módulo Atuador injetando tráfego ativamente na rede.

4.3.4 Camada de Atualização

Esta camada é responsável pela atualização das bases de informações, onde verifica as informações contidas nas bases de dados e faz suas atualizações.

As consultas poderão ser feitas diretamente de qualquer camada, porém inserções devem ser feitas somente através desta camada. Ela terá deve manter a integridade, confiabilidade e consistência das informações armazenadas.

Nesta camada, localizam-se os agentes SUA (*System Updating Agent*), responsáveis pela atualização das bases DFDB, IIDB, RADB, STDB BD WIDS e *File Repository*.

4.3.5 Camada de Administração

A camada de administração tem como finalidade administrar e determinar a integridade de todos os agentes do sistema. Nesta camada, localizam-se os agentes MCA (*Main Controller Agent*).

O agente de Gerenciamento está contido no NIDIA na camada de Administração, juntamente com o agente MCA (*Main Controller Agent*). Isso agrega ao NIDIA a capacidade de gerenciamento por parte do administrador do sistema, com uma interface gráfica rica em opções de configuração, monitoramento, análise e geração de relatórios.

4.3.6 Camada de Armazenamento

Esta camada é responsável por manter de forma persistente informações provenientes das demais camadas. Nesta camada, localizam-se as bases de dados utilizadas pelo NIDIA. Segue uma descrição das mesmas:

- STDB (*Strategy DataBase*) é à base de dado responsável por registrar as estratégias adotadas por uma organização qualquer em relação à sua política de segurança. Ela é importante para garantir a adaptabilidade do IDS em diversos casos;
- RADB (*Reaction DataBase*) estão contidas as informações referentes às ações que devem ser tomadas de acordo com a severidade do ataque detectado. Também varia de acordo com a política de cada instituição;
- IIDB (*Incidents of Intrusion and Forensic Information DataBase*) registra os danos causados por ataques bem-sucedidos e tentativas de ataques. Este contém informações que podem ser úteis na identificação de tentativas de ataques provenientes de uma mesma origem de uma mesma origem ou domínio ou simplesmente serem usadas em investigações futuras;
- DB WIDS (*Wireless Intrusion Detection System DataBase*) é um banco de dados completo com informações detalhadas sobre toda a atividade das comunicações na rede sem fio;
- *File Repository* é um deposito de dados das informações sobre os dispositivos móveis.

4.4 Integração do Modelo Proposto ao NIDIA

A solução proposta amplia o poder de detecção do Projeto NIDIA através da integração entre os dois modelos. Essa integração se torna facilitada devida as duas arquiteturas serem estruturadas em camadas, possibilitando a adaptação dos componentes do IDS-H aos agentes do modelo NIDIA. Para tanto, foram inseridos novos agentes e bancos de dados do projeto proposto à arquitetura do NIDIA, conforme a Figura 4.6. Os elementos destacados em azul são determinados na arquitetura NIDIA como agentes e bancos de dados. Nesta nova proposta, novos elementos são inseridos que são representados pela cor branca. Os elementos em vermelho fazem parte dos quais foram integrados.

Na camada de monitoramento o IDS-H utiliza um sistema de coleta de pacotes baseada em agente sensor de rede e sensor de rádio, possibilitando a coleta dos dados em nível de rede cabeada, como em rede sem fios. Estes agentes substituem a figura do Agente de Monitoramento, descrito no IDS-H, visto que realizam a mesma função.

Os agentes de filtragem e o de detecção de *Botnet* são inseridos na camada de análise juntamente com o agente SEA (*System Evaluation Agent*) e o agente de Detecção. No agente de Detecção, do modelo NIDIA, foram implementadas funcionalidades do agente de Análise de Assinatura, neste agente são inseridos as assinaturas para detecção de Botnet. O agente de Filtragem tem por objetivo minimizar o volume do tráfego a ser analisado pelos demais agentes desta camada, através da detecção de pacotes suspeitos. Enquanto que, o agente de Detecção de *Botnet* amplia o poder de detecção do NIDIA através da análise de detecção de *Botnet* utilizando a técnica de anomalia, através das verificações utilizando técnicas de Inteligência Artificial.

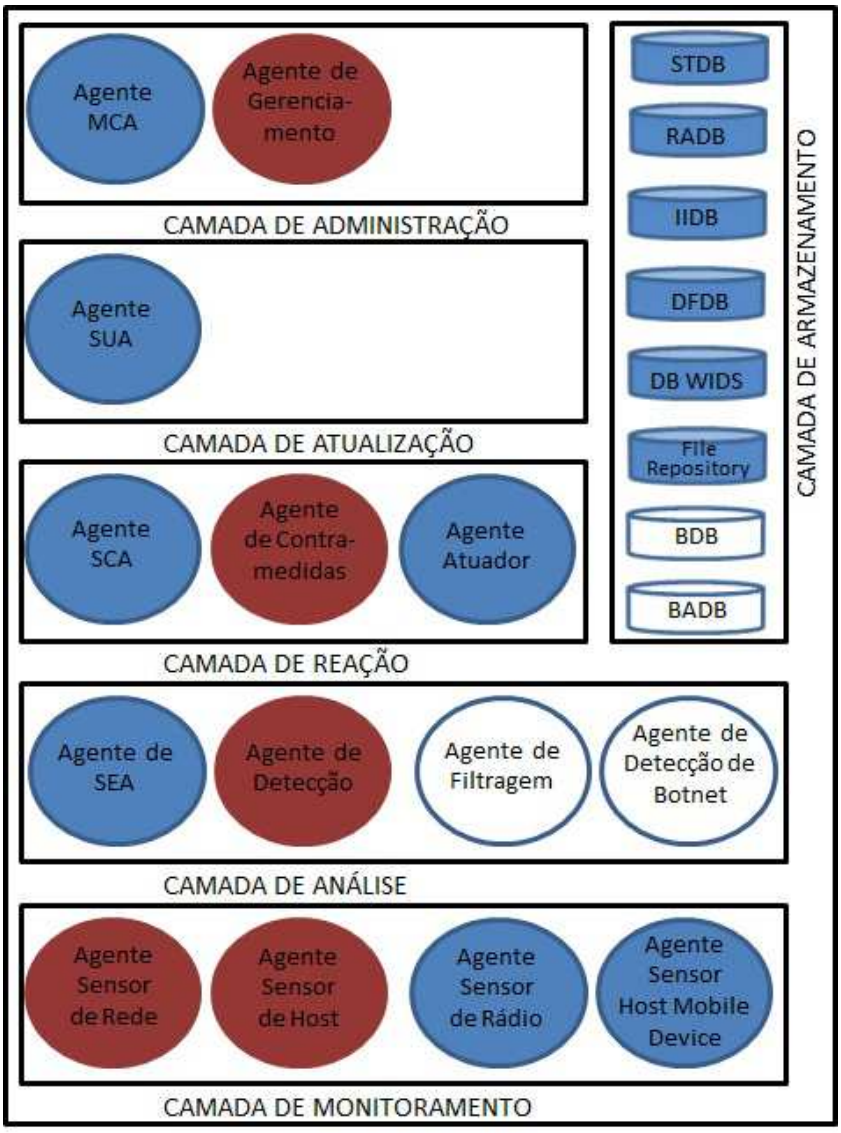


Figura 4.6 A integração do Modelo de IDS-H ao NIDIA

Na camada de reação, a utilização do Agente de Contra-Medidas do NIDIA possui a mesma finalidade do Agente de Reação proposto na arquitetura do IDS-H. Este agente age com medidas ativas e passivas aos ataques, visto que, o objetivo dele é responder aos eventos de intrusos contra a rede e seus dispositivos. Diante de sua funcionalidade, faz-se necessário a correlação entre os ataques de *Bots* e as reações existentes, além da criação de novas medidas de defesa.

O Agente Administrativo coordena as atividades do IDS-H pela integração das camadas dessa arquitetura através da interface gráfica com o usuário. Na camada de administração do Modelo NIDIA, estas ações são operadas de acordo com o Agente de Gerenciamento, sendo assim, novas funcionalidades do IDS-H foram incorporadas a este agente, como a interface de análise, de monitoramento, de inserção de dados e relatórios voltados para detecção de *Botnets*.

Na camada de armazenamento são inseridos o banco de dados de *BlackList* e o banco de dados de anomalia de *Botnets*, juntamente com os bancos de dados existente no NIDIA: STBD (*Strategy Database*), RADB (*Reaction Database*), IIDB (*Incidents of Intrusion and Forensic Information Database*), DFDB (*Standard of Intruders and Intrusions Database*), DB WIDS(*Wireless Intrusion Detection System DataBase*) e *Repository File*. As assinaturas de Bots contidas no banco de dados de assinatura do modelo proposto, são inseridas no IIDB, na qual guarda as assinaturas de intrusão do NIDIA. Quanto o banco de dados da *BlackList* (BDB, *Blacklist Database*) é do depósito de dados referentes aos IPs suspeitos de intrusos. Enquanto que, o banco de anomalia de *Botnet* (BADB, *Botnet Anomaly Database*) restringe-se a conter os dados das grandezas das ações por anomalia de *Bot*. Isso traz para o NIDIA um banco de dados completo com informações detalhadas sobre toda a atividade das comunicações das *Botnets*.

4.5 Considerações Finais

Neste capítulo, apresentou-se a proposta de uma arquitetura para um sistema de detecção de intrusos híbrido, que tem como base a aplicação da tecnologia de Otimização por Enxame de Partículas no processo de detecção de intrusões. Finalmente na seção IDS-NIDIA, apresentou-se a proposta do sistema, sua arquitetura e o seu funcionamento, bem como a integração entre os modelos. O próximo capítulo apresentará a implementação de um protótipo e os resultados obtidos na validação da ferramenta.

5 IMPLEMENTAÇÕES E RESULTADOS PARCIAIS

5.1 Introdução

Neste capítulo é apresentado a implementação de um protótipo para o modelo proposto na pesquisa. Para tanto, foram implementados os mecanismos de detecção de intrusos híbrido, usando técnicas de Redes Neurais e Otimização por Enxame de Partícula para realizar a análise de anomalia da ferramenta.

Para finalizar é apresentada a avaliação da solução proposta quando aplicada a detecção de *Botnets*. Para que esse mecanismo de detecção de intruso pudesse funcionar, foi realizado um treinamento com registros de conexões normais e conexões sob ataques de *Botnet* tanto em ambiente simulado, quando em ambiente da rede da Universidade Federal do Maranhão. Os dados obtidos com essa implantação são discutidos no decorrer do capítulo e servirão de base para avaliar a eficiência do sistema.

5.2 Implementação da Solução

A arquitetura proposta tem como base a adaptação e extensão do IDS NIDIA. A solução emprega a detecção de *Botnet* por uso estratégico de análise por assinaturas e anomalias, visto que esta última técnica emprega conceitos de Otimização por Enxame de Partículas para identificar comportamentos anormais dos *Hosts* na rede. Este monitoramento baseia-se em informações coletadas do tráfego de rede capturados por modo promiscuo.

A implementação do IDS-H é descrita através dos diagrama de classe e diagrama de atividades.

5.2.1 Diagrama de Classe

O diagrama de classe é um dos tipos de diagramas mais fundamentais utilizados em UML. Ele permite visualizar as classes contidas no sistema com seus respectivos métodos e atributos, bem como essas classes se relacionam no diagrama [57, 58]. As classes geralmente possuem atributos, que armazenam os dados dos objetos da classe, além de métodos, também denominados de operações, que são as funções que uma instância da classe pode executar [58].

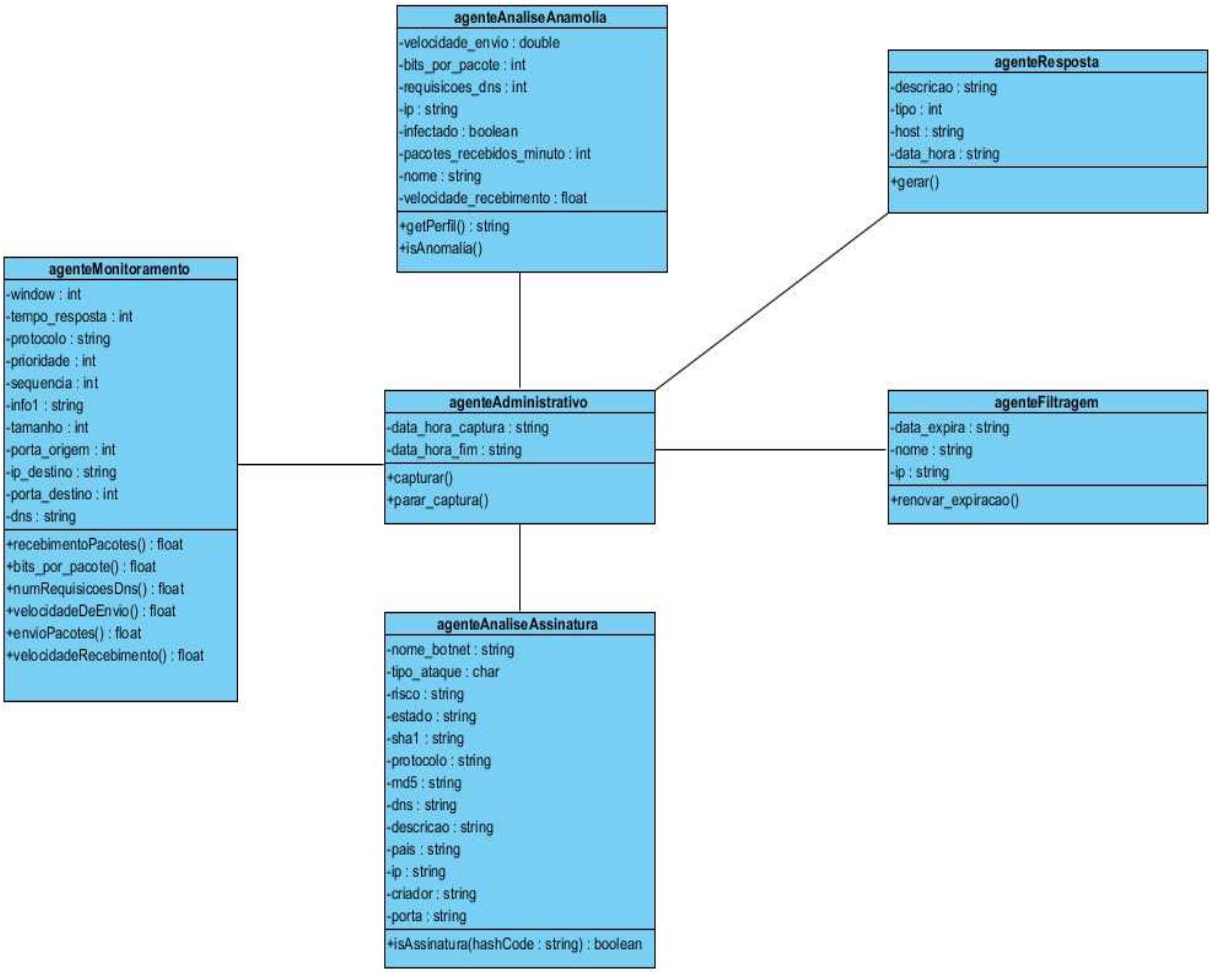


Figura 5.1 Diagrama de Classe do IDS-H

A ferramenta desenvolvida é descrita no diagrama de classe na Figura 5.1. O diagrama descreve as classes do sistema através de seus respectivos relacionamentos.

