Algoritmos Imunoinspirados Aplicados em Segurança Computacional:

Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores

Joaquim Quinteiro Uchôa

Joaquim Quinteiro Uchôa

Algoritmos Imunoinspirados Aplicados em Segurança Computacional:

Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores

Tese de Doutorado apresentada ao Doutorado em Bioinformática da UFMG, como parte dos requisitos para obtenção do título de Doutor em "Bioinformática".

Orientador:
Prof. Dr. Walmir Matos Caminhas (UFMG)

Co-Orientador:
Prof. Dr. Tomaz Aroldo da Mota Santos (UFMG)

Belo Horizonte
Doutorado em Bioinformática – UFMG
2009

Este documento foi produzido em LaTeX, utilizando-se fontes Bookman no texto e Avant Garde nos títulos. Os títulos dos capítulos utilizaram ainda fonte CBGreek. As capitulares foram feitas utilizando o pacote Lettrine e a fonte Zap Chancery. Para referências conforme normas da ABNT, foi utilizado o pacote ABNTeX.

Uchôa, Joaquim Quinteiro

Algoritmos Imunoinspirados Aplicados em Segurança Computacional: Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores. -- Belo Horizonte: UFMG, 2009.

245 p. : il.

Tese (doutorado) UFMG/Bioinformática.

1. Segurança Computacional 2. Sistemas Imunes Artificiais 3. Segurança Computacional 4. Imunoinformática. 5. Imunologia. I. Título.

-

Universidade Federal de Minas Gerais
Instituto de Ciências Biológicas
Programa de Doutorado em Bioinformática
Avenida Presidente Antônio Carlos, 6627 – Pampulha
31270-901 - Belo Horizonte – MG
Página Eletrônica: www.pgbioinfo.icb.ufmg.br
Endereço eletrônico: bioinfo@icb.ufmg.br – Telefax: 55 31 3409-2554

ATA DA DEFESA DA TESE DE DOUTORADO DE JOAQUIM QUINTEIRO UCHÔA. Aos sete dias do mês de maio de 2009 às 14h00min, reuniu-se no Instituto de Ciências Biológicas da Universidade Federal de Minas Gerais a Comissão Examinadora da tese de doutorado, indicada no dia dezesseis de março de 2009, durante a 60ª reunião do Colegiado do Programa, para julgar, em exame final, o trabalho intitulado "Algoritmos Imunoinspirados Aplicados em Segurança Computacional: Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores", requisito final para a obtenção do grau de Doutor em Ciências, Área de Concentração: Bioinformática. Abrindo a sessão o Presidente da Comissão, Prof. Dr. Walmir Matos Caminhas da Universidade Federal de Minas Gerais, após dar a conhecer aos presentes o teor das Normas Regulamentares do Trabalho Final, passou a palavra ao candidato para apresentação de seu trabalho. Seguiu-se a argüição pelos examinadores, com a respectiva defesa do candidato. Logo após a Comissão se reuniu sem a presença do candidato e do público para julgamento e expedição do resultado final. Foram atribuídas as seguintes indicações: Profª Drª Ana Maria Caetano de Faria da Universidade Federal de Minas Gerais, Belo Horizonte, MG, aprovado; Prof. Dr. Wagner Meira Júnior, Belo Horizonte, MG, aprovado; Prof. Dr. Frederico Gadelha Guimarães da Universidade Federal de Ouro Preto, Ouro Preto, MG, aprovado; Prof. Dr. Leandro Nunes de Castro Silva da Universidade Presbiteriana Mackenzie, São Paulo, SP aprovado; Prof. Dr. Tomaz Aroldo da Mota Santos, co-orientador, da Universidade Federal de Minas Gerais, Belo Horizonte, MG, aprovado; Prof. Dr. Walmir Matos Caminhas, orientador, da Universidade Federal de Minas Gerais, Belo Horizonte, MG, aprovado. Pelas indicações o candidato foi considerado APROVADO. O resultado final foi comunicado publicamente ao candidato pelo Presidente da Comissão. Nada mais havendo a tratar o Presidente da Comissão encerrou a reunião e lavrou a presente ata que será assinada por todos os membros participantes da Comissão Examinadora. Belo Horizonte, aos sete dias de maio de 2009.

Profª Drª Ana Maria Caetano de Faria – UFMG

Prof. Dr. Wagner Meira Júnior – UFMG

Prof. Dr. Frederico Gadelha Guimarães – UFOP

Prof. Dr. Leandro Nunes de Castro Silva – MACKENZIE

Prof. Dr. Tomaz Aroldo da Mota Santos – co-orientador – UFMG

Prof. Dr. Walmir Matos Caminhas – orientador – UFMG

O poema não tem palavras suficientes para o poema.

O poeta é o que resta.

(Transubstanciação, 1997)

para Natan e Sofia,
que seus caminhos sejam repletos de luz

Agradecimentos

Agradeço inicialmente à cafeína, via café e ao guaraná ralado, por ter permitido a concentração em temas abstratos e complexos, mesmo em momentos inoportunos.

Agradeço também àqueles que produziram música que foi ouvida por mim durante a implementação do código ou escrita do texto, especialmente Bach e as bandas Tool, Explosion in The Sky, King Crimson, Porcupine Tree, New Order e Kraftwerk.

Também agradeço aos inúmeros poetas, filósofos e pensadores que contribuíram para minha formação mental, em especial Heráclito, Sócrates, Nietzsche, Rimbaud, Walt Whitman, Fernando Pessoa, Sidarta e Lao Tse.

Agradeço à Vida em si, pela oportunidade que tive de discutir diferentes visões do mundo. Agradeço à Beleza, especialmente a das mulheres, pela ímpeto à vida em si.

Agradeço aos diversos autores com os quais pude "dialogar" direta ou indiretamente na confecção deste trabalho. Especiais agradecimentos para Leandro Nunes de Castro, Jonathan Timmis, Irun Cohen e Nelson Vaz.

Agradeço aos professores Walmir Matos Caminhas, Tomaz Aroldo da Mota Santos e ao ex-aluno de Engenharia Elétrica Thiago Pereira Guzella, pelo apoio durante a realização desse trabalho.

Agradeço à CAPES, pelo apoio ao projeto, através do programa PQI/CAPES. E ao DCC/UFLA pela oportunidade de afastamento para o doutorado.

Por fim, agradeço e peço humildes desculpas aos amigos e familiares que tiveram-me ausente parcial ou totalmente em diversos momentos durante esse doutoramento.

Resumo

A evolução da Internet e a contínua interconexão de computadores e outros dispositivos digitais têm criado uma série de oportunidades nas mais diversas áreas de atuação humana. Entretanto, esse processo trouxe consigo uma série de problemas, entre eles o crescente número de sistemas computacionais invadidos por intrusos. O trabalho aqui apresentado aborda esse problema utilizando metodologias e algoritmos inspirados em conceitos e processos do sistema imune inato e adaptativo. Dois algoritmos foram propostos, IA-AIS e DT-AIS, e testados em bases de dados coletadas durante o trabalho, simulando um computador em uso normal e sob ataque. O IA-AIS também foi avaliado com o KDD99, uma base pública de dados sobre intrusão, já relativamente defasada mas muito utilizada em testes comparativos. Os resultados obtidos por ambos algoritmos foram extremamente bons, apontando como bastante adequada a aplicação de metáforas mais completas sobre o funcionamento do sistema imune no problema de deteção de intrusos. Mais ainda, esses resultados mostraram os algoritmos como extremamente promissores em problemas de detecção de anomalias, apontando para um grande potencial de implementação dos mesmos em sistemas de produção.

Palavras-Chave: Segurança Computacional, Detecção de Intrusos em Redes de Computadores, Imunoinformática, Sistemas Imunes Artificiais, Sistema Imune Adaptativo.

Abstract

The evolution of Internet and the continuous interconnection of computers and other digital devices has opened many opportunities. On the other hand, this process has brought a lot of problems and pitfalls. One of these problems refers to the increasing number of computers being invaded by intruders. The work presented in this text approaches this problem by adopting strategies and algorithms inspired by concepts and processes of the inate and adaptive immune system. We have proposed two algorithms, IA-AIS and DT-AIS, and they were evaluated in datasets simulating a computer in normal use and under attack. IA-AIS was also evaluated with KDD99, a publicly available dataset used in the literature to comparative analysis. The obtained results were very good, indicating the adequacy of the use of more complete metaphors of the immune system in the intrusion detection problem. Moreover, these results showed the algorithms as very promising tools in anomaly detection problems, indicating their great potential application in production systems.

Keywords: Computer Security, Network Intrusion Detection, Artificial Immune Systems, Immunoinformatics, Adaptive Immune System.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

ACK	*Acknowledgement*
ACM	*Association for Computing Machinery*
ACTP	*Advanced Computer Techniques Project*
ADSL	*Asymmetric Digital Subscriber Line*
AG	*Algoritmo Genético*
AIM	*AOL Instant Mensager*
AJAX	*Asynchronous JAvascript and XML*
APC	*Antigen-Presenting Cell*
ARP	*Address Resolution Protocol*
ARPA	*Advanced Research Projects Agency*
ARPANET	*Advanced Research Projects Agency Network*
ASCII	*American Standard Code for Information Interchange*
BCR	B Cell Receptor
BN	*Backbone Network*
BSD	*Berkeley Software Distribution*
C/S	*Cliente-servidor*
CD	*Cluster of Differentiation*
CD4	*Cluster of Differentiation 4*

CD8	*Cluster of Differentiation 8*
cDc	*cult of Dead cow*
CE	*Computação Evolutiva*
CERN	*Centre European pour la Recherche Nucleaire*
CERT.br	*Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil*
CIDR	*Classless Interdomain Routing*
CIFS	*Common Internet File System*
CPDEE	*Centro de Pesquisas e Desenvolvimento em Engenharia Elétrica*
CTL	*Cytotoxic T Lymphocyte*
CVE	*Common Vulnerabilities and Exposure*
CWR	*Congestion Window Reduced*
DARPA	*Defense Advanced Research Projects Agency*
DCA	*Defense Communications Agency*
DCA	*Dendritic Cell Algorithm*
DDoS	*Distributed Denial of Service*
DNA	DeoxyriboNucleic Acid
DNS	*Domain Name System*
DoS	*Denial of Service*
DSL	*Digital Subscriber Line*
DT-AIS	*Danger Theory Artificial Immune System*
ECE	*ECN Echo*
ECN	*Explicit Congestion Notification*
EGP	*Exterior Gateway Protocol*
FAPESP	*Fundação de Amparo à Pesquisa do Estado de São Paulo*
FIN	*Finalize*

FTP	*File Transfer Protocol*
HIDS	*Host-Based Intrusion Detection System*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
IA-AIS	*Innate and Adaptive Artificial Immune System*
IANA	*Internet Assigned Numbers Authority*
ICMP	*Internet Control Message Protocol*
ICQ	*o nome vem da expressão inglesa "I seek you"*
IDS	*Intrusion Detection System*
IEEE	*Institute of Electrical and Electronics Engineers*
IETF	*Internet Engineering Task Force*
IGP	*Interior Gateway Protocol*
IHL	*Internet Header Length*
IMAP	*Internet Message Access Protocol*
IMP	*Interface Message Processors*
IP	*Internet Protocol*
IPTO	*Information Processing Techniques Office*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
IRC	*Internet Relay Chat*
ISN	*Initial Sequence Number*
ISO	*International Organization for Standardization*
KDD99	*Refere-se à base de dados utilizada na "The Fifth International Conference on Knowledge Discovery and Data Mining", em 1999; também chamada de "DARPA/KDD-99" ou "KDD99 Dataset"*
LAN	*Local Area Network*

LF	*Lógica Fuzzy*
LNCC	*Laboratório Nacional de Computação do Rio de Janeiro*
LSI	*Laboratório de Sistemas Integráveis*
MAC	*Media Access Control*
MAN	*Metropolitan Area Network*
MHC	*Major Histocompatibility Complex*
MILNET	*Military Network*
MIME	*Multipurpose Internet Mail Extensions*
MIQ	*Machine Intelligence Quotient*
MIT	*Massachusetts Institute of Technology*
MODEM	*MOdulator-DEModulator*
MPEG	*Moving Picture Experts Group*
MPLS	*Multi Protocol Label Switching*
MSN	*Microsoft Network*
NASA	*National Aeronautics and Space Administration*
NAT	*Network Address Translation*
NCP	*Network Control Protocol*
NFS	*Network File System*
NIDS	*Network-Based Intrusion Detection System*
NK	*Natural Killer*
NSF	*National Science Foundation*
NSFNET	*National Science Foundation Network*
NTP	*Network Time Protocol*
OLPC	*One Laptop per Child*
OSI	*Open Systems Interconection*

P2P	*Peer to Peer*
PAMP	*Pathogen-Associated Molecular Patterns*
PC	*Personal Computer*
POP	*Post Office Protocol*
PPP	*Point-to-Point Protocol*
PPPoE	*Point-to-Point Protocol over Ethernet*
PRR	*Pattern Recognition Receptors*
PSH	*Push function*
RBF	*Radial Basis Function*
RFC	*Request For Comments*
RNA	*Redes Neurais Artificiais*
RNP	*Rede Nacional de Pesquisa*
RPC	*Remote Procedure Call*
RST	*Reset*
SAGE	*Semi-Automatic Ground Environment*
SCTP	*Stream Control Transmission Protocol*
SIAs	*Sistemas Imunes Artificiais*
SIP	*Session Initiation Protocol*
SMB	*Server Message Block*
SMTP	*Simple Mail Transfer Protocol*
SRI	*Stanford Research Institute*
SSH	*Secure SHell*
SSL	*Secure Sockets Layer*
SYN	*Synchronize*
TCP	*Transfer Control Protocol*

TCP/IP	*Pilha de protocolos da Internet, composta pelos protocolos TCP (Transfer Control Protocol) e IP (Internet Protocol)*
TCR	T Cell Receptor
TI	*Tecnologia da Informação*
TICs	*Tecnologias da Informação e Comunicação*
TLR	*Toll Like Receptor*
TLS	*Transport Layer Security*
ToS	*Type of Service*
TTL	*Time to Live*
UCB	*University of California at Berkeley*
UCE	*Unsolicited Commercial E-mail*
UCLA	*University of California at Los Angeles*
UCSB	*University of California at Santa Barbara*
UDP	*User Datagram Protocol*
UFLA	*Universidade Federal de Lavras*
UFMG	*Universidade Federal de Minas Gerais*
UFRJ	*Universidade Federal do Rio de Janeiro*
URG	*Urgent*
USB	*Universal Serial Bus*
USP	*Universidade de São Paulo*
UU	*University of Utah*
VoIP	Voice over IP – Voz sobre IP
VPN	*Virtual Private Network*
WAN	*Wide Area Network*
WEP	*Wired Equivalent Privacy*
WPA	*Wi-Fi Protected Access*

XDR	*eXternal Data Representation*
XML	*eXtensible Markup Language*
XMPP	*eXtensible Messaging and Presence Protocol*

CAPÍTULO 1

Introdução

E a vida prossegue
sem nem ao menos sabermos
se são leis ou começos
os princípios do mundo.

Princípios X, 1990

Quando a vida bate à porta
é difícil resistir
e não abraçar o sol.

In*: Versos para Sussurro, 1990*

1.1 Contextualização

RECENTEMENTE tem-se assistido a grandes avanços no uso de algoritmos e modelos computacionais para resolução de problemas biológicos. Isso fomentou o surgimento e crescimento de pesquisas nas áreas de Bioinformática e Biologia Computacional, sendo cada vez mais frequente o surgimento de grupos e eventos nacionais e internacionais ligados ao assunto. Como comentado em (KITANO, 2002), a Biologia Computacional possui dois ramos distintos: 1) descoberta de conhecimento e mineração de dados em dados experimentais e 2) análise baseada em simulação, que testa hipóteses bioló-

gicas em ambientes de silício. A Bioinformática, por sua vez, possui raízes na Ciência da Computação, na Estatística e na Biologia Molecular, tendo surgido principalmente para enfrentar problemas e dificuldades surgidas com o sequenciamento genético de diversas espécies, inclusive a humana.

Em uma visão mais restrita, a Bioinformática está, portanto, ligada à Biologia Molecular, estando preocupada com o processamento de dados, objetivando a distinção e previsão de genes, configuração e funcionamento de proteínas, ações enzimáticas e problemas correlatos. Esses estudos são motivados pelos resultados das iniciativas em sequenciamento genético, e a quantidade de dados gerados sobre proteínas, DNA e RNA. Em uma visão mais ampla, Bioinformática é o estudo da aplicação de modelos, algoritmos e técnicas computacionais e matemáticas à geração, análise e gerenciamento de informação biológica ou biomédica, usando para isso, além da Biologia, Matemática/Estatística e Ciência da Computação, conhecimentos da Química e da Física. Nesse sentido, mais que uma ciência interdisciplinar, a Bioinformática é trandisciplinar, no sentido em que perpassa por várias áreas do conhecimento científico, indo muito além da mera aplicação de uma área em outra.

Os avanços na própria Biologia, por outro lado, suscitaram uma série de pesquisas sobre o potencial de desenvolvimento de métodos computacionais inspirados em modelos naturais para a resolução de diversos problemas em diversas áreas do conhecimento. A motivação para esse tipo de abordagem vem do fato que, em diversas situações, a natureza consegue resolver problemas complexos utilizando mecanismos relativamente simples, mas com grande poder de solução quando combinados. Por outro lado, em geral esses problemas ou não possuem uma solução computacional exata ou a mesma é ineficiente, e o modelo biológico propicia uma visão sobre como o problema poderia ser resolvido, utilizando elementos computacionais inspirados biologicamente.

Em geral, os estudos inspirados em modelos biofísicos estão intimamente ligados à área de Inteligência Computacional, existindo inúmeras pesquisas de modelagem de sistemas biológicos em computadores e a posterior utilização desses modelos para resolução de problemas diversos em Engenharia e Computação. Exemplos clássicos dessa abordagem são os Algoritmos Genéticos, apresentados em (MITCHELL, 1996), (HOLLAND, 1992) e (ZEBULUM; PACHECO; VELLASCO, 2002). Outro exemplo são as Redes Neurais Artificiais, apresentadas em (FAUSETT, 1994), (BRAGA; CARVALHO; LUDERMIR, 2000a) e (HAYKIN, 2001). Exemplos de estudos envolvendo simulações do funciona-

mento de redes neurais biológicas podem ser encontrados em (CHURCHLAND; SEJNOWSKI, 1996) e (DURBIN; MIALL; MITCHISON, 1991).

Dessa maneira, a Biologia tem inspirado o desenvolvimento de uma série de algoritmos, métodos e técnicas computacionais, funcionando como uma metáfora em busca de solução de problemas para os quais métodos tradicionais não são capazes de solucionar o problema em tempo adequado. A eficiência não é, entretanto, a única motivação para essa estratégia, uma vez que diversos problemas trazem grande dificuldade para o processo de modelagem ou até mesmo não permitem uma modelagem adequada para uso em sistemas baseados em algoritmos clássicos. Essa dificuldade em geral surge da existência de não-linearidade entre as variáveis do sistema.