A classe agenteMonitoramento define os objetos que representam os pacotes capturados na rede. Esta classe utiliza os seguintes métodos, que são utilizados posteriormente para a realização de treinamento e detecção de intrusos:

- velocidaDeEnvio: Calcula a velocidade de envio de pacotes em bps de um Host;
- recebimentoPacotes: Neste método é calculado a taxa de pacotes recebidos por minuto de um Host;
- envioPacotes: Este método calcula a velocidade de pacotes enviados por minuto de um Host;
- velocidadeRecebimento: Determina a velocidade de recebimento de pacotes em bps de um Host;

- numRequisicoesDns: O método faz o calculo do número de requisições de DNS de um *Host*;
- bits_por_pacotes: Neste método é verificado o número de bits por pacotes enviados ou encaminhados por um *Host*.

A Classe agenteAdministrativo utiliza o método *capturar* para encaminhar as informações aos demais agentes.

A Classe agenteAssinatura é responsável pela agrupamento das informações referentes as assinaturas que determinam a identificação dos ataques de *Botnets*. Nesta classe é definido o método *isAssinatura*, esse método verifica se um determinado pacotes pode ser um *Botnet* através de suas assinaturas. Enquanto que a Classe agenteAnomalia determina a identificação dos Hosts da rede. Nesta classe é encontrado o método *getPerfil*, que retorna o perfil do Host criado na execução da fase de treinamento utilizando o PSO e o método *isAnomalia*, utilizado para identificação das anomalias.

Na Classe agenteResposta são realizados a identificação dos relatórios realizados na varredura de pacotes, determinando que tipo de ataque foi identificado durante o processo de análise.

5.2.2 Diagrama de Atividades

O diagrama de atividade é projetado para ser uma visão simplificada do que acontece durante uma operação ou processo dentre de um sistema descrito [57]. A modelagem de atividades centra-se sobre a execução e fluxo do comportamento de um sistema. Quando usado para a modelagem de software, as atividades normalmente representam um comportamento invocado como resultado da chamada de um método. Quando usado para modelagem de negócios, as atividades podem ser desencadeadas por eventos externos, tais como uma ordem a ser colocada das ações desenvolvidas [58, 59].

No protótipo desenvolvido o diagrama de atividades é representado na Figura 5.2. O fluxo de atividades para o Sistema de Detecção de Intruso Híbrido inicia-se na captura de pacotes coletados pelo agente de monitoramento, após sua captura, estes pacotes são encaminhados para a fase seguinte.

O agente de filtragem ao receber os pacotes, realiza a filtragem dos mesmos segundo informações do banco de dados de *BlackList*, após a realização deste filtro, encontra nesta fase o ponto de decisão para as condições a seguir:

- Dados filtrados, ou seja, os dados sobre pacotes detectados serão armazenadas como ataques detectados para notificação do agente de reação, para realizar medidas de ação, finalizando o processo;
- Os pacotes sem detecção pelo filtro são encaminhados para o agente de assinatura, para serem analisados.

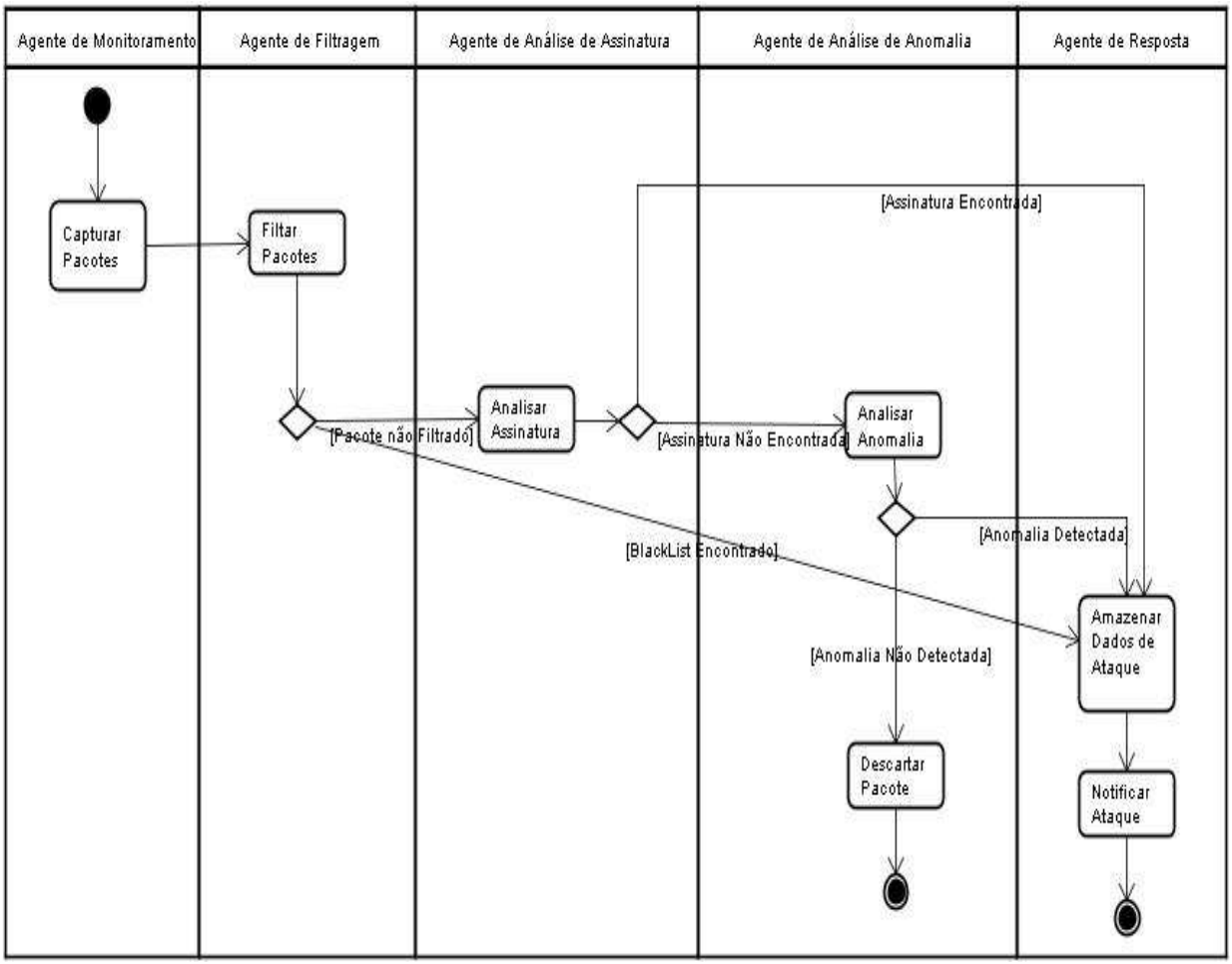


Figura 5.2 Diagrama de Atividades do IDS-H

Na fase de Análise por Assinatura, o agente verifica se alguma das assinaturas de ataques de *Botnet* foram identificadas, a partir de então será decidida ação segundo ponto de decisão. Caso uma assinatura seja encontrada é encaminhado notificação ao agente de reação, onde o mesmo tomará as medidas segundo ataque existente, finalizando assim o processo existente. Caso contrário, o agente de assinatura não detecte o ataque nos pacotes, esses serão enviados para o agente de análise por anomalia.

Na etapa seguinte, o agente de anomalia avalia todos os pacotes recebidos, segundo medidas de anomalia. Após análise será tomado um outro ponto de decisão. Se alguma anomalia for detectada, é tomada medidas de notificação ao agente de reação para

tomar as devidas ações e estabelecido o fim do processo. Se não for encontrado nenhuma anomalia os pacotes são descartados e processo será finalizado.

5.3 Protótipo

O protótipo do IDS-H possui uma interface simples para o usuário, observado na Figura 5.3. Os componentes construído no modelo foram desenvolvidos em *Java* com tecnologia para desenvolvimento Web [60], além de apresentar as características de portabilidade em várias plataformas de sistemas operacionais.

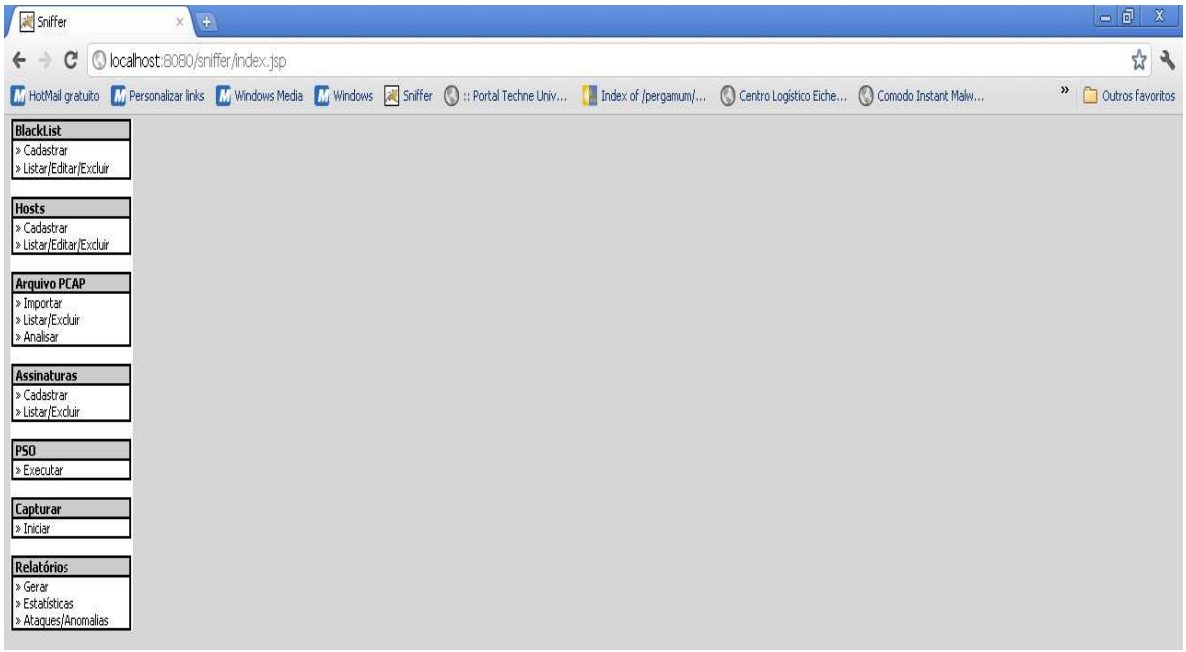


Figura 5.3 Tela Inicial do IDS-H

Foram utilizadas também as bibliotecas *Winpcap* [61] como componente para análise de rede e captura de pacotes para plataforma Windows. A interface de programação *Winpcap* pode ser usado por muitos tipos de ferramentas de rede para análise, solução de problemas de segurança e monitoramento de redes. Dentre essas ferramentas podemos destacar:

- Analisadores de protocolo e redes;
- Monitores de rede;
- Geradores de tráfego;
- Sistemas de detecção de intrusão em rede (NIDS);
- Scanners de rede;
- Ferramentas de segurança.

Para a criação do banco de dados foi utilizado o *SQL Server*[62]. Este gerenciador de banco de dados, foi desenvolvido pela empresa *Microsoft*, fornece uma plataforma de dados confiável, produtiva e inteligente que permite a execução de aplicações de missão crítica mais exigentes, reduza o tempo e o custo com o desenvolvimento e o gerenciamento de aplicações.

O IDS-H é composta pelas seguintes funcionalidades:

- *BlackList*: controle de filtro de IP maliciosos de *Botnet*;
- Arquivo PCAP: controle e avaliação dos arquivos PCAP;
- Assinaturas: controle das assinaturas de *Bots*;
- Detecção de anomalias: execução do treinamento para determinação do perfil dos *Hosts*;
- Capturar: coleta de pacotes de rede e análise de identificação de ameaças que possam ocorrer através da mesma;
- Relatórios: geração de relatórios detecção de ataques.

5.3.1 BlackList

Esta lista de IPs suspeitos é coletada através da Fundação *Shadowserver* [44], conforme Quadro 5.1, na lista são obtido os IPs notificados como comprometidos em atividades de intrusão. Dentre os serviços disponíveis no meio de coleta dessas "listas negras de IPs" encontramos a *Infiltrated.net* [63], *Lashback* [64], *Spamlinks* [65] e *Mainsleaze Spam* [66], são tratadas para serem coletadas as informações necessárias e exportadas para a ferramenta, onde são armazenados no Banco de Dados de *Blacklist*, para serem utilizadas pelo Agente de Filtragem.

8.10.3.0/24 8.10.3.0 - 8.10.3.255 acceleratebiz.com 1001 Brickell Bay Dr. group #235
12.183.57.0/26 12.183.57.0 - 12.183.57.63 TRENTON TV CABLE COMPANY TRENTON-69-57-0 (NET-12-183-57-0-1)
12.183.101.0/24 12.183.101.0 - 12.183.101.255 MondoMailerz Lewiston NY group #350
12.210.202.0/24 12.210.202.0 - 12.210.202.255 USA ATT insightbb spam
24.0.0.0/12 24.0.0.0 - 24.15.255.255 USA Comcast EASTERNSHORE-1
24.48.0.0/14 24.48.0.0 - 24.51.255.255 Adelphia Cable
24.60.0.0/14 24.60.0.0 - 24.63.255.255 USA Comcast Cable Communications Holdings, Inc. BOSTON-6

24.64.0.0/13	24.64.0.0 - 24.71.255.255 Shawcable.net Shaw Communications
24.107.0.0/16	24.107.0.0 - 24.107.255.255 USA Charter Communications
24.108.0.0/15	24.108.0.0 - 24.109.255.255 Canada Shaw Communications Inc. spam
24.119.0.0/16	24.119.0.0 - 24.119.255.255 Cableone.net
24.130.224.0/19	24.130.224.0 - 24.130.255.255 USA Comcast Cable NJ
24.147.0.0/16	24.147.0.0/16 USA Comcast Cable NJ spam
24.158.160.0/20 160 St Louis MO	24.158.160.0 - 24.158.175.255 USA Charter Communications CKVL-TN-24-158-

Quadro 5.1 Listagem Servidores C&C ativas

A Lista é armazenada em banco de dados e renovada automaticamente de acordo com funcionalidades implementadas pelo Agente SUA, responsável pela atualização das informações do NIDIA[67, 68], devido à rotatividade de IPs avaliados como suspeito. Esta função possibilita para que o banco não fique muito grande para a realização da filtragem, além de evitar a manutenção na lista de IPs, que saíram da situação de atacante.

Ao ser realizado a captura de pacotes os IPs de origem e destino são analisados e comparados com a “lista negra” do IDS-H, quando detectados, estes pacotes serão eliminados para a redução do volume de dados que passarão pela análise híbrida, sendo notificados para serem realizadas as medidas de defesa.