Esse movimento tem possibilitado o surgimento de uma área da Computação, que tem sido denominada por alguns grupos de pesquisadores como Computação Bioinspirada[1] ou Computação Natural[2]. Essa área tenta mapear mecanismos biológicos de solução de determinados problemas no desenvolvimento de novos modelos e algoritmos para resolução de problemas computacionais. Dado que a Biologia é utilizada como inspiração e metáfora, muitos desses algoritmos acabam depois por distanciar de seu referencial biológico, ou possuírem apenas um tênue contato com a realidade biológica em si.

Uma área da Computação Bioinspirada que tem chamado a atenção recentemente é a Engenharia Imunológica. Por Engenharia Imunológica entende-se "(...) uma estrutura formal para o desenvolvimento de sistemas imunológicos artificiais" (DE CASTRO, 2001). Como comentado nesse mesmo texto, os Sistemas Imunes Artificiais (SIAs) surgiram a partir das tentativas de modelar e aplicar princípios imunológicos no desenvolvimento de novas ferramentas computacionais. Detalhes sobre SIAs podem ser encontradas em (DE CASTRO; VON ZUBEN, 1999) e (DE CASTRO; VON ZUBEN, 2000a). Entre inúmeras aplicações dos SIAs, podem ser citadas: reconhecimento de padrões (GREENSMITH; AICKELIN; CAYZER, 2005) (DE CASTRO; TIMMIS, 2002c) (WILSON; BIRKIN; AICKELIN, 2007), clusterização de dados (DE CASTRO; VON ZUBEN, 2000b), aprendizado (DE CASTRO; VON ZUBEN, 2002), mineração de dados (FREITAS; TIMMIS, 2007), otimização (KIM; LEE, 2004) (DE CASTRO; TIMMIS, 2002a) (BATISTA, 2009), etc.

[1] Ver, por exemplo o Laboratório de Computação Bioinspirada da USP (`http://www.icmc.usp.br/~biocom/`) ou o Journal of Bio-Inspired Computation Research (`http://www.ripublication.com/jbicr.htm`).

[2] Ver por exemplo (DE CASTRO, 2006a).

1.2 Motivação

Os usuários de computadores têm-se deparado cada vez mais com problemas sérios de segurança em redes de computadores. A internet tem-se tornado um "celeiro" para vírus de computadores e são cada vez mais comuns as notícias de invasão de *sites*. Existem várias ferramentas para detecção de intrusos no mercado de Informática, mas poucas incorporam técnicas de aprendizado que permitam o aprimoramento automático. As ferramentas disponíveis atualmente em geral apresentam problemas de desempenho ou de eficiência e são baseadas principalmente em comparação de assinaturas com dados de pacotes de redes ou registros do sistema. Um maior desempenho do sistema só é possível com uma grande base de dados de assinaturas de ataques, mas com um custo alto em termos de eficiência. Também é problemático o grande número de falsos positivos e negativos gerados por essas ferramentas.

Uma motivação pessoal deste pesquisador para com esse trabalho foi a experiência como administrador de servidores e laboratórios e o consequente contato com problemas diversos de Segurança Computacional. Foi possível observar diversas tentativas de ataques aos servidores sob minha responsabilidade, tanto por usuários externos como internos. Assim, este trabalho tem uma grande motivação prática, que é a obtenção de solução para um problema bastante comum em redes de computadores. Apesar de existirem ferramentas que dificultam em muito o processo de intrusão, cada vez mais esse processo é automatizado, exigindo pouco conhecimento técnico por parte do invasor. Assim, é necessário o desenvolvimento de ferramentas que auxiliem o administrador de redes no processo de detecção de intrusos, de preferência em tempo real, quando ainda é possível evitar o sucesso da invasão.

A utilização de algoritmos imunoinspirados, por sua vez, advém da grande eficiência do sistema imune humano como resposta às infecções, quando comparado aos modelos computacionais de prevenção de intrusão. Dessa maneira, durante a preparação para a saída para doutoramento, a leitura de (DE CASTRO, 2001) possibilitou o contato com a área de SIAs e o questionamento sobre o potencial de uso dessa abordagem em Segurança Computacional. Esse questionamento foi ampliado com a leitura inicial de alguns trabalhos com aplicações de SIAs nessa área, com alguns resultados promissores, mas sem ainda aplicações práticas. Tendo por vista a atividade do sistema imune de resposta a elementos infecciosos, surgiu a perspectiva de proposta e uso de

modelos computacionais baseados em conceitos biológicos para detecção de intrusos em redes de computadores.

Durante o contato inicial com a área de SIAs, foi possível perceber, portanto, que havia ainda um rico campo a ser explorado, na busca de modelos computacionais inspirados biologicamente para a resolução de um problema complexo em Computação. A relevância desse tema pôde ser verificada em vários trabalhos em temas correlatos: em (KEPHART, 1994), por exemplo, são propostos modelos computacionais para combate a vírus de computadores utilizando-se a abordagem dos SIAs. Outras pesquisas semelhantes são apresentadas em (DASGUPTA; ATTOH-OKINE, 1997) e (DE CASTRO; VON ZUBEN, 2000a). Essas leituras ajudaram a fomentar o tema da pesquisa apresentada neste texto: formular modelos e algoritmos inspirados no funcionamento do sistema imune para detecção de intrusos em redes de computadores

Uma motivação paralela para o trabalho surgiu a partir do interesse em melhor compreender o funcionamento do sistema imune, principalmente sob uma ótica de sistemas cognitivos. Nessa visão, o sistema imune é considerado um elemento ativo para o processo de aquisição e armazenamento de conhecimento, possuindo fortes similaridades com Redes Neurais Artificiais e Algoritmos Genéticos. Dado o interesse particular em compreender como o ser humano apreende e armazena conhecimento, apontou-se a possibilidade em verificar, ao menos em uma pequena parte, como o sistema imune contribui para esse processo.

1.3 Objetivos

Este texto tem o objetivo de apresentar os resultados de pesquisa e desenvolvimento e análise de ferramentas de segurança computacional baseadas em algoritmos imunoinspirados. Especificamente, buscou-se obter com este trabalho a formulação e validação de modelos e algoritmos inspirados no funcionamento do sistema imune para desenvolvimento de ferramentas de detecção de intrusos em redes TCP/IP. Apesar do grande potencial do uso da metáfora do sistema imune na aplicação do problema, isso não é suficiente para a obtenção de resultados teóricos e práticos que validem esse tipo de abordagem. Dessa maneira, este trabalho teve como objetivos específicos:

1. Verificar a possibilidade do uso do sistema imune como metáfora para o problema de detecção de intrusos em redes de computadores. Aqui,

houve a preocupação em avaliar a validade da inspiração biológica como base teórica para a formulação de modelos e algoritmos computacionais para detecção de intrusão.

2. Propor modelos de utilização de metáforas imunológicas, adequando os conceitos biológicos ao problema computacional. Uma questão pontual que existe quando da utilização de uma metáfora biológica em um problema de Computação ou Engenharia é a adequação dos conceitos e a extensão do poder da metáfora em si. Existem óbvios limites para a metáfora, que deve ser transposta para um modelo formal, utilizando-se diversas simplificações dos conceitos e processos biológicos. Além disso, há a necessidade de modelar os conceitos no problema, que precisa ser enxergado sob essa ótica metodológica. Assim, neste trabalho, houve o objetivo de mapear elementos do problema de detecção de intrusos em uma contrapartida biológica, respondendo a questões, por exemplo, sobre o que seria um antígeno no contexto de detecção de intrusos.

3. Desenvolver algoritmos computacionais inspirados em modelos imunológicos e que contivessem elementos que pudessem suprir as deficiências de algoritmos já existentes na literatura, mas ainda com problemas para aplicação em segurança em redes de computadores. A maior parte dos trabalhos correlatos a que foi possível ter acesso durante a realização deste trabalho possuíam problemas de eficiência ou desempenho quando aplicados a ambientes mais reais de intrusão. Dessa maneira, surgiu a necessidade de propor alternativas que contivessem elementos intrínsecos para um melhor resultado efetivo.

4. Utilizar uma abordagem mais "biológica" quando do desenvolvimento de modelos e algoritmos computacionais, inspirando-se em uma visão cognitiva do sistema imune. Esse objetivo foi inspirado na motivação de compreender um pouco mais o funcionamento do sistema imune e sua contribuição para a construção e armazenamento de conhecimento. Com isso, buscou-se neste trabalho a proposição de modelos e algoritmos mais "plausíveis" do ponto de vista imunológico, quando comparado à maior parte dos algoritmos e modelos existentes na literatura.

5. Desenvolver um *framework* para a implementação computacional dos modelos e algoritmos propostos, que servisse de base para estudos futuros tanto na proposta e uso de algoritmos imunoinspirados como também na modelagem do sistema imune. Para isso, havia a necessidade de criar

uma biblioteca de funções e objetos que permitissem uma rápida implementação e uso de modelos e algoritmos inspirados no sistema imune.

6. Implementar os algoritmos computacionais, avaliando o seu potencial efetivo para uso na resolução do problema de detecção de intrusos. Para tanto, após a implementação dos algoritmos, esses deveriam ser testados em bases de dados relacionadas à intrusão em redes de computadores. Para comparação com outras abordagens, os algoritmos deveriam ser testados com o KDD99, a única base pública na área conhecida por este pesquisador. Na ausência de bases mais recentes, fez parte das propostas do trabalho a geração de bases de dados com a simulação de diversos tipos de ataques.

1.4 Principais Contribuições do Trabalho

A maior contribuição desse trabalho é a proposição e análise inicial de dois algoritmos imunoinspirados, IA-AIS e DT-AIS, aplicados ao problema de detecção de intrusos. Os modelos incluem elementos do sistema imune inato e adaptativo, bem como um relacionamento entre esses elementos. Os modelos propostos obtiveram altas taxas de classificação correta, possuindo grande potencial para uso efetivo em ambientes computacionais. Os algoritmos em si são relativamente simples, apesar de embasados em uma visão mais biológica sobre o funcionamento do sistema imune, quando comparados aos algoritmos existentes na literatura.

Os resultados obtidos com as realizações dos testes com o IA-AIS e DT-AIS mostram a extrema adequação do uso de metáforas mais abrangentes sobre o sistema imune no processo de detecção de intrusos. Essas metáforas permitiram obter uma alta taxa de detecção, mesmo em um ambiente relativamente "hostil", com dados extremamente variáveis, mas com poucas diferenças entre um pacote normal e um de ataque. Dessa maneira, os dois algoritmos mostraram bom desempenho em um ambiente bastante próximo a situações reais, indicando a possibilidade de implementação dos mesmos como parte de sistemas comerciais de detecção de intruso. Além disso, esses resultados apontam para a viabilidade concreta do uso de metáforas inspiradas em uma visão mais completa do sistema imune, quando aplicadas ao processo de detecção de intrusos, ou mesmo em problemas de detecção de falhas ou detecção de anomalias.

Outra contribuição do trabalho é a implementação inicial do AISF, um *framework* criado para permitir a implementação de modelos alternativos, mas usando uma abordagem similar, de uma forma relativamente prática. Além dos algoritmos e do *framework*, uma contribuição adicional foi a criação de conjuntos de dados envolvendo situações atuais de ataque. As bases coletadas durante a realização deste trabalho serão disponibilizadas ao público acadêmico, para possibilitar estudos comparativos mais atualizados na área de detecção de intrusos. A principal base utilizada pela grande maioria dos trabalhos ainda é o KDD99, que já encontra-se bastante defasado e não oferece grandes desafios aos algoritmos aplicados na área. Cabe destacar que a coleta dessas bases foi motivada pelo desconhecimento da existência de bases públicas atualizadas e que pudessem ser utilizadas.

1.5 Estrutura do Texto

O trabalho desenvolvido é claramente multidisciplinar, o que trouxe desafios para sua apresentação, pois criou a necessidade de expor os conceitos essenciais das áreas envolvidas de uma forma concisa e direta. Houve portanto, uma preocupação em abordar esses conceitos focando três tipos de pesquisadores: i) aqueles advindos da área biológica, interessados na aplicação da Imunologia em problemas práticos de Engenharia e Computação; ii) aqueles interessados em Segurança Computacional, em busca de discussão e soluções sobre o assunto; iii) aqueles interessados especificadamente na área de Sistemas Imunes Artificiais, em busca de novos algoritmos e formas de utilização dessas ferramentas. Assim, este texto foi produzido com o desafio de buscar uma apresentação do conteúdo de forma a atender a esses profissionais.

Com a motivação de um texto focado em três tipos diferentes de leitores, optou-se por estruturar o documento da forma como se segue. O Capítulo 2 tem o propósito de apresentar conceitos básicos de Redes de Computadores, com especial destaque para o TCP/IP e seu histórico. Os detalhes e o histórico são apresentados para que o leitor possa entender melhor as causas da existência de vulnerabilidades em redes de computadores, assunto discutido no Capítulo 3. Nesse capítulo são apresentados também uma breve conceituação sobre as falhas e ataques existentes na área de Segurança Computacional. O objetivo principal desses dois capítulos, portanto, é permitir ao leitor uma

melhor compreensão do problema de detecção de intrusos em redes de computadores.

Na seqüência, o Capítulo 4 apresenta uma rápida revisão bibliográfica em Imunologia e Imunoinformática, destacando-se alguns conceitos e discussões que têm impacto na proposição de algoritmos imunoinspirados. Os algoritmos imunoinspirados, por sua vez, são abordados no Capítulo 5, que também apresenta o uso de Sistemas Imunes Artificiais na área de Segurança Computacional, destacando-se principalmente detecção de intruso em redes de computadores. Estes dois capítulos foram produzidos com a intenção de melhor contextualizar o leitor quanto aos métodos propostos e adotados neste trabalho.

No Capítulo 6, são abordados a proposta de trabalho e a metodologia utilizada, bem como uma análise sobre as decisões tomadas para o desenvolvimento da pesquisa. Essa discussão é completada no Capítulo 7, que apresenta e analisa os resultados obtidos neste trabalho de doutoramento. Por fim, o Capítulo 8 destaca as principais contribuições do trabalho como um todo, apontando também para diversos trabalhos futuros que puderam ser vislumbrados durante a realização desta pesquisa em si.

CAPÍTULO
2

Redes de Comunicação de Dados

Os conceitos vão pois ao infinito e, sendo criados, não são jamais criados do nada.

Deleuze & Guatari, O que é a Filosofia?

There was a time that the pieces fit, but I watched them fall away.
Mildewed and smoldering, strangled by our coveting
I've done the math enough to know the dangers of our second guessing
Doomed to crumble unless we grow, and strengthen our communication.

Tool, CD Lateralus, Schism

2.1 Comentários Iniciais

ENTRE as diversas tentativas de encontrar um elemento ou processo que diferenciasse os seres humanos dos demais animais, algumas abordagens, especialmente no início do século passado, focaram-se no aspecto comunicativo. Hoje sabe-se que diversas espécies utilizam-se de mecanismos comunicativos relativamente complexos, mas claramente a comunicação é um elemento extremamente importante para o *homo sapiens*.

Historicamente, a humanidade tem experienciado diversos mecanismos comunicativos, indo das imagens nas cavernas à transmissão de dados di-

gitais por ondas eletromagnéticas. Nesse contexto, atualmente destacam-se as redes de comunicação digitais, especialmente entre dispositivos computacionais. A sociedade contemporânea é caracterizada, inclusive, pela contínua e presente evolução das *Tecnologias da Informação e da Comunicação (TICs)*, as quais produzem impactos significativos em todas as esferas sociais, modificando inclusive os mecanismos de organização social.

Um personagem característico das TICs é a Internet, que agrega em si elementos tanto informáticos quanto comunicativos. A *Internet* pode ser encarada, em uma visão mais técnica, como o conjunto de protocolos, especialmente o TCP/IP, que permite a interligação de computadores de diferentes plataformas de *hardware* e *software*. Em uma visão mais sociológica, entretanto, a *internet* é um meio de comunicação extremamente poderoso, eficaz e eficiente para a comunicação de pessoas em diferentes partes do mundo. Por esse motivo, inclusive, não é incomum encontrar textos, alguns acadêmicos inclusive, utilizando o termo "internet", com inicial minúscula, dado o seu impacto como meio de comunicação, comparando-se ao rádio e à televisão em termos de impactos sócio-culturais.

Dado esse contexto é praticamente consenso entre pesquisadores de diversas áreas de que o computador e a Internet vem cada vez mais propiciando modificações nos contextos políticos, sociais e econômicos e mesmo culturais, como apontado em: (SCHAFF, 1992), (LÉVY, 1995) ou (CASTELLS, 2001). Por isto, a profunda evolução tecnológica contínua e presente do século XX até hoje tem permitido caracterizar o modo de vida atual como: a "sociedade da informação", ou "sociedade do conhecimento". Nessa sociedade, a base de produção é a informação e o conhecimento, enquanto que na sociedade industrial o capital é o motor do modo de produção e a principal fonte de poder.

Dessa maneira, a Internet cada vez mais torna-se um bem de consumo essencial, sendo num futuro próximo sua difusão deverá ser equiparável ao da energia elétrica, água encanada ou telefonia. Grandes quantias de valores são negociadas através da Internet já há um bom tempo, e não é de causar espanto que uma das maiores empresas atuais, tanto em termos de marca quanto em valor, é uma empresa que funciona apenas na Internet: o Google[1]. Dessa maneira, é cada vez mais comum que bens imateriais atinjam valor de mercado superior a bens materiais.

A volatilidade das redes digitais criam um novo ambiente, o *ciberespaço*, sobre o qual ainda não estão suficientemente claras as regras para seu uso.

[1]Google: `http://www.google.com/`.

É um espaço virtual complexo e heterogêneo, com profundos impactos na sociedade atual. O ciberespaço não envolve atualmente apenas as redes de computadores, mas cada vez mais adentra em outros espaços, como a televisão digital ou telefonia celular. Assim, serviços e ameaças digitais começam a atuar também nos espaços alternativos, já sendo bastante comuns os relatos de SPAM e vírus em celulares, por exemplo. Apesar disso, obviamente as redes digitais de computadores são o maior ponto de apoio do espaço virtual e o ambiente deste trabalho, sendo abordadas com uma maior profundidade técnica na próxima seção.

2.2 Redes de Computadores

As redes de comunicação de dados são cada vez mais presentes na sociedade atual. Mesmo indivíduos que não lidam diretamente com a tecnologia são impactados por elas, podendo ser citado o exemplo do sistema bancário que permite atualmente que transações bancárias possam ser realizadas fora da agência de origem com relativa facilidade.

Apesar desse profundo impacto, entretanto, as redes de computadores são um fenômeno relativamente recentes. O telégrafo, por exemplo, um dos seus pioneiros, surgiu apenas em 1837, podendo ser considerado, como o primeiro sistema eletrônico de comunicação de dados. O primeiro telefone capaz de transmitir conversação surgiu apenas em 1876. Mas somente em 1951 ocorreu a primeira ligação telefônica à distância sem a necessidade de uma telefonista e os serviços de fax foram introduzidos apenas em 1962 (FITZGERALD; DENNIS, 2005).