5.3.2 Coleta de Assinatura

As assinaturas da ferramenta proposta foram coletadas durante 4 meses em diversas fontes, principalmente nos relatórios da Fundação *Shadowserver* [44], enviados semanalmente a usuários cadastrados, sobre ataques recentes notificados sobre *Botnet*. Além dessa fonte, foram verificadas outras fontes de coletas como as ferramentas de S*andboxes* online e gratuitas: Comodo [69], GFI Labs [70] e *ThreatExpert* [71], e cadastrada no sistema.

Uma das principais características em ataques *Bot* que utilizam canais IRC é a execução de suas ações através da comunicação na porta 6667 e protocolo TCP, como principal e 6668, 6669, 7000 e 8000 como portas alternativas. No entanto podemos encontrar o uso de IPs específicos de algumas *Botnet*, além do DNS usado para comunicação entre o *Bot* e o Servidor C&C.

Nas ferramentas de *Sandboxes* foram coletadas duas grandes informações referentes as assinaturas: o MD5(*Message-Digest algorithm*) e o SHA-1(*Secure Hash Algorithm*).

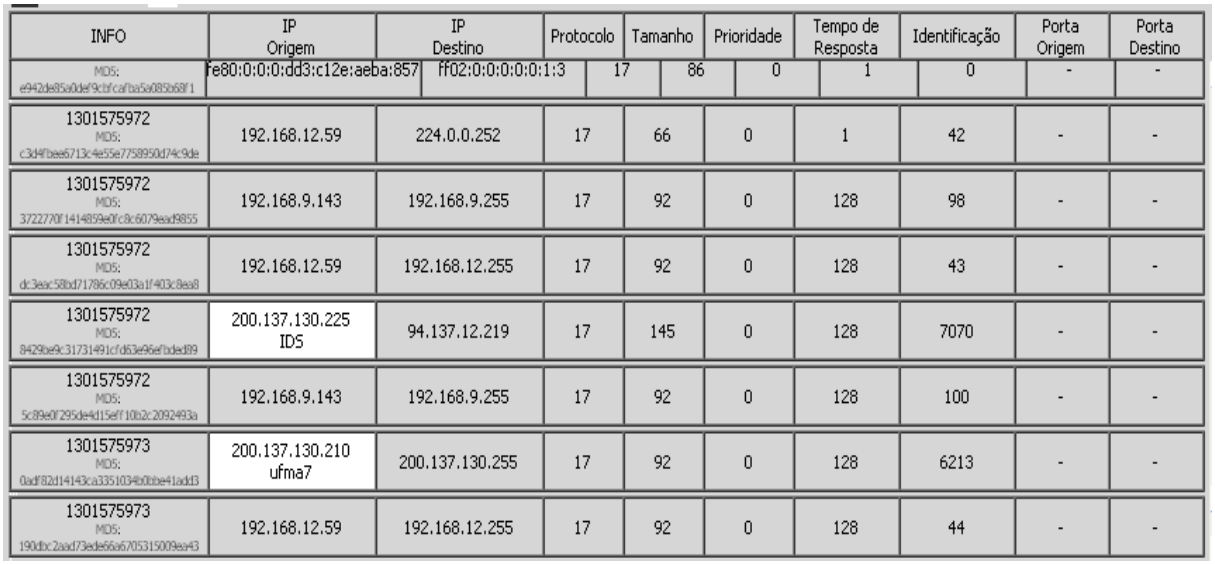
O MD5 é um algoritmo de *hash* de 128 bits unidirecional, representados por uma sequência de 32 caracteres hexadecimais, muito utilizado por softwares que usam protocolo P2P para a verificação da integridade de arquivos, vulnerabilidades e *logins* [71]. O método para a verificação é realizado pela técnica de comparação das duas *hash*, sendo uma da mensagem original confiável e outra da mensagem recebida [72,73].

O SHA-1 é um algoritmo criptográfico, de 160 bits, baseado nos mesmos princípios dos algoritmos MD5. Assim como o MD5, o SHA-1 é bastante utilizado para calcular *checksums* de arquivos. Na sua lógica de aplicação, cada arquivo possui uma assinatura que é deve ser única, o *checksum* [72]. Caso o arquivo seja interceptado e modificado por terceiros, este *checksum* mudará. As ferramentas *Sandboxes* fazem a avaliação desses arquivos, verificando o *checksum* válido, com o que está em análise, caso haja modificações acusará o comprometimento da segurança, informando SHA-1 e o MD5, do arquivo infectado.

A grande vantagem da utilização do MD5 e do SHA-1 é que detecta *Botnet* de todas as classificações, provenientes de IRC, HTTP e P2P, esta última é a de maior dificuldade de detecção pelas ferramentas tradicionais.

5.3.3 Captura de Pacotes

A captura dos pacotes é realizada pelo agente de monitoramento, conforme Figura 5.4, que ao serem coletados podem ser armazenados no formato *pcap*, que podem ser lidos por outras ferramentas posteriormente. Na figura são observados que no momento da coleta de pacotes são identificados quais pacotes são pertencentes aos cadastrados na ferramenta, marcados pela cor branca. No momento da captura é feito a análise de cada pacote pelos agentes de detecção, os pacotes detectados pelo filtro são eliminados e notificados para serem tomadas as medidas cabíveis. Os demais pacotes são encaminhados para a análise por assinatura e anomalia. Como a ferramenta trabalha em modo promiscuo e escuta todo o tráfego, são coletadas informações tanto dos *Hosts* conhecidos, quanto de *Hosts* desconhecidos pela ferramenta.



INFO	IP Origem	IP Destino	Protocolo	Tamanho	Prioridade	Tempo de Resposta	Identificação	Porta Origem	Porta Destino
MD5: e942de85a0def9cbfcafba5a085b68f1	fe80:0:0:0:dd3:c12e:aeba:857	ff02:0:0:0:0:0:1:3	17	86	0	1	0	-	-
1301575972 MD5: c3d4fbae6713c4e55e7758950d74c9de	192.168.12.59	224.0.0.252	17	66	0	1	42	-	-
1301575972 MD5: 3722770f1414859e0fc8c6079eaaf9855	192.168.9.143	192.168.9.255	17	92	0	128	98	-	-
1301575972 MD5: dc3eac58bd71786c09e03a1f403c8ea8	192.168.12.59	192.168.12.255	17	92	0	128	43	-	-
1301575972 MD5: 8429be9c3173491cfd63e96efbded89	200.137.130.225 IDS	94.137.12.219	17	145	0	128	7070	-	-
1301575972 MD5: 5c89e0f295de4d15eff10b2c2092493a	192.168.9.143	192.168.9.255	17	92	0	128	100	-	-
1301575973 MD5: 0adf82d14143ca3351034b0bbe41add3	200.137.130.210 ufma7	200.137.130.255	17	92	0	128	6213	-	-
1301575973 MD5: 190dbc2aad73ede66a6705315009ea43	192.168.12.59	192.168.12.255	17	92	0	128	44	-	-

Figura 5.4 Captura de Pacotes

Essas informações são de responsabilidades da Classe Agente Administrativo, que geram as informações dos pacotes necessárias para a realização das análises nas demais fases do modelo proposto.

5.3.4 Análise por anomalia

A análise de anomalia é dividida em três etapas o Cadastro de Hosts, a Fase de Treinamento e Verificação da Anomalia.

5.3.4.1 Cadastro de *Hosts*

No modelo proposto, os *Hosts* da rede são cadastrados inicialmente com nome e endereçamento IP, para que possam ser reconhecidos na fase de treinamento de PSO e RNA, estes *Hosts* são avaliados durante o treinamento para identificação dos padrões sob execução de tráfego normal e ataque.

5.3.4.2 Fase de Treinamento

Na geração dos registros normais dos *Hosts* são coletados dados através da captura do tráfego da rede cabeada ou rede *wireless*, conforme verificado na Figura 5.5. O perfil dos *Hosts* é determinado através de dados históricos coletados durante um período de operação normal dos mesmos, denominado de **Fase de Treinamento**.

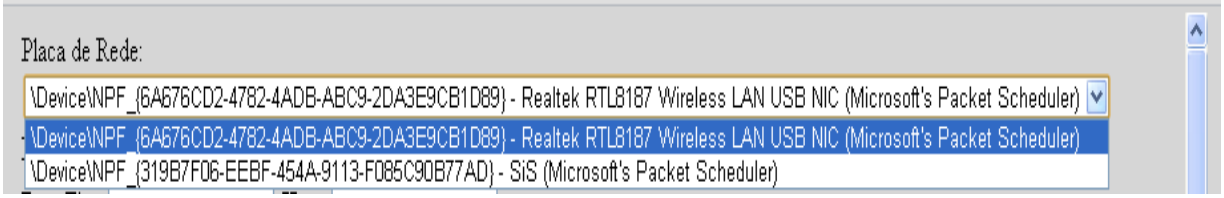


Figura 5.5 Tipo de Placa de Rede para captura de tráfego

A Fase de Treinamento é executada através do trafego da rede em funcionamento normal para adaptação definição de perfil dos *Hosts*, estes perfis são determinado por um conjunto de variáveis que posteriormente são comparadas aos tráfegos, a fim de identificar se há ocorrência de intrusão.

Nesta fase foram utilizadas as técnicas de RNA e PSO, para determinação de melhor técnica de inteligência artificial. Para a análise das técnicas foram utilizadas as variáveis utilizadas nas técnicas são: velocidade de recebimento de pacotes, velocidade de envio de pacotes, número de pacotes enviados, número de pacotes recebidos, número de requisições de DNS e número de bits por pacotes.

5.3.4.2.1 Treinamento utilizando a técnica de PSO

Na fase de treinamento utilizando PSO foi observado o tráfego da rede em dois momentos distintos, o uso da rede em tráfego normal e sob ataque. Ao longo do tempo (*t*) a variável *x* receberá os valores, que serão modificados na mudança da posição no espaço de busca de *2D*, este espaço é determinado por uma matriz de dados, apresentado na figura 5.6, onde é observado uma amostragem do número de pacotes enviados.

Para a realização do problema foi determinado a matriz é dada de ordem *n* por *m*, onde *n* são as linhas e *m* as colunas. Um elemento de uma matriz A que está na i-ésima linha e na j-ésima coluna é chamado de elemento *i,j* ou *(i,j)-ésimo* elemento de A.

$$A = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,m} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,m} \\ \cdots & \cdots & \cdots & \cdots \\ v_{n,1} & v_{n,2} & \cdots & v_{n,m} \end{bmatrix}$$

A fórmula de determinação do $pBest_i(t)$ é:

- $S_0(v_i) = \max\{D[v_{i,j}]\}$, sendo para $1 \leq i \geq n$ e $1 \leq j \geq m$, onde *i* determina a linha de mudança do tempo da variável *x*.

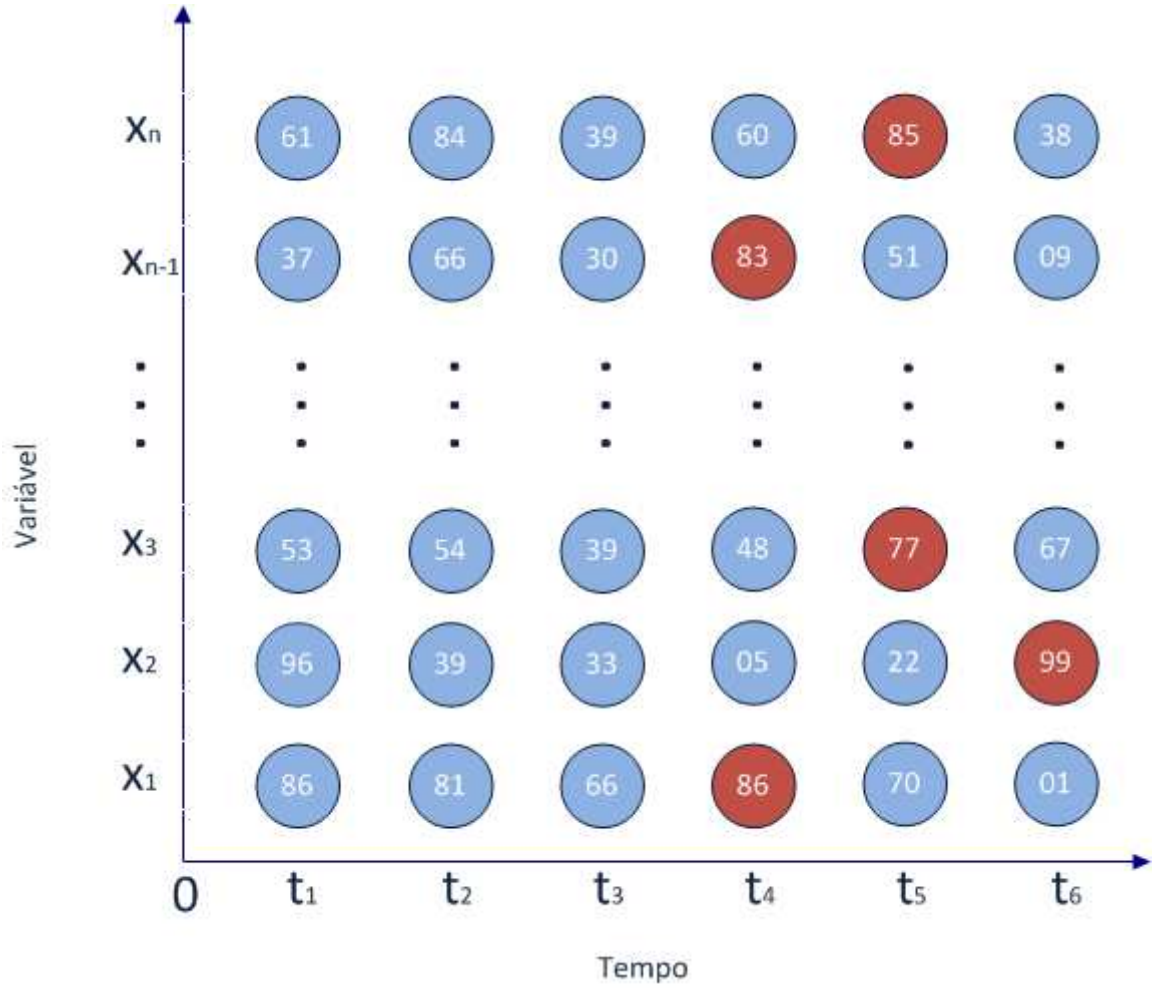


Figura 5.6 Amostragem da determinação do valor de uma variável

Para cada variável *x* é determinado o $pBest_i(t)$, conforme demonstrado nas funções abaixo:

$$S_0(v_1) = \max\{D[v_{1,1}], D[v_{1,2}], D[v_{1,3}] \dots, D[v_{1,m}]\}$$

$$S_0(v_2) = \max\{D[v_{2,1}], D[v_{2,2}], D[v_{2,3}] \dots, D[v_{2,m}]\}$$

...

$$S_0(v_n) = \max\{D[v_{n,1}], D[v_{n,2}], D[v_n, v_3] \dots, D[v_n, v_m]\}$$

Após a identificação dos valores de $pBest_i(t)$, temos então um vetor de valores V_i={ $pBest_1(t)$, $pBest_2(t)$,..., $pBest_n(t)$}, para este vetor é aplicado a fórmula a abaixo, na obtenção de um valor *gBest(t)*:

$$S_0(v_0^*) = \max[S_0(v_1), S_0(v_2), \dots, S_0(v_n)]$$

O algoritmo para esta função é demonstrado a seguir:

```
//obtendo os valores pBest_i(t)
int[] maximos;
maximos = new int[10];
int max = 0;
for(i=0;i<valores.length;i++){
        max = 0;
        for(j=0;j<valores.length;j++){
                if(max<valores[i][j]){
                        max = valores[i][j];
                        maximos[i] = max;
                }
        }
}
//obtendo o valor gBest(t)
max = 0;
for(i=0;i<maximos.length;i++){
        if(max<maximos[i])
                max = maximos[i];
}
```

Quadro 5.2 Algoritmo de PSO

Verificamos que sua implementação é simples, diante outras técnicas de Inteligência Artificial. Além de não necessitar grande quantidade de informações para obtenção de resultados satisfatórios.

Ao final da aplicação o valor *gBest(t)* é assumido por todos os *Hosts*, como valor padrão para especificação para verificação posteriormente pelo Agente de Anomalia.

5.3.4.2.2 Treinamento utilizando a técnica de RNA

Na abordagem utilizando RNA a rede foi observada através de dois momentos para definição de um perfil padrão: condições de tráfego normal e tráfego sob ataques de *Bots*. Para desenvolvimento desta técnica foram seguidos os procedimentos descritos na figura 5.7.

Figura 5.7 Etapas de Aplicação da RNA

O primeiro passo dessa aplicação é a coleta de dados e separação das variáveis a serem utilizadas, realizadas pelo Agente de Monitoramento, gerando um arquivo de treinamento. A próxima etapa é configuração da RNA, através da seleção do paradigma neural mais apropriado a aplicação, neste caso, o paradigma mais comum de aprendizado no caso do reconhecimento de padrões é o supervisionado, associado a uma rede direta multicamadas (MLP) e da determinação da topologia da rede a ser utilizada, determinado pelo algoritmo do quadro 5.3.

```
public static void main(String[] args) throws IOException {
        InputStream is = new FileInputStream("train.data");
        InputStreamReader isr = new InputStreamReader(is);
        BufferedReader br = new BufferedReader(isr);
        TrainingSet trainingSet = new TrainingSet();
        int nrows = 80;
        int nin = 6;
        int nout = 1;
        double train[][] = new double[nrows][nin];
        double target[][] = new double[nrows][nout];

        int k = 0;
        String lin = br.readLine();
        while ( lin != null ) {
           String data[] = lin.split(" ");
           for (int i=0; i < nin; i++)
              train[k][i] = Double.parseDouble(data[i]);
           target[k][0] = Double.parseDouble(data[nin]);
           k++;
```

```
            lin = br.readLine();
        }
        for (int i=0; i<nrows; i++)
        {
            trainingSet.addElement(new SupervisedTrainingElement(train[i],target[i]));
        }
        System.out.println("Criando a rede...");
        MultiLayerPerceptron myMlPerceptron = new
MultiLayerPerceptron(TransferFunctionType.TANH, nin, nin+1, nout);
        System.out.println("Treinando a rede...");
        myMlPerceptron.learnInSameThread(trainingSet);
        System.out.println("Salvando a rede...");
        myMlPerceptron.save("myMlPerceptron.nnet");
    }
```

Quadro 5.3 Algoritmo em Java da RNA

A rede neural multicamada de *perceptrons*, representada na Figura 5.8, foi criada possuindo seis neurônios na camada de entrada, seis neurônios na camada de escondida, e um neurônio de saída.

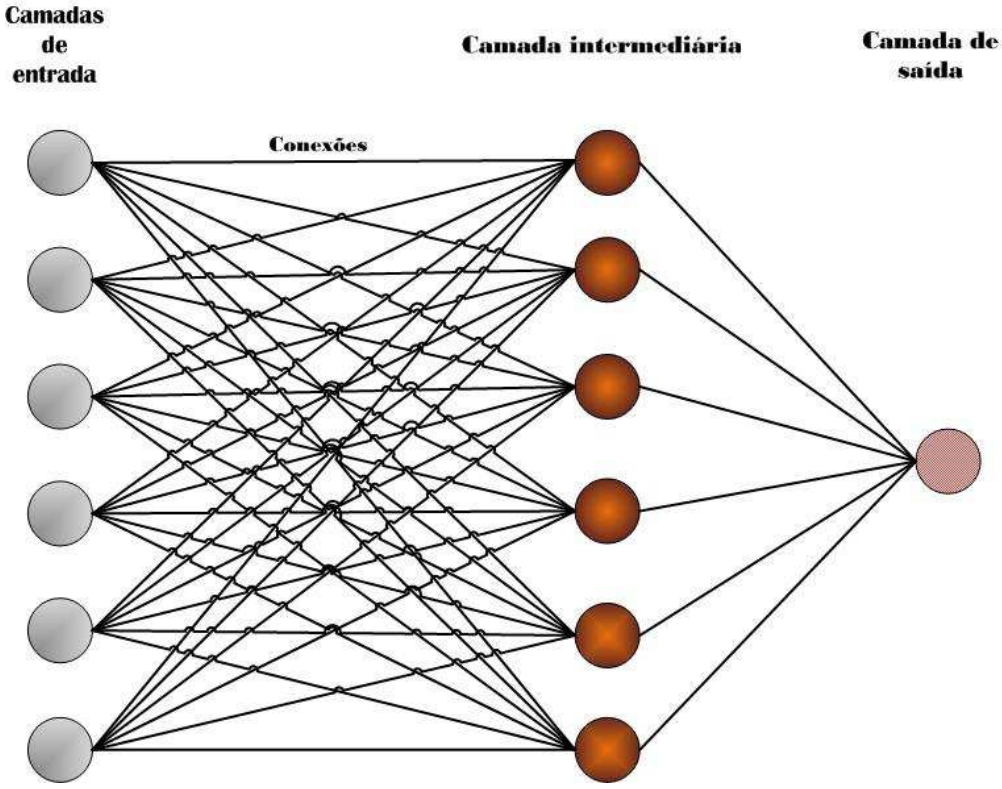


Figura 5.8 Arquitetura da Rede Neural Utilizada

Após a sua criação a rede neural é treinada utilizando o algoritmo de aprendizagem de rede, o Backpropagation, implementado através da biblioteca Neuroph, que tem por finalidade implementação de redes neurais na linguagem *Java*, para obtenção dos valores das variáveis citadas acima. A etapa de testes é comentada na seção 5.4.

5.3.4.3 Verificação da anomalia

Esta etapa é realizada após a eliminação de pacotes pelo filtro e pela análise de assinatura, facilitando assim o processo de análise, visto que, o volume de dados se torna bem inferior.

A verificação por anomalia ocorre através da determinação de valores atribuídos durante as fases de treinamento. Estes valores são utilizados para serem comparados quando a ferramenta realiza a análise, para tanto são definidos valores do perfil padrão dos *Hosts* sob condições normais de tráfego e valores sob condições de ataques.

Para tráfego normal do *Host* foi atribuídos valor -1 e em condições de ataque valor 1, este valor é comparado ao valor da saída desejada para esse mesmo registro de teste. Então são determinados se há ou não anomalia nos pacotes dos tráfegos pela rede, demonstrado posteriormente na seção de resultados.

O procedimento foi aplicado a ambas as técnicas, visto que as duas realizam os mesmo treinamentos sob o mesmo conjunto de dados, apenas aplicando algoritmos diferentes.

5.4 Testes e Resultados

De acordo com o modelo proposto foi desenvolvido um protótipo do IDS-H e realizados testes de validação da ferramenta, como apresentado na Figura 5.9. Para realização dos testes de detecção foi utilizado como ambiente de captura uma rede criada do LABSAC (Laboratório de Sistemas e Arquiteturas Computacionais) do DEE (Departamento de Engenharia de Eletricidade) da UFMA (Universidade Federal do Maranhão).

O ambiente foi composto de 04 computadores e um *notebook*, com interface cabeada, conectados à Internet através de um *switch* conectado à rede da universidade. Em três das máquinas citadas, foram criadas 03 máquinas virtuais, usando o *software VirtualBox* [74], para cada computador, ampliando assim o número para 09 *Hosts* na rede de teste, sendo

a quarta máquina disponível utilizada para controle do *Botmaster* e o *notebook* usado como IDS-H.

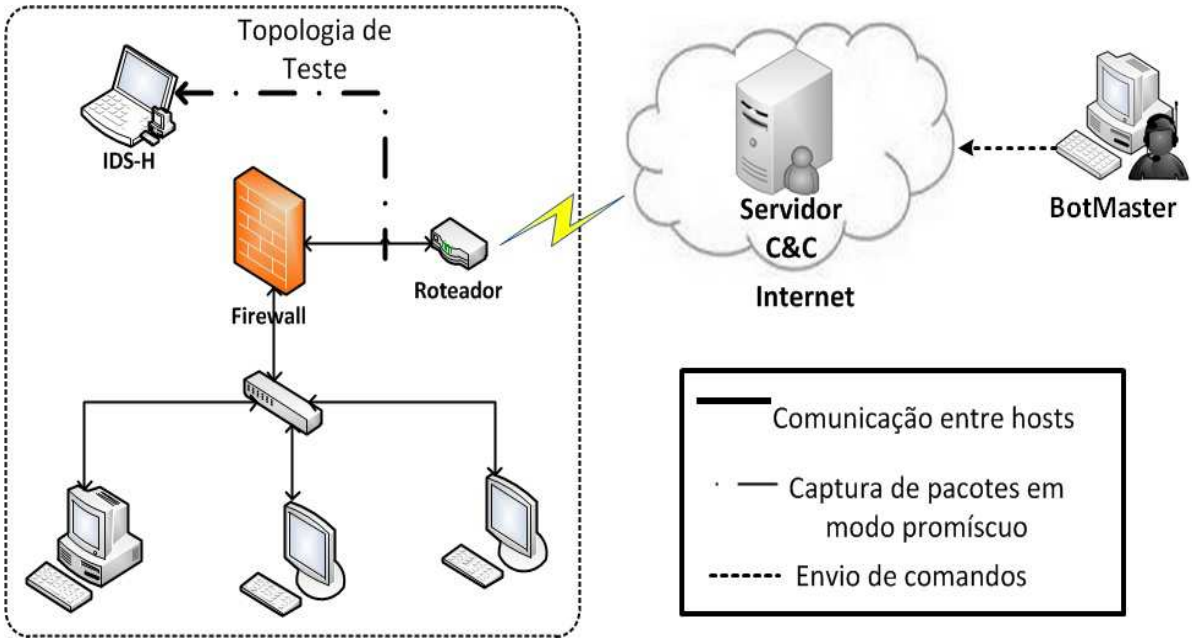


Figura 5.9 Cenário de Teste do IDS-H

Com relação às características das máquinas utilizadas temos:

- **Máquinas de Teste:** Sistema operacional *Windows* XP SP3, utilizando *software VirtualBox* para emular 3 máquinas virtuais. Nas máquinas virtuais foram usados Sistema operacional *Windows* XP SP3. Como recurso físico as máquinas contam com 80 *Gigabyte* de disco rígido, memória 1Gb e processador *Pentium* IV 2.8GHz.
- **Máquinas do *BotMaster*:** Sistema operacional *Windows* 7, utilizando *software* mIRC para integração com o Servidor C&C. Como recurso físico a máquina conta com 250 *Gigabyte* de disco rígido, memória 2G e processador *Phenom* 8600 B.
- ***Notebook* da IDS-H:** Sistema operacional *Windows* XP, utilizando *software* IDS-H. Como recurso físico a máquina conta com 320 *Gigabyte* de disco rígido, memória 3G e processador *Dual Core*T4400 2.2GHz. O *hardware* se mostrou mais do que suficiente, já que esta máquina é apenas responsável pela coleta dos dados.

Para a realização dos testes de detecção da ferramenta foi necessário um treinamento dos *Hosts* para a determinação um perfil padrão. Esta fase teve duração de 03 horas, realizada no dia 16 de fevereiro, onde foram coletados 1.084.008 pacotes, ou seja aproximadamente 350Mb de dados em tráfego normal como: verificação de *e-mail*,

navegação em páginas, *downloads*, entre outras atividades básicas. Os testes foram aplicados tanto as técnicas de PSO e como a de RNA, esses resultados estão especificados posteriormente nos resultados. O mesmo procedimento foi aplicado a um conjunto de dados em ataque, sendo coletados 1.364.130 pacotes ou 402Mb de dados desse tráfego, no dia 17 de fevereiro.

5.4.1 Resultado da Fase de Treinamento

Na fase de treinamento os *Hosts* foram submetidos ao uso normal da rede para determinarmos um perfil padrão, a ser utilizado na posteriormente na aplicação de detecção por anomalia, estes dados são comparados ao tráfego em testes para ser verificado a condição de anomalia, neste período foram coletados os seguintes valores:

Variável	**Resultado do Treinamento (Unidade)**	
	PSO	**RNA**
Velocidade de recebimento de pacotes	272	270
Velocidade de envio de pacotes	157	156
Número de pacotes enviados	94	95
Número de pacotes recebidos	43	44
Número de requisições de DNS	03	03
Número de bits por pacotes	12112	12112

Tabela 5.1 Formação dos Registros do Treinamento em PSO e RNA

A tabela 5.1 demonstra que os testes realizados na fase de treinamento utilizando PSO e RNA, das variáveis designadas para determinação de intrusos, são bem próximos, no entanto estes valores estão mais precisos nas redes neurais, observado na detecção por anomalia. No entanto a técnica de PSO para a definição do seu perfil foram utilizados 23 minutos de processamento, enquanto que, na técnica de RNA esses tempo foi superior a 1 hora e 15 minutos. Quando utilizamos a metade dos dados coletados na aplicação do PSO, o resultado não houve mudanças significantes, no entanto em RNA, não foi possível determinado o padrão, demonstrando que a técnica de PSO pode ser aplicada a um conjunto reduzido de dados, além de estabelecer resultados em menor intervalo de tempo.

5.4.2 Teste com *Botnet* Baseada em Protocolo IRC

Na construção do primeiro cenário de teste foi necessário a implementação de um ambiente para atuação da *Botnet*. Neste estudo foi utilizado a *Botnet RxBot* versão 7.6 [7], o *software* de comunicação *mIRC* [76] e canal de comunicação IRC, como servidor C&C, localizado na Internet. Esse teste foi realizado no período entre 17 e 18 de fevereiro de 2011, período no qual a solução foi utilizada na rede do cenário descrito, foram coletados 2.921.816 pacotes em aproximadamente 1,2 Gb.