Outro fato que aponta para a juventude das redes de computadores é a rápida evolução da Internet, atualmente um fenômeno global e cada vez mais onipresente. Entretanto, por volta de 1980, a Internet resumia-se a um projeto de pesquisa envolvendo algumas dúzias de *sites*. Como apontado por (COMER, 2007), desse período aos dias atuais a Internet tem crescido a uma taxa exponencial, o que fica claro quando se avalia o número de computadores conectados a Internet.

Esse crescimento exponencial trás, obviamente, uma série de dificuldades para os estudiosos da "Grande Rede", seja sob o ponto de vista dos impactos da Internet na sociedade, seja do ponto de vista técnico, o funcionamento da Internet e das redes de computadores em si. Como também apontado em (COMER, 2007, p.34):

> *A ligação de computadores em rede é um assunto complexo. Existem muitas tecnologias, e cada uma possui características que a distingue das outras. Muitas organizações criaram, independentemente, padrões de ligação em rede que não são totalmente compatíveis. Muitas empresas criaram produtos comerciais e serviços de ligação em rede que usam as tecnologias de maneiras não-convencionais. Finalmente, a ligação em rede é complexa porque existem múltiplas tecnologias que podem ser usadas para interconectar duas ou mais redes. Como conseqüência, são possíveis muitas combinações de redes.*

Para exemplificar essa complexidade, vamos tomar como exemplo uma sessão de conferência em modo textual[2]. Supondo-se um caso relativamente comum, podem estar participando dessa "reunião virtual" usuários da Internet com diferentes formas de conexão:

- o usuário *X* conecta-se utilizando uma rede local da empresa ou escola em que trabalha;
- o usuário *Y* conecta-se por meio de uma conexão discada, utilizando uma placa de *fax-modem*;
- o usuário de banda larga *K* conecta-se por meio de uma conexão ADSL[3], através de um *modem* ADSL externo conectado a uma placa de rede comum no computador desse usuário;
- o usuário *W* conecta-se por meio de uma conexão *wireless* em seu *notebook*, usando para isso um provedor *wireless* local;
- o usuário *Z* conecta-se por meio de um celular inteligente, com recursos de acesso à Internet, utilizando um provedor da operadora da linha de celular.

Dessa maneira, os usuários interconectam-se usando diferentes meios e recursos. À parte desse caos técnico, entretanto, para os usuários os serviços são relativamente transparentes, não aparentando, em qualquer momento, que o tipo de acesso dos usuários pode ser tão diferenciado. Isso ocorre porque uma série de padrões, alguns de *facto* e outros de *jure*, permitem a interconexão dos diferentes dispositivos, fazendo com que falem uma mesma

[2]Isso é possível utilizando-se de diversos serviços, como o Skype (`http://www.skype.com/`), GoogleTalk (`http://www.google.com/talk/`), IRC (`http://www.irc.org/`), ou mesmo alguns serviços de *chat* via *Web*, como o do UOL (`http://batepapo.uol.com.br/`).

[3]ADSL – *Asymmetric Digital Subscriber Line* permite o envio e recebimento de informações digitais em alta velocidade utilizando-se o cabeamento telefônico comum, exigindo apenas centrais telefônicas digitais para poderem estar disponíveis do ponto de vista técnico.

linguagem. No caso das redes de computadores, a maior parte desses padrões dá-se sob a forma de protocolos. Um *protocolo* é um conjunto de regras a ser utilizado na troca de mensagens. No telégrafo, por exemplo, existem dois protocolos básicos sendo utilizados: 1) a utilização do código Morse para representar os símbolos do alfabeto; 2) o uso de um sinal longo para representar o traço "-" e de um sinal breve para representar o ponto ".".

Assim, existem diferentes tipos de protocolos envolvidos numa comunicação digital de dados, atuando em diferentes processos ou etapas. De forma similar, existem também diferentes tipos de redes de comunicação de dados. Como essas diferenças ocorrem por vários aspectos (forma de conexão, alcance, meio físico, utilizando protocolos básicos em uso, etc), existem também diferentes formas de classificação das redes de comunicação de dados. Em geral, a classificação mais utilizada é o alcance geográfico da rede:

Rede Local: uma rede local (*Local Area Network* – LAN) abrange um espaço físico bem delimitado em uma área relativamente pequena, como um andar de um prédio, um único edifício ou mesmo um grupo de edifícios. Em geral, LANs utilizam circuitos de redes compartilhados, sendo que os computadores revezam-se pelo uso do circuito. Assim, por exemplo, podem ser classificadas como LANs a rede existente em um prédio específico de uma universidade.

Rede *Backbone*: uma rede *backbone* (*Backbone Network* – BN) é o recurso geralmente utilizado para conectar diversas redes locais ou remotas. Em geral, não conectam usuários diretamente, mas as redes utilizadas por esses indivíduos. Dada essa característica, as redes *backbone* são extremamente velozes, utilizando-se geralmente de fibras óticas ou canais dedicados de satélite. Como exemplo de rede *backbone*, tem-se a rede que interliga os diferentes prédios de uma universidade. Outro exemplo de rede *backbone* é a rede RNP[4], uma rede de alta velocidade que interliga as universidades públicas e institutos públicos de pesquisa entre si e à Internet.

Rede Metropolitana: uma rede metropolitana (*Metropolitan Area Network* – MAN) permite a conexão de LANs e BNs localizadas em áreas diferentes, mas dentro de uma mesma região metropolitana. Por exemplo, uma universidade poderia utilizar-se de uma MAN para interconectar vários *campi* situados em diferentes regiões de uma mesma cidade.

[4]RNP: `http://www.rnp.br/`.

Rede de Longo Alcance: uma rede de longo alcance (*Wide Area Network* – WAN) pode abranger locais distantes, em várias cidades, países ou mesmo continentes. A rede formada pelas instituições ligadas à RNP, por exemplo, formam um tipo de WAN. Um usuário da RNP geralmente possui acesso mais veloz a um servidor dentro da RNP que a um servidor fora da RNP, mesmo que esteja na cidade em que o usuário se encontra. Outro exemplo de WAN são as, redes utilizadas pelo sistema bancário, interconectando as diversas agências entre si. Por fim, o exemplo mais conhecido de WAN é a Internet, interconectando diversos usuários ao redor do mundo.

É importante observar que, em parte por causa da expansão da Internet, atualmente é cada vez mais difícil distinguir entre esses tipos de redes. Por exemplo, dois usuários de ADSL em apartamentos vizinhos de um mesmo prédio estarão sob uma WAN, no caso de uma interconexão entre eles. Assim, como alertado em (FITZGERALD; DENNIS, 2005), qualquer classificação rígida de tecnologias de redes é certa de possuir excessões.

Também por causa da popularização da Internet, dois outros termos foram criados recentemente para distinguir entre serviços ofertados à rede local daqueles disponibilizados ao mundo externo. Assim, uma *intranet* é uma LAN que utiliza as mesmas tecnologias da Internet (*Web*, Java, HTML, etc), mas restrito à rede local. Uma universidade, por exemplo, poderia ter um servidor hospedando páginas *Web* para acesso apenas dentro da instituição. Por outro lado, uma *extranet* são os serviços de Internet ofertados principalmente ao público externo, acessados através da Internet. As páginas do *site* de uma universidade, por exemplo, fazem parte de sua *extranet*.

Sob a ótica da *intranet*, inclusive, cada vez mais instituições tem feito uso de redes privadas virtuais (VPNs – *Virtual Private Networks*). Em geral, empresas que não podem pagar por um circuito dedicado de dados, mas precisam interligar a matriz com suas filiais, por exemplo, podem-se utilizar de VPNs. Uma VPN permite a criação de uma rede local de dados através de um canal público de comunicação, como a Internet. Assim, cria-se uma LAN dentro de uma WAN. Isso mais uma vez mostra a complexidade em definir os diferentes tipos de redes de comunicação de dados.

Sob a ótica da segurança, é importante observar que existe uma visão incorreta bastante comum, inclusive no meio técnico, sobre segurança em VPNs. Por utilizar um canal criptografado de dados, imagina-se que isso irá garantir segurança no acesso aos dados e recursos compartilhados numa rede

local. Na verdade, há um aumento da insegurança, uma vez que, com a VPN, a rede local agora suporta mais computadores conectados, alguns remotamente.

A topologia é outra forma de classificação técnica que é bastante utilizada para redes de computadores. Nesse caso, essa classificação diz respeito à forma como os computadores estão ligados fisicamente entre si. Em geral a topologia é definida por uma série de fatores, como: tipo e capacidade dos equipamentos de rede utilizados, forma de conexão lógica, e até mesmo a forma como a rede é gerenciada. Entre as topologias de redes de computadores mais comuns, destacam-se:

Barramento: Nesse tipo de topologia, Figura 2.1, os computadores estão ligados em séries através de um único dispositivo físico, como um cabo ou *hub*, por exemplo. Nesse caso, qualquer computador percebe quando outro envia um sinal pelo barramento, o que possibilita a comunicação entre eles. Obviamente esses computadores devem se coordenar, via protocolos, para assegurar que apenas um computador envie sinais pela rede, a cada momento, evitando o caos no tráfego de mensagens. Esse tipo de topologia encontra-se cada vez mais em desuso, uma vez que propicia um grande volume de colisões em redes com um maior tráfego de rede. Redes Ethernet, a princípio, são redes baseadas em barramento, mas podem ser configuradas para utilizarem outras topologias.

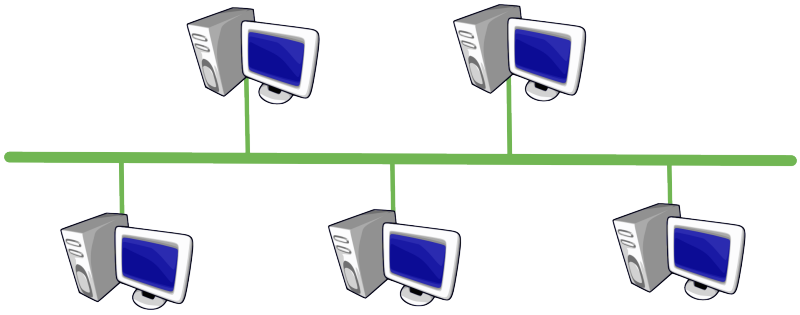


Figura 2.1: Topologia de Barramento

Anel: Neste tipo de topologia, Figura 2.2, os computadores estão ligados por um único cabo novamente, como no barramento, porém sem extremidades, formando um círculo. O sinal viaja, fazendo *loop*, pela rede e cada computador deve repetir o sinal adiante, amplificando-o, se necessário. Ao contrário da topologia de barramento, aqui os computadores não competem pelo sinal, uma vez que existe um *token*, um cartão de autorização. Um computador só pode transmitir o sinal pela rede se possuir o *token*, tendo exclusividade de uso da rede enquanto o possuir. Assim que ele

termina o uso da rede, cria um novo *token* e o libera na rede. Essa topologia é utilizada em redes Token Ring, que caíram em desuso com a larga adoção do padrão Ethernet.

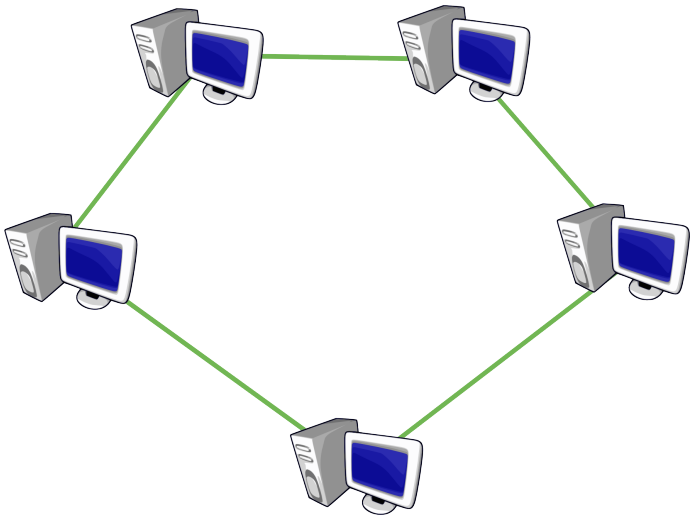


Figura 2.2: Topologia de Anel

Estrela: Neste tipo de topologia, Figura 2.3, os computadores estão ligados entre si por meio de um ponto central, em geral um dispositivo comutador de pacotes. Esse dispositivo possui várias portas, aos quais os computadores são conectados, e ele é responsável por segmentar a rede internamente, atribuindo a cada porta um segmento diferente da rede. Com isso, ele possui duas vantagens sobre a rede em barramento: evita problemas de interrupção da rede por falhas no cabo (barramento) e minimiza largamente a colisão de pacotes, melhorando o tráfego na rede. Um exemplo dessa topologia é um conjunto de computadores conectados entre si por meio de um *switch* em uma rede Ethernet. Por outro lado, uma falha no dispositivo central interrompe a rede inteira.

Malha: Nesse tipo de topologia, os computadores são ligados entre si por vários segmentos físicos, criando uma redundância de conexões da rede. Se uma das ligações falha, por exemplo, o tráfego pode fluir por outro caminho na rede. Obviamente, em geral a instalação desse tipo de topologia é dispendiosa em redes cabeadas, sendo utilizada em determinados segmentos da rede onde se deseje uma maior confiabilidade na ligação entre os pontos envolvidos. Essa é a topologia das redes *mesh*, adotada na conexão *wireless* dos *laptops* do projeto OLPC[5].

[5]OLPC: `http://wiki.laptop.org/`.

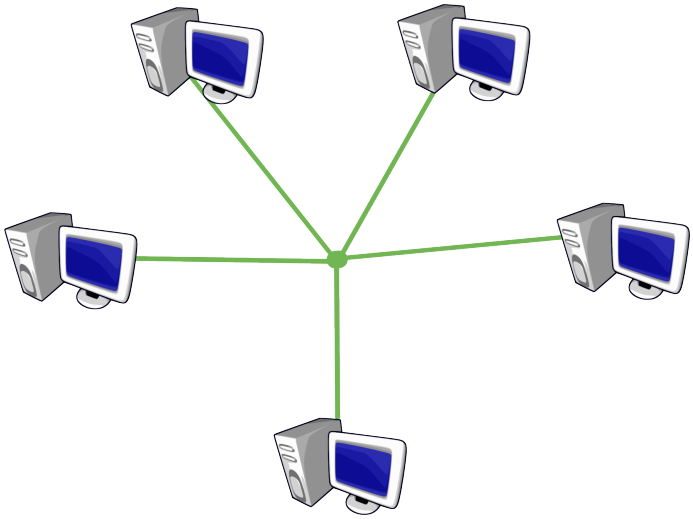


Figura 2.3: Topologia Estrela

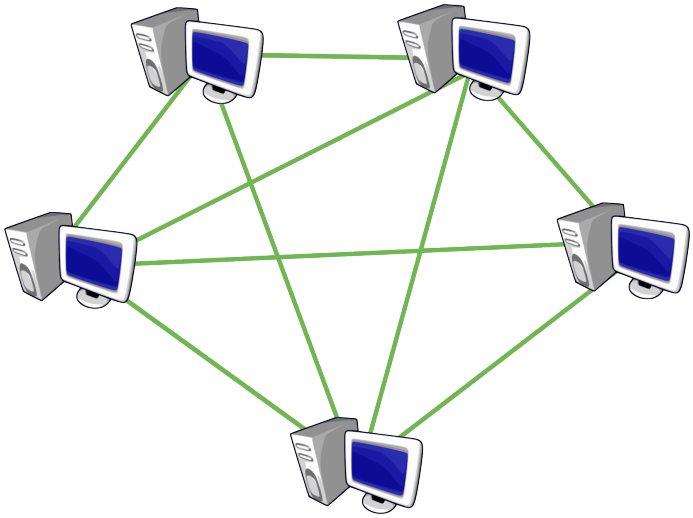


Figura 2.4: Topologia de Malha

Obviamente, em redes reais mais complexas, e especialmente na internet, predominam topologias mistas, em que redes de diferentes barramentos são conectadas. Mesmo em redes locais é comum se falar em redes com topologia estrela-barramento, por exemplo, em que várias sub-redes com topologia de barramento são conectadas entre si por meio de uma topologia estrela. E, como já comentado, em ambientes em que uma confiabilidade maior é desejada, são utilizadas conexões em malha, aumentando a possibilidade de caminhos para que um pacote possa trafegar na rede.

Dada a complexidade dos tipos de redes de comunicação de dados, existem várias maneiras de se descrever e analisar a forma como essa comunicação ocorre. A maneira mais prática e a mais adotada é fragmentar o processo de comunicação em uma série de camadas, cada qual agrupando um conjunto de

funções e recursos correlatos. Dessa maneira, desenvolvedores de *hardware* e *software* podem trabalhar em uma camada em especial, sem se preocupar com as demais. Um servidor *Web*, por exemplo, não precisa se preocupar com o tipo de conexão do usuário, se discada ou sem fio, por exemplo. Na hora de conectar um computador à rede, entretanto, isso deverá estar definido. O modelo de camadas permite, portanto, isolar as funções da rede em blocos, facilitando a compreensão e uso das mesmas.

Atualmente existem dois modelos de camadas de rede que são largamente utilizados. O primeiro modelo, um padrão de *jure*, é o Modelo de Referência para Interconexão de Sistemas Abertos, mais conhecido como modelo OSI (*Open Systems Interconection*) ou ISO/OSI, produzido pela International Organization for Standardization (ISO). Esse modelo é pouquíssimo adotado em redes reais, mas muitas de suas idéias influenciam os especialistas na área. Além disso, por fazer uma separação bastante clara entre camadas do serviço de redes, esse modelo é largamente utilizado como base em boa parte dos livros didáticos sobre o assunto. O modelo OSI é baseado em sete camadas, apresentado na tabela 2.1 e definidas à seguir.

Tabela 2.1: Modelo OSI

Camada	Principal Função	Unidade
7 – Aplicação	Interfacear a aplicação do usuário com o protocolo de comunicação	Dados
6 – Apresentação	Representação e encriptação (cifragem) de dados	Dados
5 – Sessão	Estabelecimento de sessões entre nós da rede	Dados
4 – Transporte	Conexão origem - destino confiável dos pacotes	Segmentos
3 – Rede	Determinação da rota e endereçamento lógico	Pacotes
2 – Enlace	Endereçamento físico	Quadros (*frames*)
1 – Físico	Transmissão do Sinal	*Bits*

Assim, de acordo com o modelo OSI, uma comunicação entre duas máquinas da rede ocorrerão da maneira como apresentada na Figura 2.5. Nesse caso tem-se que entre cada camada a mensagem é a mesma. Além disso, a mensagem “desce” da camada de aplicação na máquina origem até a camada

física da rede para, logo em seguida, "subir" até a camada de aplicação da máquina destino.