Para a infecção das máquinas foi utilizado a principal forma de infecção das *Botnets*, o envio de *spam*. Neste processo foram infectadas 04 máquinas virtuais. Ao receberem o arquivo *rbot.exe*, por e-mail, os usuários ao executarem o arquivo, foram notificados pelo Anti-vírus e/ou pelo *Firewall*, sobre o tipo de risco do arquivo, mesmo diante do bloqueio dessas ferramentas, o arquivo infecta as máquinas sem o conhecimento do usuário.

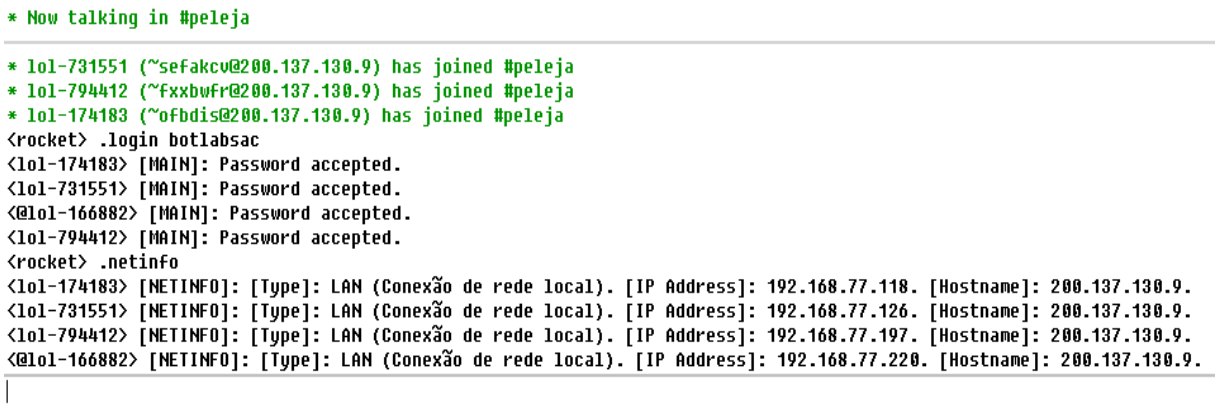


Figura 5.10 Login com o Servidor C&C

A partir da infecção, foi realizado a comunicação do *Botmaster* com o Servidor C&C, através do uso do software de comunicação *mIRC*, que lista todos os *Bots* disponíveis, e realizado o login com a *Botnet*, descrito na Figura 5.10. Nesta figura é observado que após o login, os *Bots* aceitam a senha do *BotMaste*r para comunicação entre eles a espera de comandos.

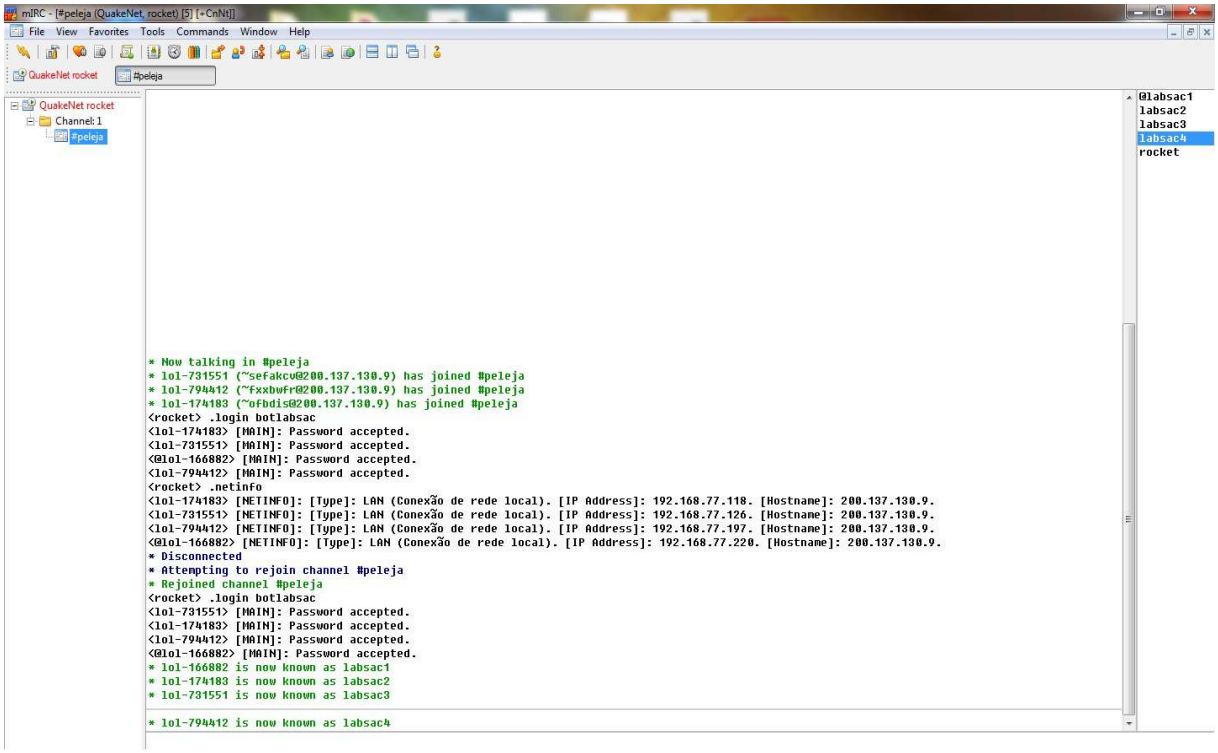


Figura 5.11 Controle do Botmaster aos *Bots* disponíveis

Após a comunicação com o Servidor C&C, o *BotMaster* visualiza todos os *Bots* disponível no momento, conforme Figura 5.11. O controlador da *Botnet* pode enviar a partir desse momento comunicação com todos os *Bots* ao mesmo tempo, ou individualmente. Para fins de pesquisa, foram alterados os nomes dos *Bots* ativos para @labsac1, labsac2, labsac3 e labsac4, descrito na Figura 5.12.

As máquinas *Bots*, sempre que eram desligadas, ficavam indisponíveis para o *BotMaster*, no Servidor C&C, mas sempre que eram ligadas faziam a conexão com o Servidor C&C, da *Botnet*, automaticamente sem a intervenção humana, para sua manipulação, era necessário apenas que o *BotMaster* realizasse o *login* com o Servidor.

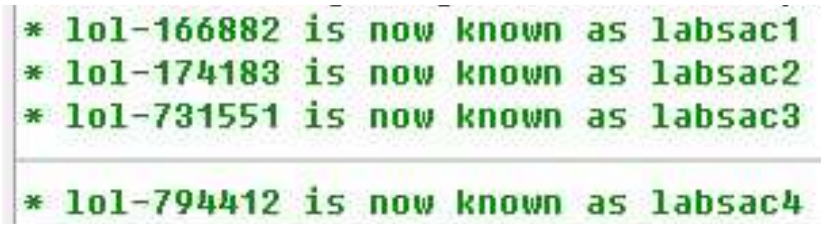


Figura 5.12 Nomes dos *Bots* ativos

Nos testes foram realizados diversas operações de interação entre o *Botmaster* e os *Bots*:

- Mudança de Nome do *Bot*

- .*nick* <novo nome>

- Captura de Tela
  - .*capture screen* <diretório da imagem salva+formato>
- Desligamento e Reinicio das Maquinas
  - .*shutdown*
  - .*reboot*
- Obter Todas as Chaves de Programas Instalados
  - .*getcdkeys*
- *Keylog*
  - .*keylog* <on/off>
- Funções DDoS
  - .ddos.(syn/ack/randon) <ip><porta><duração>
  - .*pingflood* <ip><porta><tamanho dos pacotes> <*delay*>
  - .*pingstop*
  - .*udpflood* <ip><pacotes><tamanho dos pacotes><*delay*>
  - .*icmpflood*<ip><tamanho dos pacotes>
- Visitas a url
  - .*visit* <url>
- *Download* de arquivos
  - .*download* <url> <diretório aonde será salvo> <ação (0/1)> se 1 o arquivo será executado, caso contrário só será salvo.

Dentre os ataques realizados, destacamos o ataque de Negação de Serviço Distribuído, de acordo com a Figura 5.13. O comando "*.icmpflood*" foi utilizado para a realização do ataque DDoS, direcionado à máquina de endereço IP 192.168.77.213 durante o período de 120 segundos, ou seja, 3 minutos de ataque enviados por todos os *Bots* ativos no canal de comunicação conectados no momento.

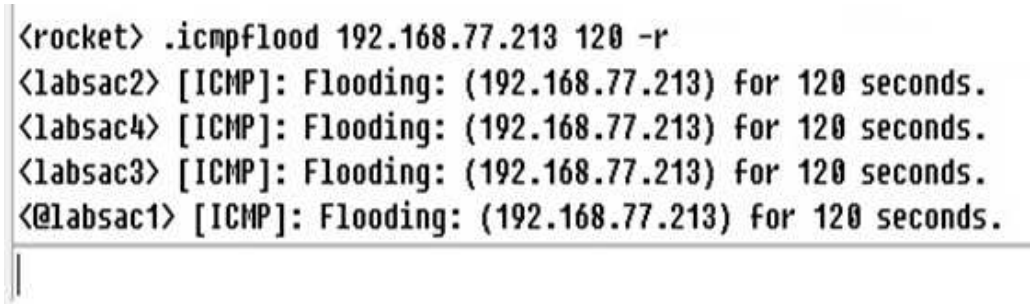


Figura 5.13 Execução de um ataque *icmpflooding*

Outro ataque de Negação de Serviço realizado durante o teste foi através do "*.ddos.random*", executado ao alvo de IP número 192.168.77.213, pela porta 5000, durante um período de 300 segundos, conforme Figura 5.14. Na figura é demonstrado a execução de cada *Bot*, ao final dessa ação, os *Bots* encaminham a resposta do ataque ao *BotMaster*, contendo as informações do IP do alvo, número de pacotes enviados, médias de pacotes por segundo e tamanho total dos pacotes durante o processo do ataque.

Nesta fase o IDS-H realizava a captura do tráfego da rede em modo promíscuo, sem interferir no tráfego da rede e a partir deste momento eram executados a verificação das análises apresentadas pela ferramenta.

Na sequência são discutidos os resultados coletados pela arquitetura implementado no cenário descrito acima (Figura 5.9).

```
<rocket> .ddos.random 192.168.77.213 5000 300
<labsac4> [DDoS]: Done with flood (0KB/sec).
<labsac2> [DDoS]: Done with flood (0KB/sec).
<labsac3> [DDoS]: Done with flood (0KB/sec).
<labsac2> [DDoS]: Flooding: (192.168.77.213:5000) for 300 seconds.
<labsac4> [DDoS]: Flooding: (192.168.77.213:5000) for 300 seconds.
<labsac3> [DDoS]: Flooding: (192.168.77.213:5000) for 300 seconds.
<@labsac1> [DDoS]: Flooding: (192.168.77.213:5000) for 300 seconds.
<@labsac1> [DDoS]: Done with flood (0KB/sec).
<labsac2> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 1211653 packet(s) @ 236KB/sec (69MB).
<@labsac1> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 53772 packet(s) @ 10KB/sec (3MB).
<labsac4> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 279115 packet(s) @ 54KB/sec (15MB).
<labsac3> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 266232 packet(s) @ 51KB/sec (15MB).
```

Figura 5.14 Execução e resposta de um ataque DDoS

5.4.3 Resultado da *Botnet* Baseada em Protocolo IRC

Nos testes realizados com a *Botnet* baseada em protocolo IRC, verificou-se que mesmo os *Bots* estando utilizando um único servidor C&C, a *Botnet* possuía IP diferente de comunicação, demonstrado no tráfego coletado pelo IDS-H no quadro 5.4.

SRC: 10.0.2.15:1075	**DST: 158.38.8.251:6669**	Size = 48 bytes
SRC: 158.38.8.251:6669	DST: 10.0.2.15:1075	Size = 44 bytes
SRC: 10.0.2.15:1075	**DST: 158.38.8.251:6669**	Size = 40 bytes
SRC: 10.0.2.15:1075	**DST: 158.38.8.251:6669**	Size = 88 bytes
SRC: 192.168.88.130:1033	**DST: 194.109.129.222:6669**	Size = 48 bytes

SRC: 194.109.129.222:6669	DST: 192.168.88.130:1033	Size = 44 bytes
SRC: 192.168.88.130:1033	**DST: 194.109.129.222:6669**	Size = 40 bytes
SRC: 192.168.88.130:1033	**DST: 194.109.129.222:6669**	Size = 87 bytes

Quadro 5.4 Tráfego Coletado de um único Servidor C&C

No tráfego acima, foi identificado pelo agente de análise de assinatura, devido a porta 6669, usada para canal IRC para comunicação em que o *Botmaster* exerceu com os *Bots*, neste caso de utilizarmos apenas o número da porta como assinatura, o falso positivo só não ocorre, porque foi testado em uma rede simulada, mas em um grande rede é necessário a verificação da necessidade do uso das portas do canal IRC.

No.	Tempo	IP Origem	IP Destino	Protocolo	Comprimento do Pacote
1253	36.313547	192.168.77.220	192.168.77.213	ICMP	1066
1254	36.317338	192.168.77.220	192.168.77.213	ICMP	1066
1257	36.322662	192.168.77.138	192.168.77.213	ICMP	1066
1262	36.355936	192.168.77.136	192.168.77.213	ICMP	1066
1265	36.357591	192.168.77.102	192.168.77.213	ICMP	1066
1266	36.358779	192.168.77.102	192.168.77.213	ICMP	1066
1267	36.358815	192.168.77.136	192.168.77.213	ICMP	1066
1268	36.364762	192.168.77.220	192.168.77.213	ICMP	1066
1269	36.364804	192.168.77.138	192.168.77.213	ICMP	1066
1270	36.371499	192.168.77.138	192.168.77.213	ICMP	1066
1271	36.375416	192.168.77.138	192.168.77.213	ICMP	1066
1272	36.381277	192.168.77.138	192.168.77.213	ICMP	1066
1273	36.385831	192.168.77.138	192.168.77.213	ICMP	1066
1274	36.388104	192.168.77.138	192.168.77.213	ICMP	1066
1276	36.408643	192.168.77.136	192.168.77.213	ICMP	1066

Figura 5.15 Verificação de Pacotes do Ataque de ICMP *Flooding*

O uso de diversos comandos de controle de *Bots* foram utilizados, dentre eles destaca-se o "*icmpflood*", que consiste num *flooding*, ataque de negação de serviço simples que sobrecarrega o sistema da vítima [77]. O ataque foi observado conforme dados coletados e filtrados para o protocolo ICMP, descrito na Figura 5.15. Neste ataque os 04 *Bots* ativos são identificados na figura pelos IPs de Origem de números 192.168.77.102, 192.168.77.136, 192.168.77.138 e 192.168.77.220. Estes *Bots* enviaram pacotes de comprimento de 1066 *bytes*.