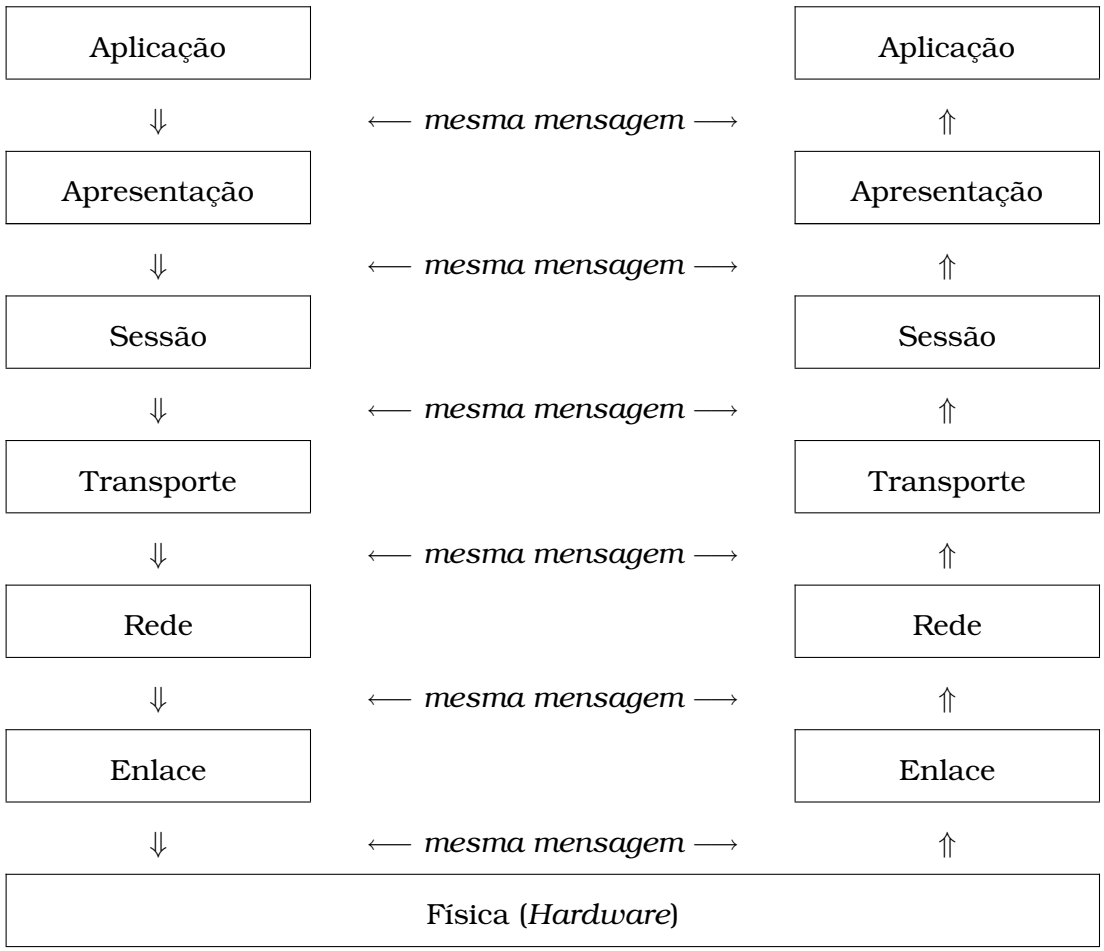


Figura 2.5: Tráfego de Mensagens no Modelo OSI

Antes de apresentar com mais detalhes as camadas do Modelo OSI, entretanto, deve-se ressaltar que este é raramente utilizado nas implementações, reais, possuindo maior valor teórico que prático. Assim, os exemplos de protocolos que serão apresentados, ao detalhar as diversas camadas, não são protocolos propostos pela OSI. Boa parte desses protocolos, inclusive, não foi projetada tomando-se o modelo OSI como referência. Feitas as devidas ressalvas, seguem-se os detalhes das camadas no modelo OSI:

Camada 1 – Física: Essa camada está relacionada ao *hardware* básico de rede, especialmente a transmissão de *bits* por um circuito de comunicação definindo as especificações físicas e elétricas dos dispositivos de rede. Ela envolve, portanto, os equipamentos de cabeamento ou outros canais de comunicação relacionados diretamente a interface de rede. Exemplos dessa camada são as especificações 10 BASE-T e 100 BASE-TX, referentes ao tipo de cabeamento de rede adotado na maioria das redes locais

atualmente. Pode ser citada também a série 802.11 do IEEE, que refere-se aos vários tipos de redes *wireless* em uso atualmente.

Camada 2 – Enlace: Essa camada também é chamada de Acesso ao Meio (*Media Access Control* – MAC), Ligação de Dados ou *Link* de Dados. Ela tem por função detectar e, opcionalmente, corrigir erros ocorridos no nível físico, sendo responsável pelo envio de uma mensagem de um computador ao próximo, no caminho da rede. Ela ainda especifica como organizar os dados em quadros, reconhecendo os limites da mensagem, bem como definir como será o formato de transmissão (se em *frames*, em *bits*, etc). Exemplos de protocolos que atuam nessa camada são: IEEE802.3 (Ethernet[6]), Token Ring[7], Frame Relay[8], PPP[9], e a série IEEE 802.11, sobre redes *wireless*.

Camada 3 – Rede: Essa camada é responsável pelo roteamento dos pacotes de rede. Ela define a melhor rota pela rede para o envio de um pacote, de sua origem ao destino. Essa camada também é responsável pelo endereçamento dos pacotes, convertendo endereços lógicos (como endereços

[6]Ethernet é uma tecnologia de interconexão para redes locais baseada em *frames* que define cabeamento e sinais elétricos para a camada física, e formato de pacotes e protocolos para a camada de controle de acesso ao meio. A Ethernet encontra-se em uso desde os anos 80, sendo que a partir dos anos 90 vem sendo a tecnologia de LAN mais amplamente utilizada e tem tomado grande parte do espaço de outros padrões de rede cabeadas.

[7]Protocolo de redes que utiliza uma ficha (*token*), um *frame* de três *bytes*, que circula numa topologia em anel em que as estações devem aguardar a sua recepção para transmitir. A transmissão dá-se durante uma pequena janela de tempo, e apenas por quem detém o *token*. Este protocolo foi descontinuado em detrimento de Ethernet e é utilizado atualmente apenas em redes mais antigas.

[8]Frame Relay é uma técnica de comutação de quadros para conexão digital, atualmente caindo em desuso com a evolução de tecnologias alternativas, como MPLS (*Multi Protocol Label Switching*), *cable modem* e as diversas variações do DSL (*Digital Subscriber Line*), como o ADSL. Entretanto, antes dessas tecnologias, ou onde essa tecnologia ainda não encontra-se disponível pelas empresas de telefonia, é uma das formas de conexão de médias e grandes empresas à Internet.

[9]PPP - *Point-to-Point Protocol* é um protocolo ponto-a-ponto com o objetivo de transportar todo o tráfego entre dois nós de rede através de uma conexão física única, como: cabo serial, linha telefônica, cabo USB, cabo de rede *crossover*, etc. A maioria dos provedores de internet utilizam o PPP para acesso discado à Internet. Ele também é utilizado nas conexões DSL, mas de forma encapsulada, geralmente sobre Ethernet, sendo chamado de PPPoE (*Point-to-Point Protocol over Ethernet*) nesse caso.

IPs) em endereços físicos das interfaces de redes. Entre os protocolos que atuam nessa camada, encontram-se: ARP[10], ICMP[11] e o IP[12].

Camada 4 – Transporte: Essa camada provê transferência transparente de dados de maneira eficiente e confiável, e com baixo custo computacional. Uma atribuição dessa camada é a de fragmentar um grande volume de dados na origem da transmissão recebidos das camadas superiores, em pacotes menores (se necessário), reagrupando novamente os pacotes menores em pacotes maiores no destino. A camada de transporte pode oferecer serviços orientados ou não à conexão. Quando operando no modo orientado à conexão, ela é capaz de garantir que todos os pacotes envolvidos em uma transmissão de dados sejam recebidos corretamente e em ordem. Os exemplos mais clássicos de protocolos nessa camada são o SCTP[13] e, principalmente o UDP e o TCP[14].

Camada 5 – Sessão: os protocolos dessa camada especificam como estabelecer uma sessão de comunicação entre dois pontos da rede. Essa camada é responsável, portanto, por iniciar, manter e encerrar sessões lógicas de comunicação, formando um circuito virtual de comunicação entre dois pontos da rede. Pode-se entender um circuito virtual como um caminho explícito na rede entre dois pontos. Um exemplo de circuito virtual é uma ligação telefônica: ao fazer a discagem, o aparelho de quem faz ligação é conectado ao telefone de destino, estabelecendo um caminho dedicado para o fluxo de voz. O SIP[15] é um exemplo de protocolo nessa camada. Serviços que usam RPC[16] também explicitam claramente essa

[10] O protocolo ARP (*Address Resolution Protocol*) traduz endereços IPs em endereços físicos (*MAC Address*) das interfaces de redes.

[11] ICMP (Internet Control Message Protocol) é um dos protocolos centrais da Internet, sendo utilizado para envio de mensagens de erro (como serviço não disponível ou que um roteador não pode ser alcançado. Em geral, as aplicações de rede não fazem uso direto desse protocolo, com excessão de ferramentas de monitoração, como ping ou traceroute.

[12] Os protocolos TCP, UDP e IP são a espinha dorsal da Internet e serão vistos com mais detalhes na Seção 2.4.1.

[13] O SCTP (*Stream Control Transmission Protocol*) é um protocolo baseado em seqüência de *bytes*, ao contrário do TCP, transportando dados em vários pacotes. Ele foi criado para atender aplicações que precisavam de um protocolo de transporte sem as limitações do TCP, entre eles envio de pacotes fora de ordem e preservação dos limites da mensagem. Atualmente é utilizado principalmente em aplicações de Voz sobre IP – VoIP.

[14] Os protocolos TCP, UDP e IP são a espinha dorsal da Internet e serão vistos com mais detalhes na Seção 2.4.1.

[15] SIP (*Session Initiation Protocol*) é um protocolo utilizado para iniciar sessões de comunicação interativa entre usuários, estabelecendo chamadas e conferências em aplicações envolvendo *streaming*, como Voz sobre IP.

[16] Uma Chamada de Procedimento Remoto (*Remote Procedure Call* – RPC) é uma tecnologia de comunicação entre processos que permite a um aplicativo chamar um procedimento em outro computador na rede. Com o uso desse recurso, o desenvolvedor não se preocupa com

sessão. O TCP, apesar de ser um protocolo da camada de transporte, faz uso dessa camada no estabelecimento e encerramento de sessões entre os aplicativos.

Camada 6 – Apresentação: essa camada especifica como os dados devem ser apresentados, estabelecendo um contexto entre as aplicações, isolando a parte sintática da parte semântica da mensagem. Entre os serviços ofertados por essa camada, encontram-se: codificação, compressão e/ou criptografia de dados. Entre os protocolos desta camada, encontram-se o ASCII[17], MPEG[18], XDR[19], MIME[20], SSL e TLS[21].

Camada 7 – Aplicação: essa camada é a mais próxima do usuário, pois é a camada onde residem os diversos serviços de redes. Cada tipo de serviço, em geral, utiliza-se de um ou mais protocolos diferenciados. Assim, por exemplo, o serviço de *e-mail* faz uso principalmente dos protocolos

---

detalhes de implementação da interação remota, pois a chamada se assemelha a chamadas de procedimentos locais. Entre os serviços que utilizam o RPC, destaca-se o NFS (*Network File System*), sistema de arquivos em rede, muito utilizado para compartilhamento de arquivos em redes UNIX.

[17]O padrão ASCII (*American Standard Code for Information Interchange*) é uma codificação de caracteres baseada no alfabeto inglês e no uso de sete *bits* para a representação .

[18]O MPEG é um conjunto de padrões de compressão usado em vídeo e áudio e que foi desenvolvido pelo grupo MPEG (Moving Picture Experts Group), donde o nome. Possui várias versões em uso atualmente, sendo as mais utilizadas: MPEG-1, MPEG-2 e MPEG-4. O formato MP3, formato de compressão de áudio bastante difundido, na verdade refere-se à camada 3 do MPEG-1, e mais recentemente à camada 3 do MPEG-2.

[19]XDR (*eXternal Data Representation*) é um padrão criado em 1995 pelo IETF (Internet Engineering Task Force), para permitir que dados pudessem ser trocados entre computadores de arquiteturas computacionais heterogêneas.

[20]O protocolo básico de correio eletrônico suporta apenas o padrão ASCII para mensagens, limitando o conteúdo das mensagens de *e-mail*, tanto na inclusão de anexos quanto no uso de caracteres de outras línguas que não a inglesa. Para resolver esse problema, surgiu o MIME (*Multipurpose Internet Mail Extensions*), que permite o envio de outros tipos de informação por *e-mail*, incluindo caracteres não utilizados no idioma inglês, usando codificações diferentes do ASCII, assim como arquivos binários contendo imagens, sons, filmes, e programa de computadores. O MIME é também um componente fundamental de protocolos como o HTTP, que requer que os dados sejam transmitidos em contextos semelhantes a mensagens de *e-mail*, mesmo que o dado a ser transmitido não seja realmente um *e-mail*.

[21] O TLS (*Transport Layer Security*) e o seu antecessor, o SSL (*Secure Sockets Layer*), são protocolos criptográficos que proporcionam comunicação segura na Internet para serviços como *e-mail* (SMTP), navegação por páginas (HTTP) e outros tipos de transferência de dados. Esses protocolos são os responsáveis pela conexão segura nos navegadores: o "`s`" em `https://` indicava, antes do surgimento do TLS, o uso do SSL, por exemplo. O SSL foi criado originalmente pela Netscape, sendo que a versão 3.0 data de 1996. Essa versão serviu de base para a criação do TLS 1.0, em 1999, pelo IETF, sendo que existem diferenças mínimas entre o SSL 3.0 e o TLS 1.0, que são essencialmente o mesmo protocolo.

SMTP[22], IMAP e POP[23]. Os serviços *Web* utilizam-se do protocolo HTTP[24]. Outros serviços clássicos dessa camada são o acesso remoto (Telnet ou SSH[25]) transferência de arquivos (FTP e Rsync[26]), compartilhamento de arquivos (SMB, CIFS e NFS[27]) ajustes de relógio (NTP[28]) e mensagem instantânea (XMPP, MSN, ICQ e IRC[29]).

Como comentado anteriormente, o modelo OSI é mais utilizado como referência teórica. A maioria das implementações atuais utilizam-se do *Modelo TCP/IP* também chamado de *Arquitetura Internet*. Esse modelo não é definido formalmente, ele tem que ser interpretado a partir de várias normas e práticas que modelam a Internet. Assim esse modelo, ao invés de ter sido desenvolvido por um comitê formal, é o resultado do esforço de milhares de pessoas que desenvolvem partes da Internet. Uma descrição do Modelo TCP/IP pode ser

[22]SMTP (*Simple Mail Transfer Protocol*) é o protocolo utilizado para envio de *e-mails* na Internet.

[23]POP (*Post Office Protocol*) e IMAP (*Internet Message Access Protocol*) são protocolos para leitura de *e-mail* por aplicativos clientes. Com uso do POP, as mensagens contidas numa caixa de correio eletrônico podem ser transferidas seqüencialmente para um computador. Já com IMAP, por padrão as mensagens ficam armazenadas no servidor e o usuário pode ter acesso a suas pastas e mensagens em qualquer computador, tanto por *webmail* como por cliente de correio eletrônico.

[24]HTTP (*Hypertext Transfer Protocol*), é um protocolo de comunicação básico da *World Wide Web*, permitindo a transferência de textos interconectados, os hipertextos, geralmente escritos em HTML (*HyperText Markup Language*) ou XML (*eXtensible Markup Language*).

[25]Telnet e SSH (*Secure SHell*) são dois protocolos para acesso remoto a servidores, geralmente para execução de comandos em modo texto. O Telnet vem sendo sucedido pelo SSH, pelo fato deste último possuir conexão criptografada entre cliente e servidor.

[26]FTP (*File Transfer Protocol*) é um protocolo básico para transferência de arquivos na Internet. O Rsync é um aplicativo para sincronização de arquivos e diretórios, utilizando protocolo nativo ou SSH.

[27]SMB (*Server Message Block*) é o protocolo básico da Microsoft para compartilhamento de recursos computacionais, especialmente arquivos e impressoras, em redes Windows. Esse protocolo também recebeu o nome de CIFS (*Common Internet File System*), como uma estratégia de *marketing*, a partir de 1996. Como já comentado anteriormente, o NFS (*Network File System*) é utilizado para compartilhamento de arquivos em redes UNIX

[28]NTP (*Network Time Protocol*) permite a sincronização de relógios de computadores com um servidor de referência.

[29]O IRC (*Internet Relay Chat*) é um protocolo de comunicação que foi muito utilizado nos anos 90, caindo em desuso com o surgimento dos sistemas de mensagens instantâneas, como o MSN (Microsoft Network). O primeiro sistema de mensagens instantâneas a se popularizar foi o ICQ (o nome vem da expressão inglesa "*I seek you*"). Além do MSN e ICQ, existem atualmente várias redes de mensagens instantâneas, como Yahoo Messenger, AIM (AOL Instant Mensager), a maioria utilizando protocolos proprietários, ou seja: essas redes são fechadas, não permitindo que um administrador instale um servidor local desses serviços. Uma alternativa que vem crescendo é o uso do XMPP (*eXtensible Messaging and Presence Protocol*), também conhecido como Jabber, um protocolo aberto, com bases XML, para sistemas de mensagens instantâneas. Atualmente, o XMPP é a base do Google Talk, sistema de mensagens instantâneas da Google. Uma das vantagens do Jabber, é que o usuário de um servidor (como o Google Talk)) consegue manter conversação com usuários de outras redes Jabber (como Jabber.org ou mesmo uma rede local).

encontrada na RFC 1122 (BRADEN, 1989), que apresenta quatro camadas de protocolos usados na arquitetura da Internet:

Camada de Aplicação: combina as funções das duas camadas superiores, apresentação e aplicação, do Modelo OSI. Essa camada inclui protocolos de serviços destinados diretamente aos usuários, como o HTTP, e protocolos de suporte, que disponibilizam funções comuns de sistema, como DNS ou TLS/SSL. Assim, nessa camada situam-se a maioria dos protocolos que o usuário faz acesso direto, como: SMTP, FTP, SSH, POP, IMAP, além dos já citados HTTP, DNS e TLS/SSL.

Camada de Transporte: proporciona serviços de comunicação entre dois pontos para as aplicações. Existem dois protocolos básicos nessa camada: 1) TCP (*Transmission Control Protocol*); que, oferece um serviço orientado à conexão; e 2) UDP (*User Datagram Protocol*), um serviço não orientado a conexão, para envio de datagramas. Alguns outros protocolos foram criados mais recentemente nessa camada, para atender necessidades específicas. É o caso do RTP (*Real-time Transport Protocol*) e do SCTP (*Stream Control Transmission Protocol* para aplicações de *streaming*, como transmissão de áudio e vídeo ou VoIP.