Os *Bots* ao finalizarem o ataque encaminham o resultado ao seu *BotMaster*, descrito na Figura 5.16.

```
<labsac2> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 1211653 packet(s) @ 236KB/sec (69MB).
<@labsac1> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 53772 packet(s) @ 10KB/sec (3MB).
<labsac4> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 279115 packet(s) @ 54KB/sec (15MB).
<labsac3> [ICMP]: Done with  flood to IP: 192.168.77.213. Sent: 266232 packet(s) @ 51KB/sec (15MB).
```

Figura 5.16 Resposta do Ataque de ICMP *Flooding*

Observa-se no gráfico 5.1, que o número de pacotes recebidos durante um ataque de negação de serviço, recebe uma elevada variação. Inicialmente os *Host* A e *Host* B, operavam normalmente nos instantes t_1, t_2 e t_3, no instante t_4 passaram a sofrer ataques DoS, sofrendo a alteração dos seus valores, passando a adotarem valores de anomalia, este foi detectado pelas duas técnicas utilizadas.

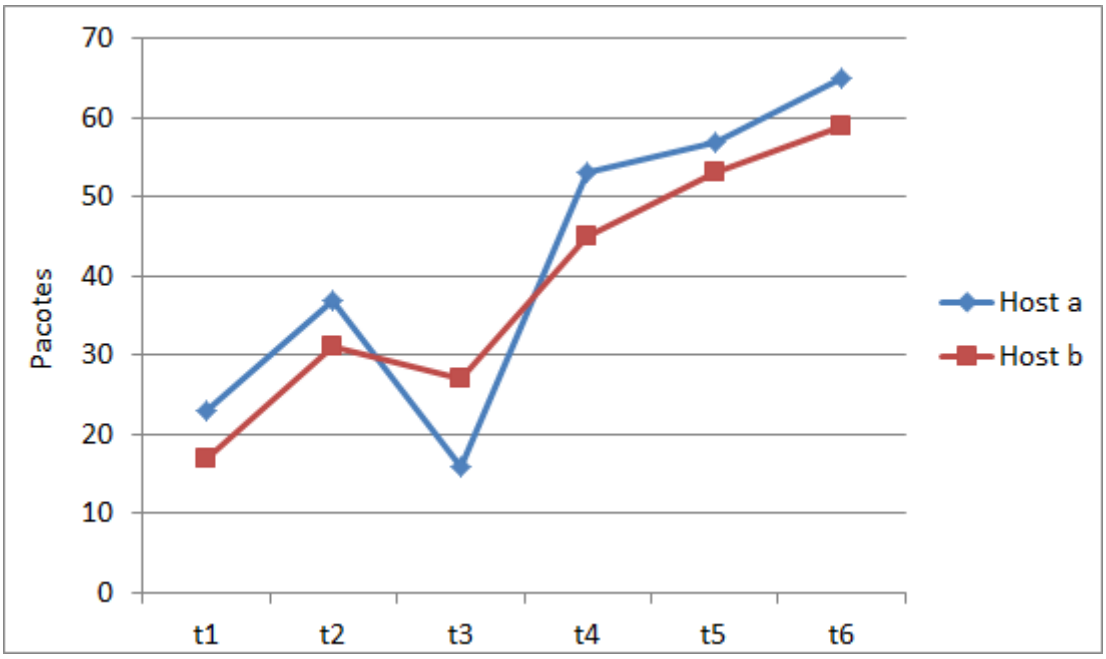


Gráfico 5.1 Número de pacotes recebidos

Na verificação do outro ataque simulado, DDoS. *random*, é demonstrado na Figura 5.17, o envio de pacotes pelos *Bots* usado para a realização do ataque de Negação de Serviço Distribuído. Nesta ação os IPs 192.168.77.102, 192.168.77.220, 192.168.77.136 e 192.168.77.138 realizam o ataque enviando pacotes com 1514 *bytes* por pacotes, correspondente a 12112 *bits* em cada pacote, tamanho máximo permitido para os pacotes, ao IP 192.168.77.213. No entanto, na arquitetura proposta, pode-se observar que a avaliação do número de *bits* por pacote, com grandeza em determinação do perfil, se torna falha nas duas técnicas de inteligência artificial, pois em um ataque como o DDoS usado os pacotes possuem valores iguais. Além dessas características, na *Botnet* utilizada podemos usar os comandos

pingflood e *udpflood*, que permitem ser acrescentado o tamanho do pacote segundo desejo do controlador da *Botnet*:

- *.pingflood* <ip> <quantidade de pacotes> <tamanho dos pacotes> <intervalo>
- *.udpflood* <ip> <quantidade de pacotes> <tamanho dos pacotes> <intervalo>

1002834	2068.06791	192.168.77.102	192.168.77.213	IPv4	1514
1002835	2068.06793	192.168.77.102	192.168.77.213	IPv4	1514
1002837	2068.08316	192.168.77.220	192.168.77.213	IPv4	1514
1002838	2068.08318	192.168.77.220	192.168.77.213	IPv4	1514
1002850	2068.11773	192.168.77.136	192.168.77.213	IPv4	1514
1002851	2068.11775	192.168.77.136	192.168.77.213	IPv4	1514
1002856	2068.15760	192.168.77.138	192.168.77.213	IPv4	1514
1002857	2068.15762	192.168.77.138	192.168.77.213	IPv4	1514
1002859	2068.21649	192.168.77.102	192.168.77.213	IPv4	1514
1002860	2068.21651	192.168.77.102	192.168.77.213	IPv4	1514
1002862	2068.23429	192.168.77.220	192.168.77.213	IPv4	1514
1002863	2068.23433	192.168.77.220	192.168.77.213	IPv4	1514
1002865	2068.26746	192.168.77.136	192.168.77.213	IPv4	1514
1002866	2068.26750	192.168.77.136	192.168.77.213	IPv4	1514
1002868	2068.30738	192.168.77.138	192.168.77.213	IPv4	1514
1002869	2068.30740	192.168.77.138	192.168.77.213	IPv4	1514

Figura 5.17 Verificação de Pacotes do Ataque de Negação de Serviço Distribuído

O comportamento dos *Bots* durante um ataque faz com que se torne fácil sua identificação na análise por anomalia, devido a ação conjunta de dois ou mais *Bots*, mas de difícil detecção quando este *Bot* age sozinho, causando uma incerteza quanto ao seu ataque. Neste caso a técnica de análise baseada em assinatura se faz eficaz, visto que detecta um *Bot* isolado.

Pacotes	**No de Registros**	**Técnica de Detecção**	
		PSO	**RNA**
Normal	1.708.282	0,00%	0,00%
Ataque	1.213.534	99,00%	99,2%
Total	2.921.816	-	-

Tabela 5.2 Detecção por Anomalia ao Ataque de *Botnet* baseada em protocolo IRC

A Tabela 5.2 apresenta o número de registros normais e registros de ataques, utilizado durante os Testes com *Botnet* baseado em protocolo IRC. A detecção por anomalia quando utilizamos a técnica de RNA possui uma taxa de falso positivo menor que a técnica de PSO, de 0,8% na primeira técnica em relação à de 1% para a segunda.

5.4.4 Teste com *Botnet* Baseada em Protocolo HTTP

A segunda fase de teste foi realizada a partir do cenário predefinido, descrito na Figura 5.9. Neste caso de uso, utilizamos a *Botnet Zeus* [78], que trabalha com protocolo HTTP. Na utilização desta *Botnet* fez-se uso do *software Xampp* [79] como servidor HTTP e canal de comunicação HTTP e banco de dados *SQL Server*. Esse teste foi realizado no período entre 09 e 10 de março de 2011, foram coletados 1.945.086 pacotes em aproximadamente 730Mb.

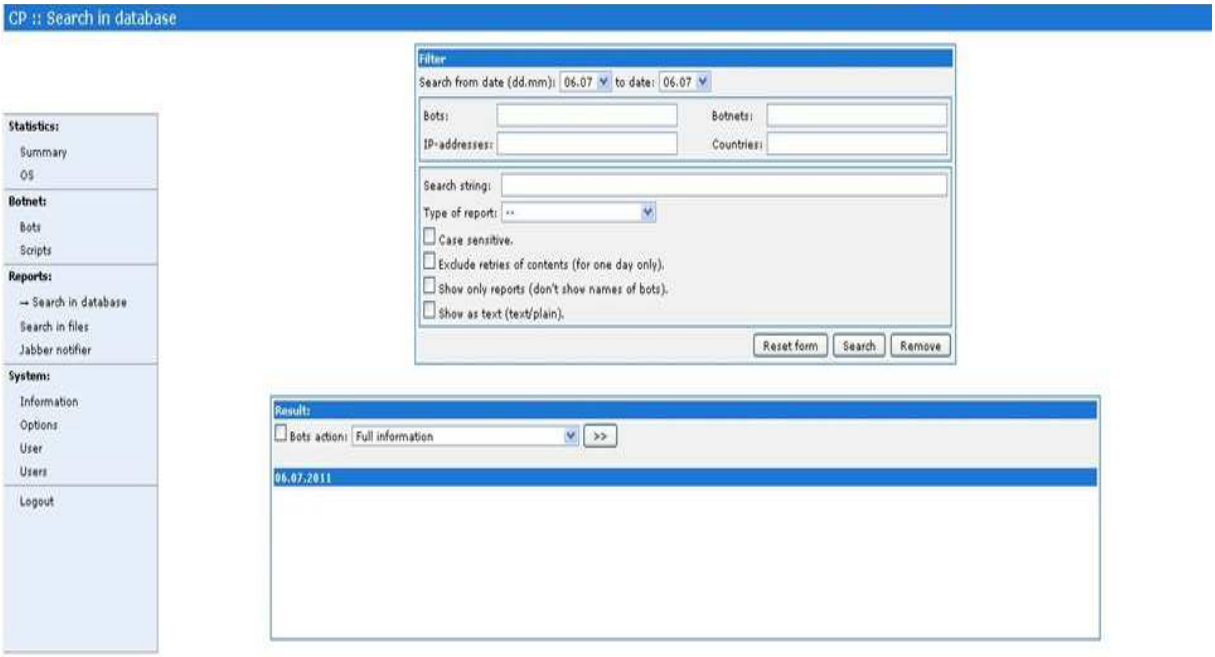


Figura 5.18 Central de Comando e Controle da *Botnet Zeus*

Para a execução do ambiente foi necessário a infecção das máquinas, usando o envio de e-mail do arquivo *ZBot*, executável de infecção dos *Bots*.

A *Botnet Zeus* utiliza uma central de controle baseado em páginas web, para operação das ações a serem encaminhadas para os *Bots*, conforme Figura 5.18. Após a infecção do *Bot*, o mesmo realiza a busca para se conectar ao servidor C&C, a partir de conexão, seus dados serão repassados para a Central de Controle do *BotMaster*.

Nos testes observamos que toda vez que a vítima realiza *login* ou preenche algum campo no site, estes são capturados e enviados para o Servidor C&C, conforme Figura 5.19. Os dados do *Bot* são armazenados no Banco de Dados, para serem analisados posteriormente pelo *BotMaster*, segundo relatórios existentes. Esses dados são criptografados e enviados pelo *Bot* ao Servidor C&C, o que dificulta a detecção por algumas ferramentas de segurança.

Figura 5.19 Dados capturados do *Bot*

A *Botnet* faz a verificação dos processos que são executados pelos *Bots* e monitora toda a execução de suas ações, observados na Figura 5.20, para analisar que tipo de serviços são utilizados pelo usuário e obter as informações necessárias para capturas de senhas e outros dados.

Nesta simulação o IDS-H realizou a captura do tráfego da rede. A seguir discutido os resultados coletados pela arquitetura implementada no cenário descrito na Figura 5.9.

id	bot_id	botnet	bot_version	path_source	path_dest	time_system	time_tick	time_localbias	os_version	language_id	process_name	process_user
1	ufma_b75eeef627_000ed5b9	-- default --	16910099			1309973512	973019	0	[BLOB - 20Bytes]	1046	C:\WINDOWS \system32 \services.exe	GRUPO\UFMA-B75EEEF627$
2	ufma_b75eeef627_000ed5b9	-- default --	16910099			1309973532	993788	0	[BLOB - 20Bytes]	1046	C:\WINDOWS \system32 \wercIsid.exe	UFMA-B75EEEF627\labsac
3	ufma_b75eeef627_000ed5b9	-- default --	16910099	https://login.live.com /ppsecure /post.srf?wa=wsigni...		1309973606	1067424	0	[BLOB - 20Bytes]	1046	C:\Arquivos de programas\Internet Explorer\iexplor...	UFMA-B75EEEF627\labsac
4	ufma_b75eeef627_000ed5b9	-- default --	16910099	https://www.paypal.com /br/cgi-bin/webscr?SESSION=d...		1309974194	1655640	0	[BLOB - 20Bytes]	1046	C:\Arquivos de programas\Internet Explorer\iexplor...	UFMA-B75EEEF627\labsac

Figura 5.20 Verificação dos processos realizados pelo *Bot*

5.4.5 Resultado da *Botnet* Baseada em Protocolo HTTP

Nos testes realizados com a *Botnet* baseada em protocolo HTTP, verificou-se que ao troca de mensagem HTTP entre o *Bot* e o Servidor C&C, demonstrado no tráfego coletado pelo IDS-H na Figura 5.21. Nesse teste a ferramenta realizou a detecção por anomalia desta *Botnet*.

31	27.474286	10.0.2.15	192.168.77.185	HTTP	224 GET /xampp/1/cfg.bin HTTP/1.1
95	29.670321	10.0.2.15	192.168.77.185	HTTP	620 POST /xampp/1/gate.php HTTP/1.1
101	29.711909	10.0.2.15	192.168.77.185	HTTP	531 POST /xampp/1/gate.php HTTP/1.1
2084	91.542849	10.0.2.15	192.168.77.185	HTTP	620 POST /xampp/1/gate.php HTTP/1.1
5317	153.305514	10.0.2.15	192.168.77.185	HTTP	380 POST /xampp/1/gate.php HTTP/1.1
7081	724.135144	10.0.2.15	192.168.77.185	HTTP	1226 POST /xampp/1/gate.php HTTP/1.1
7590	845.075355	10.0.2.15	192.168.77.185	HTTP	253 POST /xampp/1/gate.php HTTP/1.1
7611	1231.741243	10.0.2.15	192.168.77.185	HTTP	531 POST /xampp/1/gate.php HTTP/1.1
7615	1232.616770	10.0.2.15	192.168.77.185	HTTP	429 POST /xampp/1/gate.php HTTP/1.1
9211	1295.608583	10.0.2.15	192.168.77.185	HTTP	921 POST /xampp/1/gate.php HTTP/1.1

Figura 5.21 Tráfego HTTP entre o *Bot* e o Servidor C&C

O elevado tráfego facilita a verificação de pacotes enviados entre o *Host* e a conexão da Internet. Durante esse teste, verificamos que a troca de mensagem pode possuir informações relevantes entre os componentes da *Botnet*, tornando necessário a ampliação da detecção focada nas informações do pacote para redução de falsos positivos, que não foi implementado na ferramenta.