Camada de Internet: todos os protocolos de transporte da Internet utilizam o protocolo IP (*Internet Protocol*) para transferir dados da origem ao destino. O IP é um protocolo não orientado a conexão e sem garantia de entrega do pacote de dados. Além do IP, também faz parte dessa camada, o ICMP (*Internet Control Message Protocol*), utilizado para envio de mensagem de erros e verificação de disponibilidade de serviços.

Camada de Ligação: é a camada utilizada para interfacear a rede física em si. É também chamada de camada de acesso ao meio. Essa camada possui um grande número de protocolos, variando de acordo com o tipo de rede física em uso: Ethernet, Token Ring, IEEE 802.11, etc.

Ao conhecer os modelos OSI e TCP/IP é inevitável não compará-los. Isso é tão característico, que representações recentes do modelo TCP/IP, como em (FITZGERALD; DENNIS, 2005) costumam apresentá-lo com cinco camadas renomeando o nome de algumas delas, o que é apresentado na Figura 2.6. Os dois modelos possuem várias semelhanças, sendo que boa parte das camadas possuem a mesma função. Entre as diferenças sensíveis, encontram-se:

- representações anteriores do modelo TCP/IP não mencionavam a camada física. Isso ocorria principalmente porque para os projetistas da Internet, essa camada tinha pouco interesse prático, não influenciando os desenvolvimento dos protocolos básicos;
- o modelo TCP/IP não possui a camada de apresentação, deixando essa tarefa para a camada de aplicação;
- o modelo TCP/IP também não possui a camada de sessão, parte das tarefas dessa camada, como o estabelecimento e encerramento de sessões, ficam na camada de transporte. Entretanto, a maior parte das tarefas da camada de sessão OSI ficam na camada de aplicação no modelo TCP/IP.

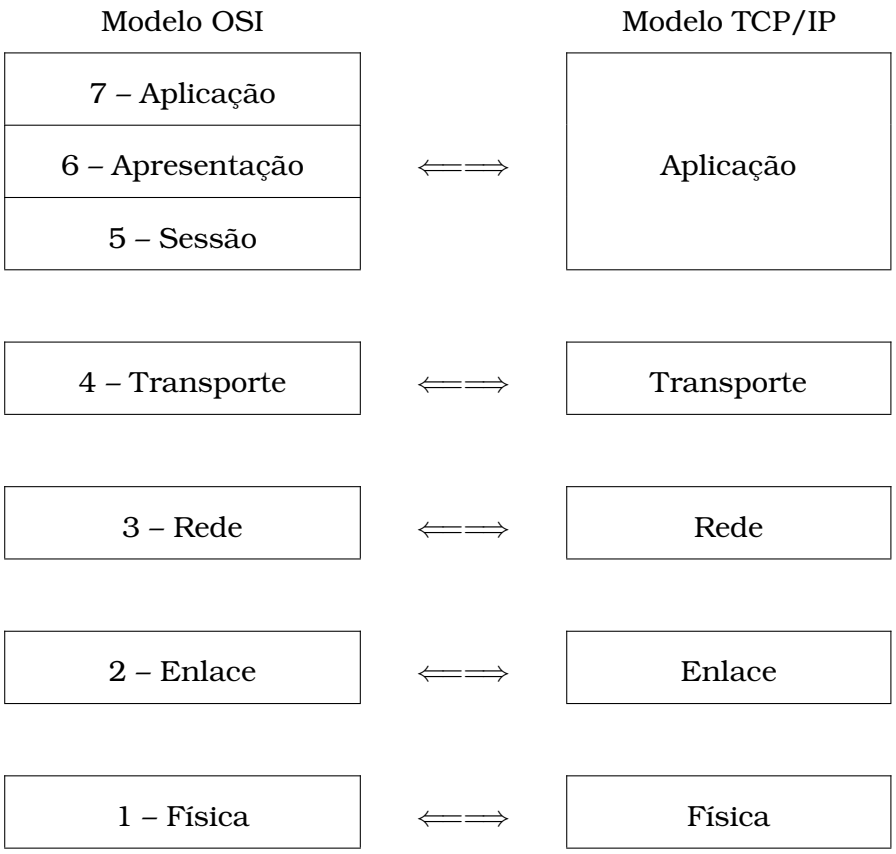


Figura 2.6: Comparação entre os Modelos OSI e TCP/IP

Em termos conceituais, a maior diferença entre os dois modelos é a inexistência formal de camadas para os serviços de sessão, apresentação e aplicação. No modelo TCP/IP se a aplicação precisa de recursos de sessão mais elaborados que o ofertado pelo TCP, então isso deverá estar implementando na aplicação: se ela precisa de criptografia dos dados e suporte a MIME, então

ela deverá ser implementada com suporte a esses recursos. E essa implementação pode ser acoplada, a aplicação fazendo ao mesmo tempo, além de seus serviços, o estabelecimento de sessões e a criptografia dos dados, por exemplo. Se isso trás um certo nível de complexidade, por outro permite um alto grau de flexibilidade na implementação de aplicações para a Internet.

Mais recentemente, tem sido cada vez mais comum a existência de autores de textos acadêmicos como FitzGerald e Dennis (2005), Comer (2007) e Matthews (2006), que utilizam o modelo TCP/IP e não o modelo OSI como o foco em seus trabalhos. Em parte, isso é devido à popularização da Internet. Mas isso também se deve ao fato que esse modelo é relativamente suficiente para uma compreensão sobre redes de computadores. Isso mostra não só o alcance prático, mas também teórico da Internet. O presente trabalho utiliza-se da Internet tornando necessário uma apresentação mais detalhada dos protocolos básicos da Internet, em especial o TCP/IP, o que será feito na Seção 2.4.1. Antes disso, entretanto, será feito um breve histórico da Internet, essencial para entender o seu funcionamento atual, bem como algumas de suas falhas e limitações.

2.3 Nascimento e Evolução da Internet

2.3.1 Origens e Motivações

Sob um ponto de vista técnico, a Internet é a maior rede de computadores existente na atualidade, sendo o resultado da interconexão de milhares de redes eletrônicas, criando um meio de comunicação global (RNP - REDE NACIONAL DE PESQUISA, 1996). Cada uma dessas redes possui estrutura e objetivos próprios, bem como utilizam-se de tecnologias diferenciadas. A interconexão entre elas é garantida pelo uso de uma linguagem comum, os protocolos de comunicação, que irão permitir a comunicação entre equipamentos com diferentes plataformas de *hardware* e *software*.

Existe uma versão corrente do nascimento da Internet que apregoa uma origem militar para a mesma, com o nome de ARPANET, em 1969. Essa variante vai além, sugerindo que a sua criação foi motivada para que redes militares pudessem operar, inclusive, em caso de ataque nuclear. Esse exemplo de teoria pode ser verificado no verbete da versão portuguesa da Wikipedia[30].

[30]Verbete: `http://pt.wikipedia.org/wiki/Internet`, visualizado em 24 de outubro de 2008.

Entretanto, como apontado por (PETER, 2007), a internet não surgiu no Pentágono, não surgiu na ARPANET, e muito menos foi criada com objetivos militares; sendo, inclusive, que o requisito de operação em caso de ataque nuclear não fez parte do *design* da Internet.

A visão da Internet surgida com fins militares advém de relações com seu contexto de criação. No início da década de 50, a Força Aérea dos Estados Unidos deu início a um projeto ambicioso, denominado SAGE (*Semi-Automatic Ground Environment*), como uma resposta à repercussão da explosão da primeira bomba de hidrogênio soviética em 1953 (CARVALHO, 2006). Esse sistema consistia de vinte e três centros de processamento de dados, com dois computadores de grande porte em cada, operando de forma distribuída, interligados entre si por linhas telefônicas.

Quando o SAGE ficou totalmente pronto, em 1961, ele já era inútil contra as novas ameaças, mísseis balísticos intercontinentais, mas trouxe uma série de inovações, entre elas o uso do MODEM, monitores de vídeo interativos e metodologias de Engenharia de *Software*. Essa experiência, obviamente foi estendida para outros projetos. Por volta dessa época, o mundo assistia a uma liderança isolada dos soviéticos na corrida espacial. Essa liderança ficou visível com o lançamento do Sputnik I, primeiro satélite artificial e o primeiro objeto fabricado pelo homem a orbitar a Terra.

Como resposta ao lançamento do Sputnik, o Departamento de Defesa dos Estados Unidos criou a ARPA (*Advanced Research Projects Agency*) em 1957, uma agência de pesquisas com o objetivo de restabelecer os EUA na vanguarda da ciência e tecnologia[31]. Essa agência tinha a missão de prevenir surpresas tecnológicas, como o Sputnik, com pesquisas de alto risco, seja casos de tecnologias em estágios iniciais ou além dos objetivos dos departamentos militares. A ARPA quase foi desfeita no ano seguinte, quando seu programa de satélite passou para a recém-criada NASA (*National Aeronautics and Space Administration*). Ela só sobreviveu porque mudou o foco de suas pesquisas básicas de longo prazo, com a participação das universidades, até então fora dos planos do Departamento de Defesa.

Como apontado em (CARVALHO, 2006), essa reinvenção tornou a ARPA uma agência de elite, respeitada cientificamente. Além disso, sua nova direção promoveu uma descentralização do gerenciamento, valorizando mérito

[31] Como apontado em uma página no *site* da DARPA (`http://www.darpa.mil/body/arpa_darpa.html`), a ARPA teve seu nome modificado para DARPA (*Defense Advanced Research Projects Agency*) em 1972, voltando a chamar ARPA em 1993. A partir de 1996, o nome voltou a ser DARPA mais uma vez, o qual persiste até hoje.

científico e técnico acima de objetivos militares. Houve um desmembramento da ARPA em pequenas unidades, para identificar novas áreas de pesquisas de longo prazo, sendo que uma dessas áreas foi a de Comando, Comunicação, Controle e Inteligência. Em 1962, foi contratado para um de seus escritórios, o IPTO (*Information Processing Techniques Office*), o psicólogo experimental Joseph Licklider, que trabalhou anteriormente no SAGE e na BBN, uma empresa formada por ex-professores e alunos do MIT.

Licklider tinha à época grande contato com pesquisas sobre computadores e havia desenvolvido toda uma visão a respeito da simbiose homem-máquina, enxergando a interação entre homens e computadores não apenas para comandar e controlar, mas também para interligar várias técnicas de computação eletrônica, que estavam separadas na época. Entre as pesquisas que passaram a ser desenvolvidas sob seu comando, encontravam-se as sobre novas interfaces homem-computador e sistemas de tempo compartilhado (*time sharing systems*), uma forma de compartilhar os recursos de um computador entre vários usuários.

Licklider, após tomar posse na ARPA, deu início a uma série de memorandos denominados "*On-Line Man Computer Communication*" (Comunicação *Online* entre Homens e Computadores) (LICKLIDER, 1962). Nesses memorandos, ele discutia a idéia de uma rede galática, um conjunto global de computadores interconectados, através dos quais qualquer um poderia facilmente acessar dados e programas de qualquer local. Como apontado em (LEINER *et al.*, 1997) e (LEINER *et al.*, 2003), os conceitos de rede galática de Licklider estavam muito próximos, em essência, à Internet atual. Apesar disso, Licklider deixou a ARPA em 1964, sem iniciar o projeto dessa rede.

Seu substituto, Robert Taylor, começou a arquitetar um projeto em 1966 para interligar os diferentes computadores das instituições de pesquisas financiadas pela ARPA. O objetivo principal dessa interligação era otimizar o uso desses caros recursos computacionais e desenvolver conhecimento de técnicas de comunicação de dados através de redes de computadores. Como apontado em (CARVALHO, 2006), esse foi o ponto de partida para a ARPANET, a rede de computadores da ARPA, sem que, na época, se soubesse como implantar direito uma rede com essa complexidade e com um histórico de vários projetos de redes da ARPA falhos na intercomunicação de máquinas diferentes.

É importante ressaltar, portanto, que a ARPANET em si foi criada para compartilhamento de recursos entre instituições de pesquisa, via *time sharing*. A idéia era que uma instituição pudesse utilizar poder computacional de ou-

tras instituições, quando fosse necessário efetuar cálculos mais complexos. Isso, como bem destacado em (PETER, 2007), não só não tinha objetivos militares, como não pode ser considerada a origem da Internet. Apesar disso, é óbvia a importância da ARPANET para a Internet em si, tanto por seus sucessos na interligação de computadores de diferentes arquiteturas, como por duas de suas importantes contribuições: o protocolo TCP/IP e o sucesso no uso de comutação de pacotes (*packet switching*).

A comutação de pacotes é uma técnica de comunicação em redes, onde pacotes (unidades de informação) são roteados entre *links* de dados compartilhados, havendo o compartilhamento do canal por vários usuários. Esse tipo de transporte contrasta com comutação de circuitos *circuit switching*, que estabelece uma conexão dedicada entre dois nós da rede, para uso exclusivo durante a comunicação. O exemplo mais familiar de rede utilizando comunicação por comutação de circuitos é a rede telefônica tradicional: quando é feita uma conexão entre dois usuários, é estabelecido um circuito entre os dois aparelhos telefônicos. O exemplo mais comum de rede por comutação de pacotes é, obviamente, a Internet, em que um pacote de dados é roteado, da origem ao destino, entre diversos pontos da rede.

A técnica de comutação de pacotes permite, entre outros, que o caminho de dados entre dois nós da rede seja alterado a qualquer momento, o que propicia uma maior robustez na comunicação. Apesar das vantagens aparentes desse tipo de técnica nos dias atuais, não havia qualquer consenso na década de 60 sobre o seu uso, e vários pesquisadores duvidavam das reais possibilidades de sucesso com seu uso. O que garantiu o seu uso foi a grande variedade de resultados em experiências, utilizando diversas formas de implantação. Os conceitos básicos dessa técnica foram originados de forma independente e simultânea nos EUA e na Inglaterra, sendo que o primeiro trabalho publicado sobre a teoria de comutação de pacotes data de julho de 1961, por Leonard Kleinrock (KLEINROCK, 1961).

Nos EUA, a Força Aérea havia encomendado à RAND Corporation um projeto de comunicação mínima essencial, permitindo, em caso de ataque inimigo, alguma forma de disparar uma operação de contra-ataque. Paul Baran, principal engenheiro da RAND, propôs o uso de redes de comunicação distribuídas (não-centralizadas) (BARAN, 1964), com flexibilidade de comutação de pequenos blocos de mensagens[32]. De certa maneira, talvez seja esse fato a motivação para uma visão "militarista" do surgimento da Internet, ainda mais

[32]Esses blocos no futuro receberam o nome de pacotes.

que a ARPANET foi construída com consultoria do próprio BARAN, sendo apresentada como um exemplo de rede recomendada pela RAND.

Como comentado em (CARVALHO, 2006), os pontos da rede operariam o *Hot Potato Routing* (roteamento da batata quente), ou seja: assim que um nó recebesse um pacote, deveria determinar a melhor rota ao destino e encaminhar o pacote ao próximo nó do caminho. A RAND, por fim, enviou um relatório à Força Aérea que solicitasse à empresa de telefonia na época (AT&T) a construção dessa rede, uma vez que ela apenas fazia pesquisas e não projetos. A AT&T respondeu na época que essa rede não funcionaria, e que não iria construí-la para incompatibilizar e competir consigo mesma. Assim, como também destacado em (CARVALHO, 2006, p.13), *"A maior ameaça às novas redes de comunicações não estava na União Soviética, mas no monopólio da companhia telefônica norte-americana"*.

Em 1965, a RAND emitiu um parecer formal recomendando que a Força Aérea atuasse sem a ajuda da AT&T. Estava tudo caminhando para isso, quando foi criada uma nova agência, a DCA (*Defense Communications Agency*) para gerenciar todos os sistemas de comunicação militares, constituída por uma equipe focada em tecnologias analógicas. Com isso, Baran percebeu que, se o projeto fosse colocado em prática, não iria funcionar e criar problemas com a técnica proposta. Ele preferiu acionar seus contatos para "congelar" o projeto até um momento mais propício (BARAN, 1999).

Na Inglaterra, por sua vez, havia também uma preocupação com o atraso científico e tecnológico. Em 1964, o governo assumiu o controle de um projeto que visava levar à indústria de informática inovações surgidas nas universidades. Donald Davies assumiu o comando desse projeto, o ACTP (*Advanced Computer Techniques Project*) e priorizou pesquisas em interface homem-computador e sistemas de tempo compartilhado (CARVALHO, 2006).

As pesquisas esbarraram também no sistema de comunicação vigente, projetado para uso de comutação de circuitos analógicos para tráfego de voz. Davies, assim como Baran, possuia formação em Computação, e propôs a divisão das mensagens em pedaços de tamanhos fixos, que denominou de *pacotes*, termo que prevalece até hoje. Davies propôs a implantação de uma nova rede, mas apesar de alinhado ao discurso de modernização tecnológica do governo, a burocracia e o desinteresse não levaram o projeto adiante. Como apontado em (CARVALHO, 2006), foi construída apenas uma rede experimental, a Mark I, que funcionou de 1967 a 1973. Em 1973, essa rede foi substituída pela Mark II, que ficou em operação até 1986.

Apesar de ter deixado a ARPA sem implementar seu projeto de rede, Licklider influenciou vários pesquisadores sobre a importância de seus conceitos de rede, entre eles Robert Taylor e Lawrence Roberts (LEINER *et al.*, 1997). Assim, sob a liderança de Robert Taylor, o IPTO seguia firme na decisão de interligar os computadores das instituições financiadas. Taylor contratou Lawrence Roberts, que fez uma apresentação do projeto da ARPANET em um simpósio sobre Sistemas Operacionais em 1967, promovido pela ACM (*Association for Computing Machinery*).

Nesse simpósio, Roberts pôde entrar em contato com Donald Davies, ficando a par do funcionamento das redes por comutação de pacotes estudadas pelos britânicos, bem como as pesquisas na área pela RAND. Isso convenceu Roberts das vantagens do uso de redes de pacotes para a ARPANET, que contou com a consultoria do próprio Baran (CARVALHO, 2006). Obviamente, a implantação da ARPANET tinha vários desafios a enfrentar:

- críticas da poderosa DCA, por causa do uso de rede de pacotes;
- oposição de algumas instituições que não queriam compartilhar seus preciosos computadores e que viam no projeto a possibilidade de redução do orçamento futuro;
- grande variedade de computadores que tinham que ser interconectados.

A ARPA teve autonomia e força para passar pelas objeções. Como forma de conectar computadores incompatíveis entre si, a ARPA utilizou a estratégia de implantar uma arquitetura que permitisse dividir as tarefas de conectividade em camadas. Essas camadas estavam dispostas em uma hierarquia conceitual, indo do nível concreto (sinais elétricos) ao abstrato (aplicações dos usuários), em uma forma semelhante às estruturas atuais de redes de computadores. Ao adotar essa estratégia, a ARPA tornou mais gerenciável a complexidade do sistema e permitiu que o desenvolvimento da rede pudesse ocorrer de forma descentralizada. Obviamente, os projetistas da ARPANET não tinham idéia de como separar as funções da rede em camadas, muito menos como as interfaces e protocolos de comunicação iriam funcionar (CARVALHO, 2006).

Uma das preocupações críticas àquela época era sobre como fazer o roteamento de pacotes em cada um dos diferentes sistemas operacionais que eram utilizados nas instituições. A alternativa foi a de utilizar minicomputadores como nós intermediários, denominados de IMP (*Interface Message Processors*), precursores dos atuais roteadores. Cada IMP estava conectado a um

único *host*, mas os vários outros IMPs, através de linhas telefônicas. Dessa forma, a ARPANET começou com duas camadas conceituais.

Outra estratégia que propiciou o desenvolvimento da ARPANET foi o estilo de gerenciamento do projeto, muito próximo do estilo universitário, com vários elementos descentralizados e informais. Por exemplo, o grupo que desenvolveu o NCP *Network Control Protocol*, protocolo de comunicações da ARPANET, era formado por alunos de graduação. Os trabalhos desses alunos foram inicialmente organizados por um deles, Stephen Crocker, em documentos denominados RFCs (*Request For Comments* (solicitação de comentários). As RFCs inicialmente eram impressas e distribuídas aos membros do grupo via correio tradicional. Elas criaram uma cultura de colaboração e interação para o desenvolvimento de idéias relacionadas à rede que são, até hoje, os documentos de registro principal para os padrões da Internet, mantendo o mesmo espírito de quarenta anos atrás. De certa maneira, elas prenunciaram, numa escala restrita e especializada, o sucesso de colaboração em rede, caso do *software* livre e, mais recentemente, da Wikipedia.

Em setembro de 1969, foi instalado o primeiro IMP da ARPANET, na UCLA (*University of California at Los Angeles*), onde trabalhava Kleinrock. Os trabalhos iniciais dele em comutação de pacotes propiciaram a escolha dessa instituição para ser o primeiro nó da rede, bem como o responsável pelo seu gerenciamento. Antes do final desse mês, quatro nós estavam interconectados, utilizando linhas telefônicas da AT&T: a UCLA, a UCSB (*University of California at Santa Barbara*), a UU (*University of Utah*) e o SRI (*Stanford Research Institute*).

A empresa escolhida para implementar os IMPs foi a BBN, com projeto tendo por gerência Robert Kahn. Em 1972, um funcionário da BBN, Ray Tomlinson, escreveu um programa para enviar e receber mensagens eletrônicas (*eletronic mail*, ou *e-mail*), com o objetivo de coordenar as atividades dos técnicos e cientistas da ARPANET. O *e-mail* tornou a partir daí a aplicação mais utilizada na Internet, contrariando totalmente a visão de que a ARPANET seria utilizada principalmente para o compartilhamento de recursos computacionais, via *time sharing*. A aplicação inicial de Ray Tomlinson foi extendida pouco depois para que pudesse listar, selecionar, arquivar e responder mensagens, o que mostra o espírito de compartilhamento e colaboração na computação em seus dias iniciais.

Em outubro de 1972, o IPTO organizou uma grande e bem sucedida demonstração da ARPANET, o que serviu para convencer até os mais céticos

sobre as vantagens das redes por comutação de pacotes. A partir daí houve todo um interesse crescente na ARPANET, com a visão de uma rede formada por múltiplas redes independentes. Dessa maneira, a ARPANET seria uma rede ligando outras redes (uma "*inter-net*"), em que cada rede poderia ser modelada para atender às necessidades específicas do ambiente e seus usuários. Como apontado em (LEINER *et al.*, 1997), a idéia de uma arquitetura aberta, introduzida por Kahn, foi guiada por quatro regras críticas:

1. Cada rede deveria funcionar por si só e não deveria precisar de alterações internas para ser conectada à ARPANET.
2. Comunicação deveria ser feita na base do melhor esforço: se um pacote não chegasse ao destino, ele deveria ser retransmitido rapidamente da origem.
3. Caixas pretas (depois chamados de roteadores ou *gateways*) deveriam ser utilizadas para conectar as redes.
4. Não deveria haver controle global no nível de operação da rede.

A expansão da ARPANET trouxe novos problemas, entre eles a dificuldade de uso do NCP em outros tipos de meio (redes por rádio ou satélite). Foram criadas redes nesse formato, mas todas incompatíveis entre si. Para interconectar essas redes heterogêneas foi criado o Projeto Internet, de onde surgiu o nome atual da rede. Esse projeto contou com a participação de Vinton Cerf, que trabalhou no desenvolvimento e projeto do NCP, e Kahn. Em 1974, é publicado um trabalho dos dois com a proposta de um protocolo para a intercomunicação de redes de pacotes, o TCP (CERF; KAHN, 1974). Interessante observar que inicialmente o TCP foi enxergado como um programa para controlar a transmissão e aceitação de mensagens, sendo inicialmente denominado *Transfer Control Program*.

Como apontado em (LEINER *et al.*, 1997), o TCP mostrou-se adequado para circuitos virtuais, incluindo aplicações de transferência de arquivos e *logins* remotos, mas não para aplicações avançadas de rede, como transmissão de voz em pacotes. O TCP foi reorganizado em dois protocolos: 1) o IP (*Internet Protocol*), que providenciava endereçamento e encaminhamento dos pacotes; 2) o TCP (*Transfer Control Protocol*), responsável pelo controle de fluxo e recuperação de pacotes perdidos. Para aplicações que não precisavam do TCP, foi criada uma alternativa, o UDP (*User Datagram Protocol*), que possibilitava acesso direto ao IP.

O TCP/IP foi implementado inicialmente pela BBN, em sistemas de grande porte. Quando surgiram os *desktops*, pensou-se inicialmente que o TCP/IP seria muito grande e complexo para poder ser executado em computadores pessoais. Entretanto, David Clark e sua equipe no MIT produziram uma implementação para as estações Xerox Alto e para o IBM PC, provando que estações de trabalho também poderiam fazer parte da Internet. Por fim, a ARPA encomendou à UCB (*University of California at Berkeley*) a implementação do TCP/IP no BSD Unix (LEINER *et al.*, 1997). Essa implementação foi um elemento chave para a adoção em larga escala da Internet.

O sucesso da ARPANET fomentou o surgimento de várias outras redes, em geral cada uma com sua própria filosofia, protocolos e aplicações. Isso exigia um grande esforço para a implementação dos (*gateways*), aplicações ou dispositivos para interligar redes distintas. Em 1983, entretanto, a transição da ARPANET para o TCP/IP foi finalizada, tendo corrido tudo bem, apesar dos receios iniciais. Além disso, propositalmente a ARPANET deixou de rodar aplicativos suportando outros tipos de rede, inclusive NCP, o que forçou a migração de todas as redes para adoção do TCP/IP.

Dessa maneira, por volta de 1985, a Internet estava estabelecida como uma tecnologia suportando uma grande comunidade de desenvolvedores e pesquisadores e começando a ser utilizada por outras comunidades. A ARPANET havia tornado o *backbone* de interligação entre as várias redes acadêmicas. O crescimento da Internet fomentou mudanças tecnológicas e administrativas, entre elas a divisão dos endereços IPs em classes e do CIDR (*Classless Interdomain Routing*). Para dar conta desse roteamento, cada vez mais mutável, as tabelas de roteamento deram origem a protocolos dinâmicos de roteamento, como o IGP (*Interior Gateway Protocol*) e EGP (*Exterior Gateway Protocol*). Na década de 80 também foi criado o DNS, por Paul Mockapetris, para substituir a tabela fixa de nomes dos *hosts*, permitindo uma arquitetura escalável e hierárquica para a denominação de pontos da rede.

Em 1984, outro fato importante foi a separação da ARPANET da rede militar dos EUA, a MILNET, evento que teve suas motivações iniciais na década anterior, quando a ARPA foi chamada a se focar em projetos militares. Em 1985, a NSF (*National Science Foundation*) iniciou o programa de criação da NSFNET, com o objetivo de oferecer infraestrutura de rede à toda comunidade acadêmica. A NSF patrocinou a criação de um novo *backbone*, passando de 6 nós com velocidade de 56Kbs para 21 nós de 45Mbs. Houve, portanto, uma explosão da rede, e a ARPANET foi desativada em 1990, após duas décadas e meia de contribuição à comunidade acadêmica.

Quando assumiu o controle da NSFNET, em 1986, Stephen Wolff reconheceu a necessidade de desenvolver uma estratégia para garantir uma infraestrutura de rede que suportasse a comunidade acadêmica e de pesquisa, ao mesmo tempo que ficasse independente do financiamento do governo. A principal abordagem adotada foi a de encorajar os escritórios regionais da NSFNET para procurar consumidores comerciais. Entretanto, ela proibia o uso de seu *backbone* para uso não acadêmico. Isso encorajou o tráfego de rede local em nível regional e fomentou a criação de redes particulares de longa distância.

Em 1995, o *backbone* da NFSNET deixou de receber fundos do governo, uma vez que os *backbones* privados eram muito melhores. Nessa época Al Gore já havia criado uma metáfora significativa para a Internet: *Information Superhighway* (Autopista da Informação), havendo toda uma "febre" de corrida para a Internet. Essa corrida foi motivada pela *Web*, uma tecnologia surgida na Europa, mais especialmente no CERN (*Centre European pour la Recherche Nucleaire*), maior centro de pesquisa à época.

Em 1989, Tim Berners-Lee, um dos pesquisadores desse centro, propôs uma arquitetura para a *Web*, bem como o protocolo HTTP. Em 1992, surge o primeiro navegador gráfico conhecido do público, o *Mosaic* para o ambiente X do UNIX, desenvolvido por Marc Andersen, futuro fundador da Netscape. O primeiro navegador gráfico havia sido criado pelo próprio Tim, mas funcionava apenas em computadores NEXT, poucos difundidos na época. A partir da *Web* a Internet difunde-se juntamente com o hipertexto e um uso cada vez maior de recursos multimídias, fazendo-se presente em todos os pontos do globo. Deve ser observado, entretanto, que o conceito de hipertexto é anterior ao nascimento da Internet, sendo inspirado nas idéias originais de Vannevar Bush (BUSH, 1945).

Conforme (BERNERS-LEE, 1996), o princípio básico da arquitetura *Web* consiste em poucas restrições para o seu formato de funcionamento na rede. Assim, no início da criação da *Web*, a proposta foi a de criar poucos protocolos para deixar a rede o mais flexível possível e manter compatibilidade entre os padrões. Assim, protocolo FTP antigo poderia ser utilizado junto com o novo protocolo HTTP no espaço de endereçamento. Além disso, os documentos de textos convencionais poderiam ser mixados com novos documentos em hipertexto.

Essa compatibilidade permitiu que a *Web* se tornasse um espaço de compartilhamento de informações, através do qual as pessoas poderiam se comunicar. A função maior da *Web*, portanto, consiste em disponibilizar in-

formação pública e privada. Nesse espaço, o hipertexto, objetiva ligar idéias e pensamentos que se relacionam com o contexto original dos documentos armazenados em sistemas de armazenamento de informações. Na *Web*, não existe um sistema fechado de banco de dados para *links*[33] – por isso é possível fazer referências hipertextuais a vários documentos permitindo o grande crescimento de informações referenciadas em formas de *links*. Essa arquitetura assim estabelecida, possui a liberdade para que pessoas façam referências sem precisar consultar os destinatários, permitindo a amplitude de escalabilidade que a *Web* tem atingido atualmente.

O impacto da *Web* foi tão significativo que em 2000 explode a famosa bolha das *pontocom*. A partir de 1995, ocorre um frenesi de investimentos especulativos na Internet e tecnologias associadas, principalmente em empresas projetadas para operar exclusivamente na *Web*, que possuíam o sufixo **.com** em seu endereço eletrônico. Muitas dessas propostas não vigoraram e, em 2000, ocorre uma fuga em massa do capital investido, principalmente na bolsa eletrônica NASDAQ. A retomada dos investimentos é inclusive fato recente, com maior moderação e estimulada pelo sucesso atual da *Web* 2.0, termo que designa uma segunda geração de comunidades e serviços baseados na plataforma *Web*, como *wikis*, aplicações interativas e redes sociais. Importante ressaltar que, apesar do termo possuir uma conotação de uma nova versão para a *Web*, ele não se refere à atualização nas suas especificações técnicas, mas a uma mudança na forma como ela é encarada por usuários e desenvolvedores.

Entre os exemplos que caracterizam essa prática, destacam-se as enciclopédias colaborativas, os *sites* de notícias, controladas pelos usuários, as aplicações *desktop* via *Web*, o compartilhamento de informação, e por fim a *blogosfera*. De acordo com Tim O'Reilly (O'REILLY, 2005), o conceito de *Web* 2.0 começou em uma seção de *"brainstorming"* em uma conferência. Nesta conferência, Dale Dougherty, pioneiro da *Web*, "*notou que longe de ter quebrado, a* Web *estava mais importante do que antes, com novas aplicações e sites excitantes aparecendo com grande regularidade*". Os participantes desta conferência perceberam que entre o responsável por uma virada na *Web* ser o colapso do *pontocom* ou a existência de uma chamada *Web 2.0*, escolheram a segunda opção. A partir deste momento teria nascido a "*Web 2.0* conference" (O'REILLY, 2005).

[33] *Links*: elo de ligação entre hipertextos, que permite "passar" com facilidade de um hipertexto para outro.

Uma ferramenta bastante utilizada na *Web 2.0* é o AJAX *Asynchronous JAvascript and XML*), conjunto de tecnologias que permitem atualizar uma parte da página sem precisar atualizá-la por inteiro. O uso de AJAX permite o desenvolvimento de páginas mais interativas para o usuário, utilizando-se de solicitações assíncronas de informações. Isso obviamente aumenta o potencial e flexibilidade da *Web* em si, a ponto de poder se sugerir o uso de aplicações tradicionais ou todo o *desktop* via navegador, fomentando o uso de Cloud Computing[34]. Em contrapartida, esse movimento trás também algumas preocupações e riscos à *Web*, especialmente sob a ótica da Segurança Computacional, uma vez que mais dados e aplicativos migram para a Internet.

2.3.2 Internet no Brasil

No Brasil, o primeiro acesso à Internet deu-se pela *Bitnet* no Laboratório Nacional de Computação do Rio de Janeiro (LNCC) e pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP). Tendo como base a Rede *Bitnet*, somente era possível a troca de *e-mail* pela Internet, sem conexão direta dos computadores à rede. Somente em 18 de julho de 1989 nasceu a Internet brasileira com a Inauguração do *Alternex* no Rio de Janeiro, e em 1991 a FAPESP consegue estabelecer as primeiras conexões a Internet e a administrar o domínio ".br".

Nessa mesma época foi implantada a RNP (Rede Nacional de Pesquisa), controlada pelo CNPq, com o objetivo de expandir a rede inter-acadêmica. Até então, esses recursos de comunicação encontravam-se mais centrados a atender objetivos de comunidades acadêmicas, porém contando-se com as ligações providenciadas por fundações de pesquisa e do LNCC. Se a RNP restringia-se quase exclusivamente a área de interesse da comunicação acadêmico-científico, algumas organizações não-governamentais, como o Ibase, passaram a fornecer as primeiras ligações de interesse privado.

Esse quadro transformou-se radicalmente a partir do final de 1994, quando órgãos como Ministério das Comunicações e Ministério da Ciência e Tecnologia lançaram uma nota conjunta que mudou a política do governo para a área, passando a estimular o surgimento de provedores *privados* de serviços Internet, de forma a atender as necessidades de todos os segmentos da sociedade. A partir desse momento, começam a entrar em funcionamento os primeiros

[34] *Cloud Computing* representa um ambiente de computação baseado em uma rede massiva de servidores, sejam virtuais ou físicos, hospedando aplicações, incluindo-se aplicativos de escritório, nessa nuvem (*cloud*) e que são acessados remotamente via *Web*.

servidores *Web* do Brasil: o LSI (Laboratório de Sistemas Integráveis) na USP, a UFRJ e a Escola do Futuro. Quando os provedores comerciais começam a propagar-se no Brasil, o número de usuários de internet vai aumentando, sendo que em 1995 já haviam mais de 20 provedores no Brasil, com 120 mil usuários (ERCILIA, 2000). Isso possibilitou que a internet migrasse dos centros de pesquisas universitárias para a população em geral.

Em 1999, a *America Online* inaugura seus serviços no país e a UOL lança no Brasil o *Brasil Online*, com a disponibilidade de contas gratuitas de *e-mail*. Ainda é possível observar que um dos ápices do crescimento da Internet no Brasil deu-se mais precisamente no ano de 2000, através do aparecimento de mais de 11 provedores gratuitos. Através desta descrição, pode-se verificar que a Internet no Brasil teve um crescimento significativo dentro da escala de conexão mundial.

Apesar do Brasil ainda ter deficiência de infra-estrutura, tanto em linhas telefônicas disponíveis à maioria da população, como a baixa taxa de pessoas que possuem computadores, no ano de 1999, o Brasil já ocupava o 6^o lugar na posição dos 15 países que mais usavam a Internet [35]. Observa-se que os países que se encontram em posições mais vantajosas ao uso da Internet que o Brasil fazem parte do Primeiro Mundo (EUA, Japão, Reino Unido, Canadá, Alemanha), os quais encontram-se muito avançados tecnologicamente.

É possível observar na mídia que o crescimento do acesso à Internet está ocorrendo rapidamente no Brasil, tendência fortalecida por projetos de expansão do acesso, e popularização da informática, como exemplo: o Projeto Computador Popular, Projeto *One Laptop per Child* (OLPC), entre outros. Conforme destacam inúmeras pesquisas divulgadas na mídia, atualmente um dos principais motivos relacionados a esse crescimento é justificado pela busca pelo acesso à Internet.

2.4 Protocolos e Serviços Básicos da Internet

2.4.1 O TCP/IP

Os sistemas na Internet comunicam-se enviando e recebendo pacotes de informação. Esses pacotes contêm parte dos dados componentes da in-

[35] Dados do instituto de pesquisa International Data Corp. de 1999 - são 27,8 milhões de linhas telefônicas no país para 166 milhões de habitantes.

formação, além de caracteres de controle e endereçamento necessários para levar os pacotes aos seus destinos e remontá-los. Tudo isto é realizado pelos dois maiores protocolos de controle de dados da Internet, o TCP/IP (*Transmission Control Protocol/Internet Protocol*), formando a arquitetura TCP/IP. O protocolo IP cuida do envio e roteamento dos dados em si, enquanto que o TCP procura estabelecer conexões de transporte.

Quando um usuário solicita, por exemplo, a abertura de um *site* em seu navegador, há uma solicitação de uma conexão TCP. Dessa maneira, do navegador do usuário até o servidor do *site* solicitado, estabelece-se uma conexão virtual via TCP. Essa conexão virtual é efetuada através de portas lógicas, ligando-se uma porta cliente a uma porta servidora. Esse conceito de ligação é denominado de *sockets*. Essa situação é ilustrada na Figura 2.7, que apresenta um exemplo de conexão *Web*, em uma rede Ethernet.