Pacotes	**No de Registros**	**Técnica de Detecção**	
		PSO	**RNA**
Normal	1.290.184	0,00%	0,00%
Ataque	654.902	99,02%	99,3%
Total	1.945.086	-	-

Tabela 5.3 Detecção por Anomalia ao Ataque de *Botnet* baseada em protocolo HTTP

Na Tabela 5.2 é apresentado o número de registros normais e registros de ataques, utilizado durante os Testes com *Botnet* baseado em protocolo HTTP, onde foi verificado a pequena diferença entre as técnica de inteligência, chegando a 0,1%, entre a taxa de falso positivos entre as técnicas. Nesta etapa a taxa de falso positivo do PSO é 0,8% contra 0,7 em RNA.

5.4.6 Teste Simulado na Rede da UFMA

A terceira fase de análise da ferramenta foi aplicação em um novo cenário de captura, descrito na Figura 5.22, onde a IDS-H foi inserida para realizar verificação dos pacotes na sub-rede da UFMA.

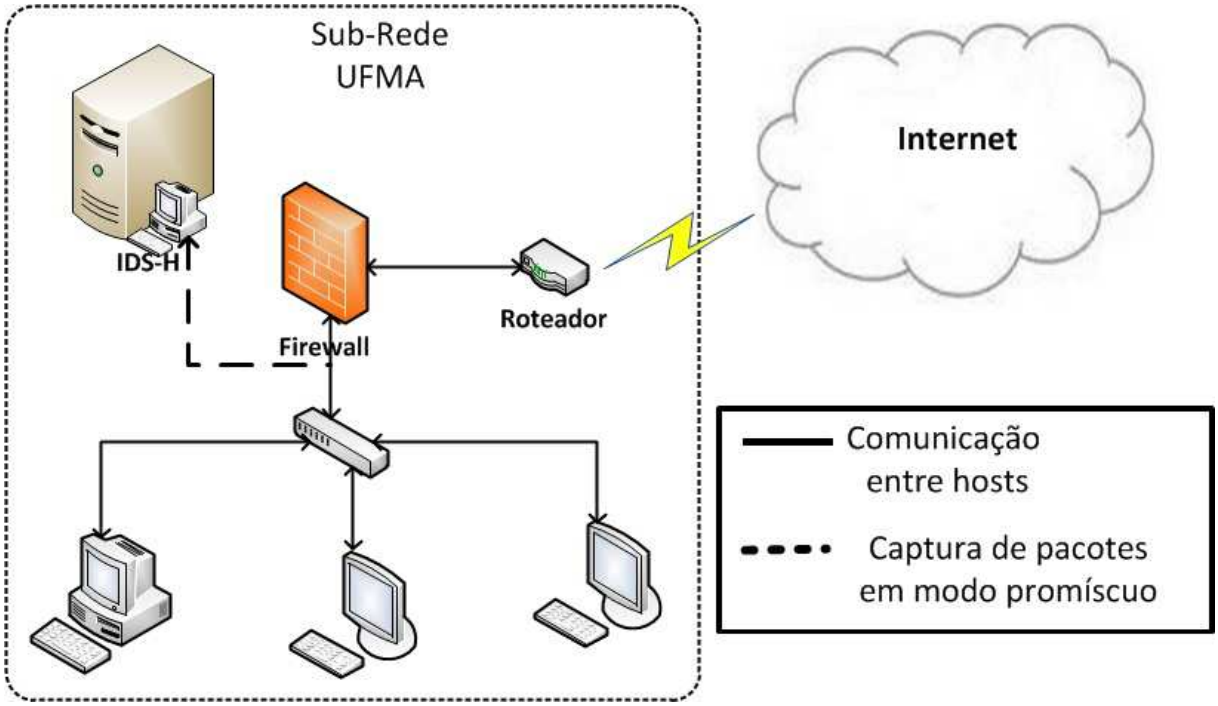


Figura 5.22 Cenário de Teste do IDS-H na sub-rede da UFMA

Durante 05 dias foram coletados pacotes do tráfego na sub-rede. Nesta avaliação foram detectadas algumas falhas na ferramenta proposta, devido a capacidade de processamento da máquina da IDS-H, descrita no item 5.4. A principal fragilidade foi verificada no tamanho do banco de dados da captura dos pacotes, que por alguns momentos fez com que a máquina não respondesse, pois a ferramenta além de fazer a captura faz as análises do pacote, automaticamente. Tendo que, por diversas vezes ser reiniciado essa captura do tráfego. Neste período a coleta dos dados fez gerar um banco de dados superior a 6 *Gigabytes*, 15.192.454 pacotes coletados.

Na sequência são discutidos os resultados coletados pelo modelo proposto implementado no cenário descrito acima, na Figura 5.22.

5.4.7 Resultado dos Testes na Sub-Rede da UFMA

Na verificação do tráfego, após os 05 dias de coletas de dados, foram analisados pela ferramenta, que em sua grande maioria dos ataques detectados foram pelo Agente de Análise de Assinatura, correspondente a 58,37% das *Botnets* encontradas na sub-rede da UFMA, conforme gráfico 5.2. Seguido pelo Agente de Filtragem, com 31,43% de detecção e 10.20% (25 ataques) pelo Agente de Análise Anomalia baseado PSO, num total de 245 invasores identificados pela IDS-H. Este número se torna mais elevado quando utilizamos a técnica de detecção por Redes Neurais, que identificou 32 ataques (12,70%) num total de 252 intrusos identificados. No entanto, o simulador das RNAs, não executa nesta aplicação em tempo real, diferentemente da técnica de PSO, comparados na Tabela.

Forma de Detecção	**PSO**		**Redes Neurais**	
	Ataques Detectados	**%**	**Ataques Detectados**	**%**
Filtragem	77	31,43	77	30,56
Análise por Assinatura	143	58,37	143	56,75
Análise por Anomalia	25	10,20	32	12,70
Total	245	100,00	252	100,00

Tabela 5.4 Tabela Comparativa entre as Formas de Detecção

Está análise demonstra a necessidade da ação conjunta do Sistema de Detecção de Intruso Híbrido para verificações de invasões provenientes de *Botnets*, e que os ataques foram identificados com maior eficácia e rapidez, devido a possibilidade de eliminação de pacotes na filtragem pela *BlackList*.

De acordo com as *Botnets* detectadas na análise por assinatura, observa-se uma prevalência das *Botnets SDBot* e *SpyBot*, no tráfego da rede, que 32,87% das *Botnets* identificadas e que a grande maioria das *Botnets* são baseadas em protocolo de comunicação IRC, conforme tabela 5.5. Estes dados foram coletados num sub-rede da UFMA, sobre o uso de tráfego sem simulação.

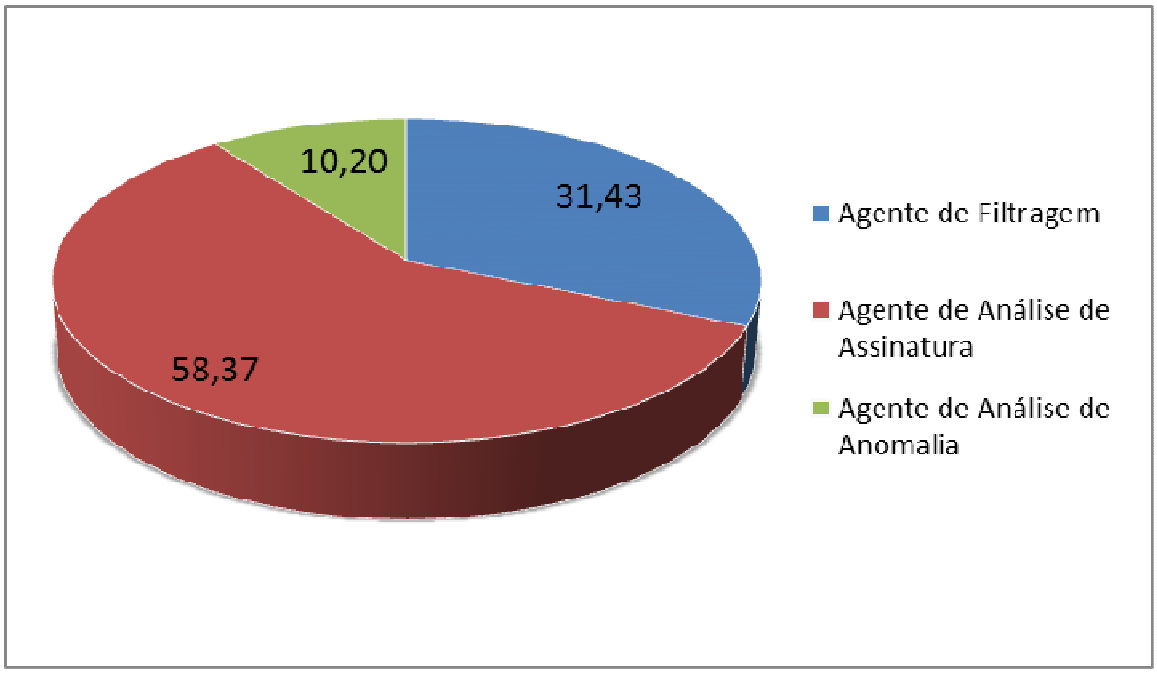


Gráfico 5.2 Percentual de Detecção segundo Agente de Análise

O estudo demonstra que a ferramenta proposta atende as necessidades para detecção de *Bots*, e que nas redes internas nas organizações não é necessário à coleta de grandes volumes de dados, pois a ação de *Botnets* não é duradora dentre de uma rede, em especial a de atividades de ataques, portanto é necessária a identificação em tempo real do intruso.

Assinatura de *Botnets*	Tipo de Protocolo	Quantidade
AGOBot	IRC, HTTP	15
Bototer	IRC, HTTP, P2P	3
CodBot	IRC	7
Dogrobot	IRC, P2P	3
IRCBot	IRC	13
Kbot	IRC	12
LoIBot	IRC	2
PushBot	IRC	3
rBot	IRC	17
SDBot	IRC	24
SlefBot	IRC	7
SpyBot	IRC, P2P	23
Zbot	HTTP	14

Tabela 5.5 *Botnets* Detectadas por Meio de Assinaturas

A Tabela 5.3 apresenta um quadro comparativo com as funcionalidades dos principais IDS apresentados neste trabalha com o modelo proposto. Demonstrando que o modelo atinge um maior número de requisitos para a detecção de *Bots*.

Detecção	**Snort**	**MABDS**	**Botsniffer**	**BotMiner**	**BotGAD**	**Modelo**
Baseado em Assinatura	**Sim**	**Sim**	Não	Não	Não	**Sim**
Correlação Horizontal	Não	Não	**Sim**	**Sim**	**Sim**	**Sim**
HIDS	**Sim**	**Sim**	Não	Não	Não	**Sim**
NIDS	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**
Bot criptografado	Não	Não	**Sim**	**Sim**	**Sim**	**Sim**
Botnet Centralizada	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**	**Sim**
Botnet Descentralizada	Não	Não	Não	**Sim**	**Sim**	**Sim**
Bot Individual	**Sim**	**Sim**	Não	Não	Não	**Sim**
Atividade em Grupo	Não	Não	**Sim**	**Sim**	**Sim**	**Sim**

Tabela 5.6 Tabela comparativa de funcionalidade das ferramentas IDS

5.5 Considerações finais

Neste capítulo, foram apresentado a modelagem e implementação do protótipo para a arquitetura proposta, além dos testes realizados no ambiente do laboratório e os resultados encontrados. Os resultados dos testes demonstram a potencialidade do protótipo implementado quanto à sua utilização na detecção de *Botnet*. No próximo capítulo será apresentado a conclusão do projeto, bem como as sugestões de trabalhos futuros.

6 CONCLUSÃO

6.1 Contribuições deste trabalho

O modelo proposto possibilita a detecção de *Botnet*, quando utilizamos a técnica de detecção baseada em anomalia utilizando PSO e RNA, auxiliado pela filtragem de dados, o que facilita o processo de análise, eliminando para as demais análises pacotes previamente infectados.

A utilização da técnica de detecção por assinatura pode identificar *Bot* que operam isoladamente ou em conjunto com outros *Bots*, no entanto, está técnica só identifica *Bots* conhecidos, o que se faz necessário à utilização da técnica por anomalia.

A utilização de RNA se mostrou mais eficaz na detecção baseada em anomalia do que o PSO, pois possuem maiores critérios de definição de treinamento. No entanto a técnica de PSO facilita no processo de treinamento da rede, visto que é de fácil implementação, não precisa de um grande volume de dados para determinação de um perfil padrão, além de executar em menor intervalo de tempo.

As técnicas de detecção utilizadas podem detectar *Bots* do mundo real, independentemente da estrutura, portanto identifica ataques de *Bots* que utilizam estrutura dos protocolos IRC, HTTP e P2P.

O uso de detecção híbrida aprimora a ferramenta potencialmente, visto que as técnicas baseadas em assinatura e anomalia para detecção de *Bots* torna uma abordagem mais promissora para combater a ameaça contra novas *Botnets* no ecossistema *online* e computadores ativos.

6.2 Considerações finais

Com o desenvolvimento desse trabalho foi possível mostrar a potencialidade dos Sistemas de Detecção de Intrusos Híbridos, principalmente pelo uso de agentes para a defesa e a manutenção da segurança das redes de computadores. No entanto, apesar das vantagens apresentadas pela solução proposta é importante relatar a necessidade de um planejamento para a implantação da ferramenta, visto que, há inúmeros riscos e limitações dessa tecnologia.

Dentre as dificuldades existentes podemos destacar que o treinamento de Otimização por Enxame de Partícula e Redes Neurais Artificiais, para determinação do perfil,

deve ocorrer com máquinas no mundo real, e que as mesmas não estejam infectadas, para que não ocorra problemas quanto aos números coletados. Faz-se necessária avaliação do conteúdo do tráfego, devido que essas informações podem ampliar o poder de IDS utilizadas para a detecção de *Botnets*.

Contudo, a realização deste trabalho foi de significativa importância, pois possibilitou o aprofundamento nesta área de pesquisa que vem crescendo de maneira intensa, que permite unir esforços em várias partes do mundo em busca de soluções que a comunidade de segurança possa ter em mãos para enfrentar os desafios atuais e futuros, principalmente pelo crescente onde de ataques pelas *Botnets* em sistemas por todo o mundo.

6.3 Sugestões para trabalhos futuros

Para a continuidade deste trabalho e sugestão para trabalhos futuros nessa área, podemos citar:

- A implementação de agentes de coleta local de *Hosts* e análise nos próprios *Hosts*;
- Ampliação das variáveis de análise de anomalia para *Bots* e outros *malwares*;
- Integração da ferramenta com outros serviços como *Honeypots* e *Firewall*.
- O aprimoramento do agente de análise de modo que seja possível agregar o índice de severidade já presente no NIDIA e, com isso, gerar alertas mais precisos.
- Análise em Testes de RNA com dados treinados em PSO.
- Testes utilizando *Botnet* P2P.
- Inspeção do conteúdo do tráfego dos *Bots*.
- A criação de um agente de análise para a detecção do servidor C&C, para determinação do local oriundos dos ataques.

REFERÊNCIAS

[1] Maryam Feily, Alireza Shahrestani and Sureswaran Ramadass, "**A Survey of Botnet and Botnet Detection**", Emerging Security Information, Systems and Technologies, Third International Conference on Emerging Security Information, Systems and Technologies, Aug. 2009, pp. 268-273.