O funcionamento da camada IP é enviar (rotear) pacotes da origem ao destino final. Para tornar isto possível, toda interface sob a rede precisa de um "endereço IP". Na versão mais difundida do protocolo IP (versão 4 – IPv4), um endereço IP consiste de quatro números separados por pontos, por exemplo: "167.216.245.249", em que cada número está entre 0 e 255. Os computadores em uma mesma rede tendem a ter vizinhos com IP semelhantes. Por exemplo, pode-se supor que a máquina com IP "167.216.245.250" situa-se próxima à máquina com o endereço IP "167.216.245.249".

Um pacote IP típico possui um cabeçalho de 192 *bits*, Figura 2.8, e contém, entre outras informações: endereços de origem e destino, tamanho do pacote e tipo de protocolo usado. Nessa figura, cada linha representa uma *word*[36] de 32 *bits* (4 *bytes*). Os campos apresentados na Figura 2.8 não foram traduzidos, com o objetivo de manter a nomenclatura utilizada na RFC 791 (POSTEL, 1981b), documento que é utilizado como referência básica do IP. Esses campos são detalhados a seguir:

Version: Versão do Protocolo.

IHL: *Internet Header Length* é o comprimento do cabeçalho do IP, em termos de *words* de 32 *bits*, apontando, portanto para o início da área de dados. Esse campo é necessário porque o cabeçalho IPv4 pode conter um

[36]Em Computação, o termo "*word*" é utilizado para representar a unidade de dados utilizada por uma arquitetura de *hardware*, em geral o tamanho básico do bloco de memória utilizado. Uma *word* é simplesmente um tamanho fixo para um grupo de *bits* que são operados em conjunto pela CPU. As arquiteturas usuais de computadores mais recentes utilizam *words* de 32 ou 64 *bits*.

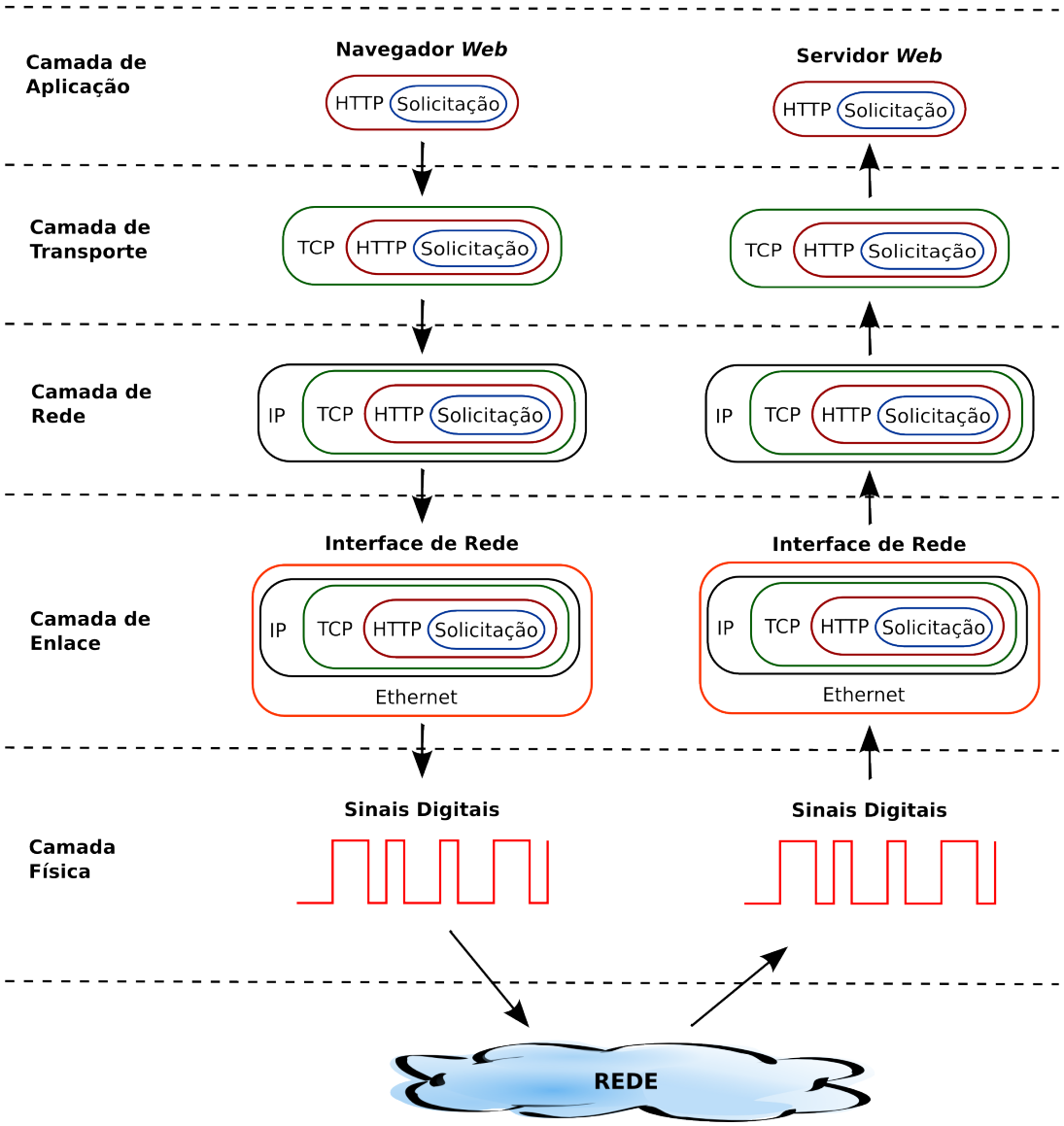


Figura 2.7: Exemplo de Conexão TCP/IP

1		2		3		4	
00	04	08	12	16	20	24	28
Version	**IHL**	**Type of Service**		**Total Length**			
Identification				**Flags**	**Fragment Offset**		
Time to Live		**Protocol**		**Header Checksum**			
Source Address							
Destination Address							
Options							**Padding**
Data							

Figura 2.8: O Pacote IP (versão 4)

número variável de opções. Um cabeçalho mínimo tem 20 *bytes* de comprimento, logo o valor mínimo em decimal no campo IHL seria 5.

Type of Service: A intenção original para o campo ToS (*Type of Service*) era especificar uma preferência para como os datagramas poderiam ser manuseados assim que circulariam pela rede, informando como o pacote deveria ser tratado. Entretanto, esse campo é ignorado pela grande maioria dos dispositivos de roteamento. Alguns trabalhos experimentais recentes, envolvendo qualidade de serviço, buscam fazer uso desses 8 *bits*.

Total Length: Comprimento total do pacote IP, em *bytes*, incluindo cabeçalho e dados. O tamanho mínimo de um pacote IP é 20 *bytes* e o máximo, 65.535 *bytes*. Obviamente, esse tamanho máximo é impraticável na maioria das redes, assim a RFC 791 especifica que os *hosts* devem suportar obrigatoriamente pacotes até 576 *bytes*. Quando chega a um roteador um pacote maior que o suportado por uma dada rede, ele pode ser fragmentado em pacotes menores.

Identification: Esse campo é usado principalmente para identificar fragmentos de um datagrama IP original, auxiliando no processo de defragmentação.

Flags: Esse campo utiliza 3 *bits* para controlar o processo de fragmentação, permitindo ou não a fragmentação e, em caso de fragmentos, identifica se o fragmento é o último de uma série.

Fragment Offset: Através desse campo, o receptor é capaz de determinar o local de um fragmento em particular no datagrama IP original.

Time to Live: O TTL (*Time to Live*) indica o tempo máximo que um pacote é permitido a permanecer na Internet, prevenindo que pacotes persistam indefinidamente. Historicamente esse campo limita a vida de um pacote em segundos, mas tornou-se uma contagem de *hops*, pontos da rede. Cada *hop* que um datagrama IP atravessa decrementa o campo TTL em um valor. Quando o campo TTL chega a zero, o pacote é descartado.

Protocol: Este campo indica o protocolo utilizado na área de dados. Entre os valores desse campo, destacam-se: 1 para ICMP, 6 para o TCP e 17 para o UDP[37].

Header Checksum: Campo de verificação (*checksum*) do cabeçalho. Esse campo é verificado e recalculado em cada ponto atravessado pelo pacote.

Source Address: Endereço IP do computador de origem do pacote.

[37]Uma relação completa das opções desse campo pode ser encontrada em `http://www.iana.org/assignments/protocol-numbers/`.

Destination Address: Endereço IP do computador a que se destina o pacote.

Options: Um pacote IP pode, opcionalmente apresentar um conjunto de opções, que devem ser suportados pelos roteadores. Normalmente, essas opções não são utilizadas, apesar da RFC 791 apresentar várias possibilidades para esse campo. Entre as opções mais utilizadas e aceitas na maioria dos roteadores, encontram-se: *Security*, que especifica o nível de segurança do datagrama; *Timestamp*, que faz com que cada roteador anexe seu endereço e horário, útil para depuração de algoritmos de roteamento; e *Record route*, que faz com que cada roteador anexe seu endereço.

Padding: Quando alguma opção é habilitada no datagrama IP, um *padding* (preenchimento) é utilizado apenas para garantir que o cabeçalho termine em uma *word* de 32 *bits*.

Data: Em seguida ao cabeçalho, viria o bloco de dados, na maior parte dos casos encapsulado em um pacote da camada de transporte (TCP ou UDP).

Como pode ser percebido pelos campos do pacote IP, ele constitui-se em um datagrama, pacote de informação num sistema de comutação de pacotes que contém endereço de origem e destino, permitindo o roteamento. Além disso, como os endereços de origem e destino são representados em 4 *bytes* na versão mais utilizada atualmente (IPv4), existe um limite de aproximadamente 4×10^9 endereços válidos, sendo que alguns trabalhos, como (ARANO, 2008), apontam para 2010 como a data em que quase todos os endereços IPv4 estariam em uso. Parte desse problema tem sido resolvida utilizando-se NAT[38] e usando endereços IPs privados[39], para máquinas que não precisem ser acessadas diretamente.

O uso de NAT permite justamente que qualquer empresa possa utilizar endereços privados ou inválidos, em suas máquinas, possuindo endereços válidos (públicos) apenas nos servidores e roteadores. O Uso do NAT é considerado uma solução temporária e, como solução mais definitiva, o IPv4 deverá ser substituído paulatinamente pelo IPv6 (DEERING; HINDEN, 1998), que suporta aproximadamente 3.4×10^{38} endereços, além de resolver alguns outros

[38]NAT (*Network Address Translation*) é uma técnica que consiste em reescrever os endereços IP de origem de um pacote de maneira que computadores de uma rede interna tenham acesso à rede pública (Internet). Essa técnica também é chamada de *masquerading*, ou IP *masquerading* e geralmente é implementada em roteadores, quando utilizada.

[39]Algumas faixas de endereços IPs na internet, como 192.168.0.1 a 192.168.255.254, são reservados para uso interno em uma rede privado e seus pacotes não poderiam ser passados para a Internet

problemas do IPv4. A adoção do IPv6, entretanto ainda é bastante restrita, pois exige substituição de equipamentos que não o suportem, motivo pelo qual este texto não irá entrar em detalhes sobre o assunto.

Enquanto o IP controla transmissões de computador a computador, na camada de rede, o protocolo TCP opera na camada de transporte, estabelecendo uma conexão lógica, um circuito virtual, entre dois sistemas, por exemplo um navegador *Web* e um servidor *Web* na Internet. O TCP provê o envio confiável e ordenado de um fluxo de mensagens de um computador a outro, controlando entre outros elementos: tamanho da mensagem e a taxa de fluxo dos pacotes na rede. Entre as funcionalidades oferecidas pelo TCP, destaca-se a confiabilidade, uma vez que ele utiliza-se de várias técnicas para proporcionar uma entrega confiável dos pacotes de dados, permitindo, entre outros: recuperação de pacotes perdidos, eliminação de pacotes duplicados e recuperação de dados corrompidos.

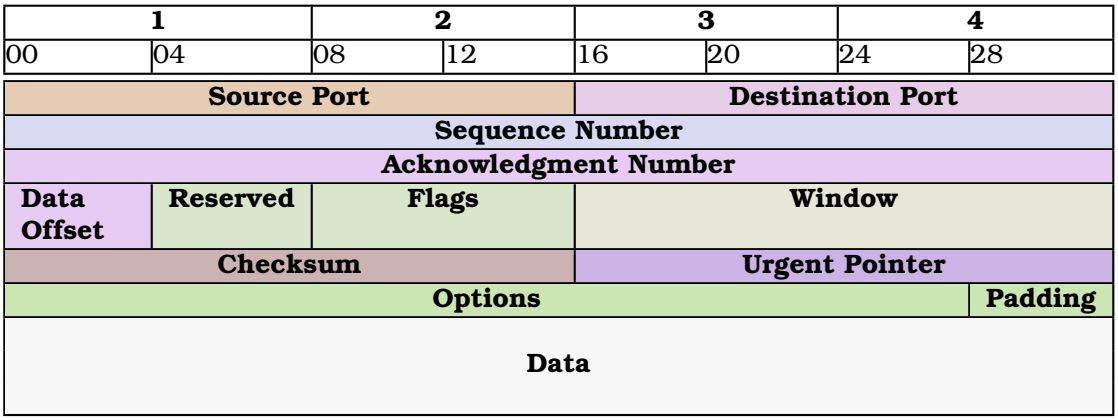


Figura 2.9: O Pacote TCP

Um pacote TCP típico possui cabeçalho com 11 *words*, Figura 2.9, das quais 10 são obrigatórias. Esse cabeçalho especifica, entre outros ítens os portos de origem e destino do pacote. Esse conceito é utilizado no modelo TCP/IP para possibilitar diferentes conexões simultâneas, cada uma associada a um serviço específico na camada de aplicação. Assim cada ponta da conexão possui um porto associado, relacionado a um aplicativo específico. Uma conexão *Web* poderia ser estabelecida, por exemplo, utilizando-se o porto 80 no servidor e o porto 16721 no computador cliente.

O leitor atento deve estar estranhando o uso do termo porto, uma vez que, em português, esse termo geralmente é chamado de “porta”, por uma tradução incorreta do inglês “port”, um falso cognato. Mesmo com a tradução incorreta, o nome “porta” foi bem assimilado por representar, metaforicamente, um significado parecido ao de porto em Informática: um ponto de entrada e saída de

objetos e mensagens. Serviços mais utilizados na Internet em geral são acessíveis em portas fixas e bem conhecidas, em geral numeradas de 1 a 1023, possibilitando a aplicativos clientes fazerem a conexão diretamente no serviço desejado[40] .

Os campos do cabeçalho do protocolo TCP, descritos na RFC 793 (POSTEL, 1981c), dizem respeito não apenas às portas utilizadas durante a conexão, mas também ao processo de estabelecimento e manutenção da conexão em si. Além disso, eles especificam como o TCP deve controlar o fluxo de mensagens. Segue-se a descrição detalhadas desses campos:

Source Port: Porta de origem no computador que envia o pacote, um campo de 16 *bits*. Isso implica que podem ser endereçadas, em cada computador, 65535 portas.

Destination Port: Porta de destino do pacote, no computador que irá receber o pacote.

Sequence Number: Número de sequência do pacote sendo transmitido, indicando a posição desse segmento no fluxo de dados. Caso a *flag* SYN não esteja habilitada no pacote, esse campo indica o número de seqüência do primeiro *byte* de dados no pacote. Em caso de presença da *flag* SYN, esse campo indica o número inicial de seqüência (ISN - *Initial Sequence Number*) e, nesse caso, o primeiro *byte* de dados possui como número de seqüência ISN+1.

Acknowledgement Number: Esse campo é utilizado para confirmar o envio de pacotes enviados anteriormente. Caso a *flag* ACK esteja ativa, ele especifica o próximo segmento aguardado. Uma vez que a conexão esteja estabelecida, esse campo é sempre utilizado.

Data Offset: Especifica o comprimento do cabeçalho do TCP, em termos de *words* de 32 *bits*. Um cabeçalho TCP pode variar de 5 *words*, 20 *bytes*, a 15 *words*, 60 *bytes*. O nome desse campo advém do fato que ele é o *offset* (balanceamento) para o início dos dados no pacote TCP.

[40]Uma relação formal dos serviços atribuídos a cada porta em si pode ser verificada em `http://www.iana.org/assignments/port-numbers`. Essa atribuição é normatizada pela IANA (Internet Assigned Numbers Authority), entidade responsável pelo gerenciamento de vários elementos da Internet, como por exemplo: alocação global de endereços IPs e estabelecimento de zonas primárias no DNS. Na Wikipedia, é possível encontrar uma relação mais completa de atribuição de serviços às portas (`http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers`), uma vez que ela lista também conflitos (dois ou mais serviços distintos utilizando uma mesma porta) e atribuições não-oficiais.

Reserved: Esse campo é reservado para uso futuro no TCP. Na RFC 793, esse campo é definido como possuindo 6 *bits*, mas dois desses *bits* já encontram-se em uso atualmente, para controle de congestão do tráfego.

Figura 2.10: *Flags* do Pacote TCP

Flags: As *flags* são *bits* de controle do pacote TCP, Figura 2.10. Na RFC 793 são definidas 6 *flags*, sendo que duas foram adicionadas mais recentemente pela RFC 3128 (RAMAKRISHNAN; FLOYD; BLACK, 2001). As *flags* em uso atualmente são as seguintes:

CWR – *Congestion Window Reduced* : Utilizada pelo *host* de envio para indicar o recebimento de pacote TCP com a *flag* ECE ativa.

ECE – *ECN Echo*: Utilizada pelo ponto da rede para informar a capacidade de uso de ECN (*Explicit Congestion Notification*).

URG – *Urgent*: Aponta que o campo *Urgent Pointer* do cabeçalho é significativo, indicando a existência de dados urgentes.

ACK – *Acknowledgement*: Aponta o campo *Acknowledgement Number* do cabeçalho como significativo, indicando que o pacote é parte de uma conexão já estabelecida.

PSH – *Push function*: Utilizada pelo remetente do segmento para indicar ao receptor que os dados do pacote em questão deve ser entregue imediatamente para a aplicação, mesmo que seja um pacote pequeno.

RST – *Reset*: Utilizada para reiniciar uma conexão que tenha ficado confusa, geralmente devido a uma falha na transmissão de pacotes.

SYN – *Synchronize*: Essa *flag*, de sincronização de números de sequência, é utilizada em conjunto com ACK para solicitar ou aceitar uma conexão: i) SYN=1 e ACK=0 indicam requisição de conexão; SYN=1 e ACK=1 indicam conexão aceita; SYN=0 e ACK=1 indicam confirmação do recebimento.

FIN – *Finalize*: Utilizada para encerrar uma conexão e indicar que o transmissor não tem mais dados para enviar.

Window: Esse campo Indica o tamanho do *buffer* do receptor, auxiliando no controle de fluxo. Ele pode ser usado pelo receptor, por exemplo, para indicar ao transmissor que diminua o fluxo de transmissão de dados.

Checksum: Campo utilizado para verificação de erros no cabeçalho e dados.

Urgent Pointer: Esse campo aponta para o *byte* onde se iniciam os dados urgentes do pacote. A interpretação desse campo só é realizada quando a *flag* URG é ativada.

Options: Campo de tamanho variável para ajustes de diversas opções, como tamanho máximo de segmento, reconhecimento seletivo, datamento dos pacotes, entre outros.