[2] Zhaosheng Zhu, Guohan Lu, Yan Chen, Zhi Judy Fu, Phil Roberts and Keesook Han, "**Botnet Research Survey**", Computer Software and Applications, 32nd Annual IEEE International, Aug. 2008, pp. 967-972.

[3] Craig A. Schiller, Jim Binkley, David Harley, Gadi Evron, Tony Bradley, Carsten Willems and Michael Cross. "**Botnets: The Killer Web App"**. Publisher: Syngress, Pub. Date: 2007.

[4] F.Sabahi and A.Movaghar. "**Intrusion Detection: A Survey**". Systems and Networks Communications, The Third International Conference on Systems and Networks Communications, Oct. 2008, pp. 23-26.

[5] Symantec.cloud. "**MessageLabs Intelligence: 2010 Annual Security Report**." Disponível em <http://www.messagelabs.com/>. Acesso em 15 de Janeiro/2011.

[6] CERT.br - O CENTRO DE ESTUDOS, RESPOSTA E TRATAMENTO DE INCIDENTES DE SEGURANÇA NO BRASIL. "**Estatísticas de spam**." Disponível em <http://www.cert.br/>. Acesso em Janeiro/2011.

[7] Miroslaw Szymczyk. "**Detecting Botnets in Computer Networks Using Multi-Agent Technology**". Dependability of Computer Systems, Fourth International Conference on Dependability of Computer Systems, July 2009, pp. 192-201.

[8] Hossein Rouhani Zeidanloo and Azizah Bt Abdul Manaf. "**Botnet Detection by Monitoring Similar Communication Patterns**". International Journal of Computer Science and Information Security, Vol. 7 No. 3, March 2010, pp. 36-45.

[9] Hossein Rouhani Zeidanloo, Mohammad Jorjor Zadeh shooshtari, Payam Vahdani Amoli, M. Safari and Mazdak Zamani. "**A Taxonomy of Botnet Detection Techniques**". 3rd IEEE International Conference on Computer Science and Information Technology, Computer Science and Information Technology, July 2010, pp. 158-162.

[10] Hailong Wang and Zhenghu Gong. "**Collaboration-based Botnet Detection Architecture**". Intelligent Computation Technology and Automation, Second International Conference on Intelligent Computation Technology and Automation, Oct. 2009, pp. 375-378.

[11] Jivesh Govil and Jivika Govil. "**Criminology of BotNets and their Detection and Defense Methods**". Electro/Information Technology, IEEE EIT 2007 Proceedings, May 2007, pp. 215-220.

[12] Wei Lu and Ali A. Ghorbani. **"Botnets Detection Based on IRC-Community"**. Global Telecommunications Conference, Dec. 2008, pp. 1-5.

[13] Claudio Mazzariello. **"IRC traffic analysis for botnet detection"**. The Fourth International Conference on Information Assurance and Security, Sept. 2008, pp.318-323.

[14] Jae-Seo Lee, HyunCheol Jeong, Jun-Hyung Park, Minsoo Kim and Bong-Nam Noh. **"The Activity Analysis of Malicious HTTP-based Botnets using Degree of Periodic Repeatability"**. International Conference on Security Technology, Dec.2008, pp.83-86.

[15] Su Chang, Linfeng Zhang, Yong Guan and Thomas E. Daniels. **"A Framework for P2P Botnets"**. WRI International Conference on Communications and Mobile Computing, Jan.2009, pp.594-599.

[16] Elizabeth Van Ruitenbeek and William H. Sanders. **"Modeling Peer-to-Peer Botnets"**. Fifth International Conference on Quantitative Evaluation of Systems, Sept.2008, pp.307-316.

[17] Yousof Al-Hammadi and Uwe Aickelin. **"Behavioural Correlation for Detecting P2P Bots"**. Second International Conference on Future Networks, Jan.2010, pp.323-327.

[18] Hyunsang Choi, Hanwoo Lee, Heejo Lee and Hyogon Kim. **"Botnet Detection by Monitoring Group Activities in DNS Traffic"**. 7th IEEE International Conference on Computer and Information Technology, Oct.2007, pp.715-720.

[19] Teodoro P. García, Verdejo J. Díaz, Fernández G. Maciá, E. Vázquez. **"Anomaly-based network intrusion detection: Techniques, systems and challenges"**. Computers & Security 28, 2009, pp.18-28.

[20] F.Sabahi and A.Movaghar. **"Intrusion Detection: A Survey"**. The Third International Conference on Systems and Networks Communications, Oct. 2008, pp. 23-26.

[21] Moses Garuba, Chunmei Liu, and Duane Fraites. **"Intrusion Techniques: Comparative Study of Network Intrusion Detection Systems"**. Fifth International Conference on Information Technology: New Generations, April 2008, pp. 592-598.

[22] Farooq Anjum and Petros Mouchtaris. **"Security for wireless ad hoc networks"**. Publisher: John Wiley & Sons, Pub. Date: 2007.

[23] Falkner de Arêa Leão Moraes. "**Segurança e Confiabilidade em IDS baseados em Agentes**", Dissertação (mestrado). UFMA, 2009.

[24] Andrew R. Baker, Brian Caswell and Mike Poor. **"Snort 2.1 Intrusion Detection, Second Edition"**. Publisher: Syngress, Pub. Date: 2004.

[25] Jako Fritz. "**Hybrid Intrusion detection network monitoring with honeypots**". Dissertação (mestrado). University of Twente, 2011.

[26] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández and E. Vázquez. **"Anomaly-based network intrusion detection: Techniques, systems and challenges"**. Computers & Security, 2009.

[27] Mohammad Reza Norouzian and Sobhan Merati. **"Classifying Attacks in a Network Intrusion Detection System Based on Artificial Neural Networks"**. 13th International Conference on Advanced Communication Techonology (ICACT), Fev. 2011, pp. 868-873.

[28] Laurene V. Fausett. "**Fundamentals of neural networks:architectures, algorithms, and applications**". Publisher: Prentice-Hall, Pub. Date: 1994.

[29] Kishan Mehrotra, Chilukuri K. Mohan and Sanjay Ranka. "**Elements of Artificial Neural Networks**". Publisher: Bradford Books, Pub. Date: 1996.

[30] Ricardo Luis da Rocha Ataide. "**Uma arquitetura para a detecção de intrusos no ambiente wireless usando redes neurais**". Dissertação (mestrado). UFMA, 2007.

[31] Enn Tyugu. **"Artificial Intelligence in Cyber Defense"**. 3rd International Conference on Cyber Conflict (ICCC), June 2011, pp. 1-11.

[32] Alissson Marques da Silva. "**Utilização de Redes Neurais Artificiais para a Classificação de SPAM**". Dissertação (mestrado). CEFET-MG, 2009.

[33] Fox K., Henning R., Reed J., Simonian, R. **"A neural network approach towards intrusion detection"**. In: 13th National Computer Security Conference, 1990, pp. 124-134.

[34] Marcel Waintraub. **"Algoritmos Paralelos de Otimização por Enxame de Partículas em Problemas Nucleares"**. Tese (doutorado). UFRJ, 2009.

[35] Xiaohui Hu. **"Particle Swarm Optimization"**. Disponível em <http://www.swarmintelligence.org/>. Acesso em Novembro/2010.

[36] Bassel Soudan and Mohamed Saad. **"An Evolutionary Dynamic Population Size PSO Implementation"**. 3rd International Conference on Information and Communication Technologies: From Theory to Applications, April 2008, pp. 1-5.

[37] Chunming Yang and Dan Simon. **"A New Particle Swarm Optimization Technique"**. 18th International Conference on Systems Engineering, Aug. 2005, pp.164-169.

[38] K. Y. Lee and Jong-Bae Park. **"Application of Particle Swarm Optimization to Economic Dispatch Problem: Advantages and Disadvantages"**. 2006 IEEE PES Power Systems Conference and Exposition, Oct. 2006, pp. 188-192.

[39] Zhang Vi and Zhang Li-Jun. **"A Rule Generation Model Using S-PSO for Misuse Intrusion Detection"**. 2010 International Conference on Computer Application and System Modeling, Oct. 2010, pp. V3-418 – V3-423.

[40] Jun Wang, Xu Hong, Rong-rong Ren and Tai-hang Li. **"A Real-time Intrusion Detection System Based on PSO-SVM"**. 2009 International Workshop on Information Security and Application. Nov. 2009, pp.319-321.

[41] G. GU, J. ZHANG, W. LEE. **"BotSniffer: detecting botnet command and control channels in network traffic"**. Annual Network and Distributed System Security Symposium, 2008.

[42] João Marcelo Ceron. **"Arquitetura Distribuída e Automatizada para Mitigação de *Botnet* Baseada em Análise Dinâmica de *Malwares*"**. Dissertação (mestrado). UFRGS, 2010.

[43] G. Gu, R. Perdisci, J. Zhang, and W. Lee, **"Botminer: Clustering analysis of network traffic for protocol- and structure independent Botnet detection"**. 17th USENIX Security Symposium, Dec. 2008.

[44] Hyunsang Choi, Heejo Lee, and Hyogon Kim. **"BotGAD: Detecting Botnets by Capturing Group Activities in Network Traffic"**. Fourth International ICST

Conference on COMmunication System softWAre and middlewaRE - COMSWARE, June 2009, pp. 16-19.

[45] Nicholas Albright. **"Fundação Shadowserver"**. Disponível em <http://www.shadowserver.org/>. Acessado em Outubro/2010.

[46] A. Ramachandran, N. Feamster, and S. Vempala. **"Filtering spam with behavioral blacklisting"**. In Proceedings of the 14th ACM conference on computer and communications security, Oct. 2007.

[47] B. Kim, H. Kim, S. Bahk. **"NSF : Network-based Spam Filtering based on On-line Blacklisting against Spamming Botnets"**, IEEE Global Telecommunications Conference. GLOBECOM 2009, Nov. 2009, pp.1-6.

[48] S.H. Kim, S.Y. Park, S.G. Kang. **"PIM Multicast Spam Countering Method Using Blacklist in Rendezvous Point"**. In International Conference on Advanced Communication Technology, Feb. 2008, pp. 539-543.

[49] A. Ramachandran, D. Dagon, and N. Feamster. **"Can DNS based blacklists keep up with bots"**. In Conference on Email and Anti-Spam, 2006.

[50] Zhen Li, Jun-Feng Tian and Feng-Xian Wang. **"Sandbox System Based on Role and Virtualization"**. International Symposium on Information Engineering and Electronic Commerce, May 2009, pp. 342-346.

[51] Katsunari Yoshioka, Yoshihiko Hosobuchi, Tatsunori Orii and Tsutomu Matsumoto. **"Vulnerability in Public Malware Sandbox Analysis Systems"**. 10th Annual International Symposium on Applications and the Internet, July 2010, pp. 265-268.

[52] Christiane Ferreira Lemos Lima. **"Agentes Inteligentes para Detecção de Intrusos em Redes de Computadores"**. Dissertação (mestrado). UFMA, 2002.

[53] S. Staniford-Chen. **"Common Intrusion Detection Frameowork"**. Computer Emergency Response Team, Sept. 1999.

[54] I. Hegazy, T. Alarif, Z. T. Fayed and H. M. Fahim. **"A framework for multiagent-based system for intrusion detection"**. In The Proceedings of the 3rd International Conference in Intelligent Systems Design and Applications, Aug.2003, pp.117-126.

[55] Lindonete Gonçalves Siqueira. and Zair Abdelouahab. **"A fault tolerance mechanism for network intrusion detection system based on intelligent agents (nidia)"**. In 3rd Workshop on Software Technologies for Future Embedded & Ubiquitous Systems. Dissertação (mestrado). UFMA, 2006.

[56] Aline Lopes da Silva. **"Modelo de IDS para Usuários de Dispositivos Móveis"**. Dissertação (mestrado). UFMA, 2008.

[57] Joseph Schmuller. **"Sams Teach Yourself UML in 24 Hours"**. Third Edition. Publisher: Sams Publishing, Date: 2004.

[58] Dan Pilone and Neil Pitman. **"UML 2.0 in a Nutshell"**. Publisher: O'Reilly Publishing, Date: 2005.

[59] Gilleanes T. A. Guedes. **"UML 2: Uma abordagem prática"**. Publisher: Novatec Editora. São Paulo, Date: 2009.

[60] Tim Downey. **"Web Development with Java: Using Hibernate, JSPs and Servlets"**. Publisher: Syngress, Date: 2007.

[61] **"Wincap"**. Disponível em <http://www.winpcap.org/>. Acessado em Julho de 2010.

[62] **"SQL Server"**. Disponível em <http://www.microsoft.com/sqlserver/2008/pt/br/default.aspx>. Acessado em Julho de 2010.

[63] **"Infiltrated.net"**. Disponível em <http://www.infiltrated.net/>. Acessado em Novembro/2010.

[64] **"Lashback"**. Disponível em <http://www.lashback.com/>. Acessado em Novembro/2010.

[65] **“Spamlinks”**. Disponível em <http://spamlinks.net/filter-bl.htm>. Acessado em Dezembro/2010.

[66] **“Mainsleaze Spam”**. Disponível em <http://mainsleazespam.com/>. Acessado em Dezembro/2010.

[67] Fernando Augusto Pestana Júnior. **“Proposta de Atualização Automática dos Sistemas de Detecção de Intrusos por meio de Web Services”**. Dissertação (mestrado). UFMA, 2005.

[68] Emanuel Costa Claudino Silva. **“Gerenciamento e Integração das Bases de Dados de Sistemas de Detecção de Intrusos”**. Dissertação (mestrado). UFMA, 2006.

[69] COMODO. **“Creating Trust Online”**. Disponível em <http://camas.comodo.com/>. Acesso em Maio/2004.

[70] **“GFI Labs”**. Disponível em <http://www.sunbeltsecurity.com/sandbox/>. Acesso em Junho/2004.

[71] **“ThreatExpert”**. Disponível em <http://www.threatexpert.com/>. Acesso em Junho/2004.

[72] C. L. Lucchesi. **“Introdução à Criptografia Computacional”**. Publisher: Editora Papirus/UNICAMP, Date:1986.

[73] J. Franco and E. Kerr. **“Criptologia: Protegendo a Informação na Era da Informação”.** Publisher: ConteXto, nº 6, Pub. Date: mar. 1995.

[74] **“VirtualBox”**. Disponível em < http://www.virtualbox.org/>. Acesso em Junho de 2010.

[75] **“NetCop Security”**. Disponível em <http://netcopsecurity.com/>. Acesso em Maio de 2010.

[76] **“mIRC”**. Disponível em <http://www.mirc.com/>. Acesso em Junho de 2010.

[77] Neal Krawetz. **“Introdution to Network Security”**. Publisher: Charles River Media, Pub. Date: 2007.

[78] **“Botnet Zeus”**. Disponível em < http://www.easy-share.com/1915034965/zeuss.rar/>. Acesso em Outubro de 2010.

[79] **“Xampp”**. Disponível em < http://www.apachefriends.org/pt_br/xampp.html/>. Acesso em Outubro de 2010.