Padding: De forma semelhante ao IP, quando alguma opção é habilitada no datagrama TCP, um *padding* (preenchimento) é utilizado apenas para garantir que o cabeçalho termine em uma *word* de 32 *bits*.

Data: Em seguida ao cabeçalho, viria o bloco de dados, na maior parte dos casos encapsulado em um pacote da camada de aplicação, como HTTP, SMTP, Telnet, SSH ou FTP.

O TCP possui mecanismos para estabelecimento e finalização de conexão, o que o faz possuir certas características da camada de sessão no modelo OSI. O mecanismo utiliza-se de uma negociação (*handshake*) de três etapas no estabelecimento de uma sessão (Figura 2.11) e quatro etapas no encerramento (Figura 2.12), garantindo uma correta autenticação e encerramento de sessões completas do TCP. O uso de *handshake* no TCP ocorre não apenas durante o início e término da conexão, mas também durante a própria conexão, quando parâmetros da mesma precisarem ser negociadas.

O TCP oferece serviços de retransmissão, ordenação e reabilitação geral de pacotes, mas com um custo de pacotes extras de envio através da Rede. Assim, ao invés do TCP, alguns serviços da Internet utilizam o Protocolo de Datagrama do Usuário (UDP - "*User Datagram Protocol*"), que não oferece essas características do TCP, e que não são requisitadas por esses serviços. Em geral são serviços que precisam enviar poucos dados e nos quais é mais vantajoso reenviar o pacote em si que controlar uma conexão completa. Por isso, inclusive, o UDP é utilizado na maioria das aplicações de *streaming* de áudio e vídeo, uma vez que, nesses casos, o reenvio de um pacote não é interessante.

O protocolo UDP, descrito na RFC 768 (POSTEL, 1980), possui poucos campos, Figura 2.13, uma vez que não faz checagem ou confirmação do envio. Além das portas origem e destino (*Source Port* e *Destination Port*), inclui apenas o tamanho do pacote inteiro (cabeçalho e dados) no campo *Length*, e uma checagem superficial do conteúdo dos pacotes no campo *Checksum*.

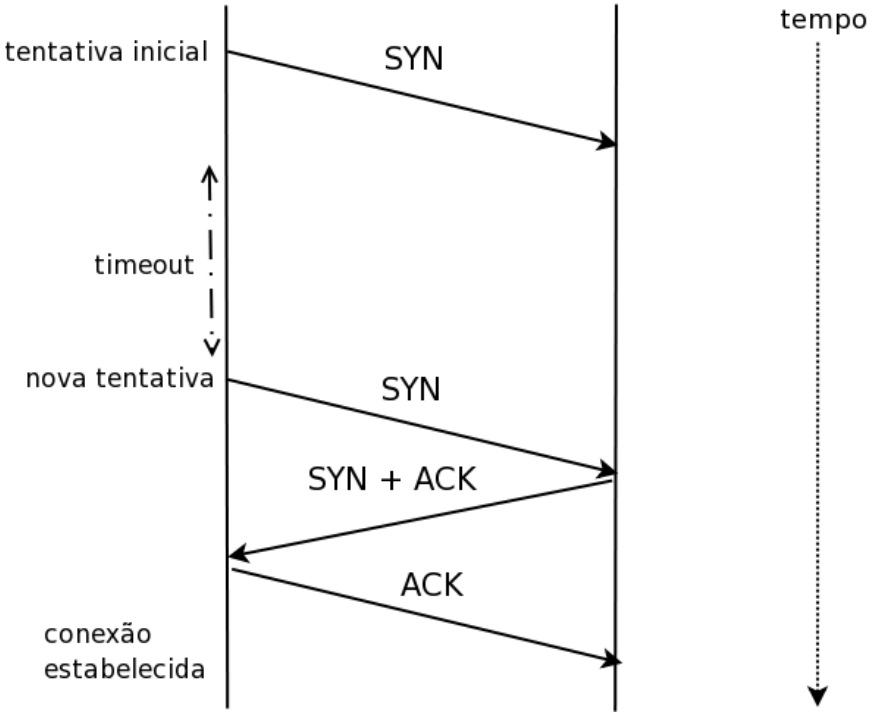


Figura 2.11: Estabelecimento de Conexão TCP

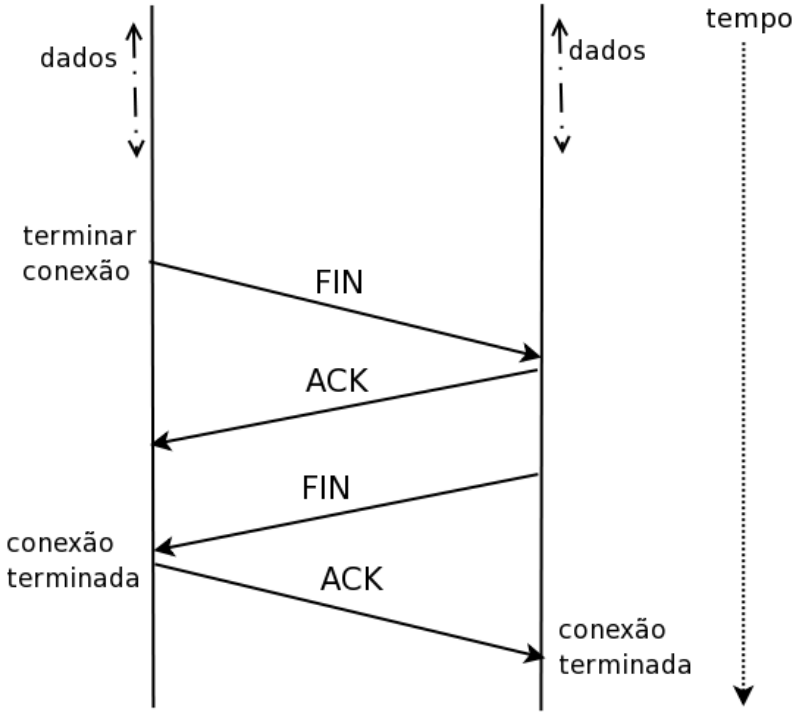


Figura 2.12: Encerramento de Conexão TCP

Dessa maneira, o UDP agrega poucas informações adicionais às já existentes no pacote IP, tendo sido criado para possibilitar o envio de datagramas, sem estabelecimento de conexão, na camada de transporte.

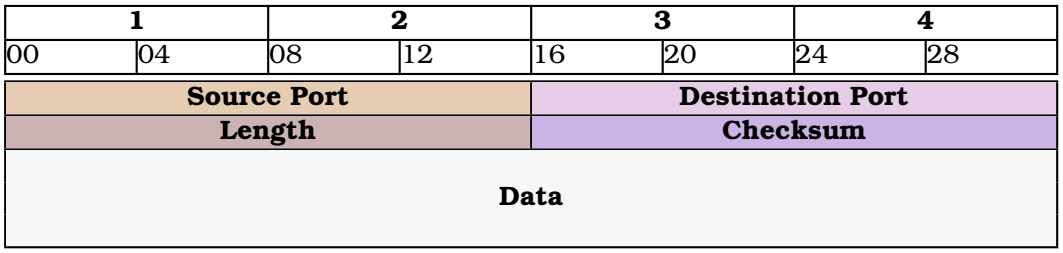


Figura 2.13: O Pacote UDP

Outro protocolo básico integrante da suíte TCP/IP é o ICMP (*Internet Control Message Protocol*), definido pela RFC 792 (POSTEL, 1981a), e utilizado para reportar erros durante o processamento de pacotes. Qualquer computador que utilize TCP/IP precisa processar mensagens ICMP e, dependendo o caso, inclusive, alterar o seu comportamento de acordo com o erro relatado. Esse protocolo não é utilizado apenas para erros, entretanto, sendo utilizado também por administradores de redes para verificar rotas e conexões. Na Figura 2.14, é apresentado o uso do comando `ping`, para o envio de 5 pacotes ICMP ao *host* `www.mit.edu`.

```
$ ping -c 5 www.mit.edu
PING www.mit.edu (18.7.22.83) 56(84) bytes of data.
64 bytes from WWW.MIT.EDU (18.7.22.83): icmp_seq=1 ttl=237 time=174 ms
64 bytes from WWW.MIT.EDU (18.7.22.83): icmp_seq=2 ttl=237 time=163 ms
64 bytes from WWW.MIT.EDU (18.7.22.83): icmp_seq=3 ttl=237 time=174 ms
64 bytes from WWW.MIT.EDU (18.7.22.83): icmp_seq=4 ttl=237 time=183 ms
64 bytes from WWW.MIT.EDU (18.7.22.83): icmp_seq=5 ttl=237 time=172 ms

--- www.mit.edu ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 163.045/173.481/183.101/6.432 ms
```

Figura 2.14: Uso de pacotes ICMP, via Comando `ping`

2.4.2 Tipos de Serviços

Os protocolos TCP ou UDP residem na camada de transporte, funcionando apenas para estabelecimento de conexões, caso do TCP, ou envio simples de datagramas, caso do UDP. Os serviços da Internet em si residem na área de dados desses protocolos, na camada de aplicação. Atualmente existe uma grande diversidade de tipos de aplicações e protocolos disponíveis, com destaques para alguns, considerados mais essenciais por usuários ou administradores, como por exemplo: serviços de *e-mail*, serviços *Web* ou o serviço de nomes (DNS).

Em princípio, existem dois modelos para o fornecimento de serviços na Internet: arquitetura cliente-servidor ou arquitetura ponto-a-ponto. No modelo cliente-servidor (C/S), os serviços são fornecidos de uma máquina ou conjunto de máquinas (a parte servidora) para várias máquinas que requisitam esses serviços (a parte cliente). Esse é o modelo mais antigo e é a base da maioria dos serviços em uso atualmente na Internet.

Mais recentemente, tem surgido alguns serviços, especialmente para compartilhamento de arquivos, que utilizam-se de uma estratégia diferenciada. Nessa arquitetura, ponto-a-ponto, ou P2P (do inglês *Peer to Peer*), não existe uma centralização de tarefas, sendo que os nós da rede atuam como clientes e servidores do serviço em questão. Em alguns casos, entretanto, o termo é utilizado inadequadamente a qualquer serviço em que usuários possam compartilhar seus arquivos. A inadequação ocorre porque alguns desses serviços são baseados em uma arquitetura cliente/servidor, pelo menos para algumas tarefas críticas, como indexação de informação e ligação dos clientes à rede de compartilhamentos. O uso do termo P2P nesses casos serve, entretanto para ressaltar o papel dos nós da rede, que passam a servir também como provedores de informação, e não apenas consumidores passivos.

É importante observar que a designação da arquitetura, P2P ou C/S, só faz sentido quando relacionada a serviços, não a computadores em si. Isso porque um computador pode ser cliente de uma aplicação, servidor em uma outra e ainda estar conectado a uma rede de compartilhamento P2P. Nesse caso, esse computador exerce diferentes papéis, em diferentes situações e momentos. A grande maioria dos servidores na Internet, por exemplo, são efetivos clientes do serviço de nomes, o DNS.

Cada interface de rede IP tem um endereço. Isso implica que para estabelecer uma conexão entre duas máquina na Internet é preciso que uma saiba o endereço da outra. Dado que os seres humanos são melhores para memorizar nomes que números, foi criado um sistema que permite associar a cada endereço IP um ou mais nomes, facilitando o estabelecimento de conexões por parte do usuário. O Sistema de Nomes de Domínio (DNS – *Domain Name System*) associa um ou mais nomes a um dado endereço IP, permitindo que, por exemplo, se use "`www.meudominio.com`" dentro de um navegador, ao invés de "167.216.245.249". Esse serviço é implementado por um servidor DNS e o usuário precisa de um endereço de IP de pelo menos um desses "servidores de nomes" em sua configuração de rede.

Além do DNS, outros protocolos são básicos na Internet atualmente. O HTTP, base dos serviços *Web*, é por exemplo base de outros serviços, como o IPP (*Internet Printing Protocol*) ou o SIP (*Session Initiation Protocol*). Além disso, tem-se uma migração de vários serviços para a *Web*. Dessa maneira, esses protocolos estão mais visíveis na Internet e, em conjunto com a suíte básica TCP/IP, são o caminho preferencial de grande parte dos ataques feitos por invasores.

2.5 Comentários Finais

O objetivo deste capítulo foi apresentar em linhas gerais o funcionamento técnico das redes de comunicação de dados, com especial destaque para o TCP/IP e a Internet. O objetivo desse foco foi o de apresentar elementos que facilitem a compreensão do trabalho, especialmente a problematização, metodologia e o contexto de trabalho.

Obviamente, este capítulo não tem o objetivo de ser um texto completo ou mesmo básico sobre a área de Redes de Computadores, uma vez que se objetivou uma abordagem mais sintética e direta do assunto. O leitor em busca de mais informações pode obter auxílio em diversos textos acadêmicos sobre o assunto, por exemplo (COMER, 2007) ou (FITZGERALD; DENNIS, 2005). Uma visão mais prática sobre o tema é abordada em (MATTHEWS, 2006). Ao leitor interessado em uma visão multimídia sobre TCP/IP, recomendamos a animação "Warriors of the Net"[41], que ilustra, de maneira bastante visual, alguns conceitos básicos da Internet.

No próximo capítulo, os elementos aqui apresentados servirão de base para compreender alguns dos vários riscos oferecidos pelos serviços de redes. O TCP/IP em si, por exemplo, possui algumas fragilidades intrínsecas, que permitem a invasores tentarem burlar, falsificar ou mesmo danificar serviços legítimos.

[41] A animação está disponível, em várias línguas, em *site* próprio: `http://www.warriorsofthe.net/`.

CAPÍTULO
3

Segurança Computacional

Creio que isto seja conseqüência de serem as crueldades mal ou bem praticadas. Bem usadas se podem chamar aquelas (se é que se pode dizer bem do mal) que são feitas, de uma só vez, pela necessidade de prover alguém à própria segurança e depois são postas à margem, transformando-se o mais possível em vantagem para os súditos.

Maquiavel, O Príncipe

O especialista em defesa oculta a si mesmo até debaixo da terra. O especialista em ataque golpeia o inimigo de cima das altas esferas do céu. Assim, ele é capaz de proteger-se a si mesmo e obter a vitória.

Sun Tzu, A Arte da Guerra

3.1 Comentários Iniciais

SE o impacto da Internet é sensível, isso ocorre tanto em seus aspectos positivos quanto negativos. Cada vez mais os crimes envolvendo o meio digital tornam-se temas de manchetes de notícias na mídia impressa ou televisiva. Nesse contexto, inclusive, é importante distinguir, ao menos em termos práticos, os conceitos de crime de computador e crime por computador. Essa distinção permite compreender melhor quais tipos de crime são cobertos pela lei e quais não o são.

Assim, *crime por computadores* são os crimes tradicionais cometidos por meios computacionais. Dessa maneira, por exemplo, tipificam-se o roubo ou o assassinato por computador. O leitor atento pode estar um pouco assustado agora, imaginando como poderia ser possível um assassinato por computador, esquecendo-se que o assassino poderia invadir o sistema computacional de um hospital, por exemplo, e alterar a medicação de um paciente para doses fortes de substâncias a que ele tenha alergia.

É comum, portanto, a ocorrência de crimes tradicionais efetuados por computador, alguns inclusive sem que o autor desses crimes esteja atento ao fato de estar cometendo um crime já previsto na lei tradicional. O envio de determinados tipos de SPAMs, por exemplo, já está previsto na lei e pode render detenção de 3 meses a 1 ano ou multa, conforme o Art. 146 do Código Penal (BRASIL, 1940). Enviar *e-mail* com ameaça de agressão pode render pena de 1 a 6 meses de detenção ou multa, de acordo com o Art. 147. Assim, apenas modificou-se o meio, o crime continua tipificado. De forma semelhante, são tipificados crimes de invasão de privacidade, envio de vírus de computador, pedofilia ou montagem de *sites* com receitas de bombas ou similares.

Por outro lado, há crimes que só ocorrem no ambiente computacional, não existindo equivalente no ambiente não tecnológico: são os *crimes de computador*. Nesse contexto, por exemplo, o Brasil não tem uma legislação contra invasão de *sites*. Quando o invasor não faz uso de informações obtidas com essa invasão (o que poderia caracterizar espionagem industrial) ou não faz alterações dos dados (o que poderia caracterizar crime de dano, vandalismo ou pichação), fica difícil caracterizar a invasão como uma contravenção penal.

Outro exemplo de crime de computador são certas práticas, que estimulam a pirataria digital, como hospedar *links* para arquivos não licenciados de filmes ou músicas hospedados em outros servidores. Isso também ainda não é tipificado pela lei brasileira e também ocorre em diversos outros países. A maioria dos *sites* usam como argumento o fato que eles não hospedam o arquivo em si, mas apenas um *link* para o mesmo.

Cabe ressaltar que alguns países já possuem uma legislação mais forte a respeito de crimes de computador, como é o caso dos EUA e da China. No Brasil, paraíso dos invasores, essa discussão está apenas no início, sendo fato recente a polêmica em torno de projetos de lei sobre o assunto. Um desses projetos exigiria o cadastro dos usuários de um provedor[1]. Outro projeto, ironizado em diversos *blogs*, exigiria a inserção da advertência, nos monitores

[1]Projeto do Senador Gerson Camata, exigindo dos provedores nome completo e número do documento de identidade do usuário, bem como identificação do terminal utilizado, data

dos computadores, de que o uso indevido do computador pode gerar infrações que sujeitam o usuário à responsabilização administrativa, penal e civil[2].

A maior polêmica atual, entretanto, recai sobre o substitutivo do Senador Eduardo Azeredo, sobre crimes contra a segurança dos sistemas informatizados[3]. Parte da polêmica recai sobre os implicativos dessa lei que, na opinião de vários especialistas[4], fere liberdades individuais e coloca várias atividades atualmente legítimas num limbo legal. Além disso, acrescenta novas responsabilidades para provedores, o que inviabilizaria várias atividades de inclusão digital, por exemplo.

À parte da polêmica sobre os projetos de lei, como informado anteriormente, ainda existem várias lacunas no que diz respeito ao direito digital. Isso cria um "limbo" legal, aproveitado por usuários mal intencionados para dispararem diversos tipos de ataques, sem se preocuparem de imediato com punições formais. Por esse motivo, cada vez mais as instituições preocupam-se com a segurança da informação, tema abordado neste capítulo.

3.2 Importância da Segurança Computacional

ATUALMENTE, é cada vez mais comum a vinculação de notícias na Internet sobre invasão ou ataques a sistemas computacionais. No momento de escrita deste texto, uma rápida pesquisa no Google, por exemplo, aponta para 312.000 páginas usando como busca "`servidor invadido`". Caso o termo seja modificado para "`página invadida`" ou "`invasão de sistemas`", é possível localizar 509.000 e 872.000 páginas, respectivamente. Mas esses termos ainda não apontam a gravidade do problema, mesmo porque parte das respostas são apenas pequenos textos ou discussões sobre o assunto. E esses resultados não são significativos quando comparados a 3.990.000 páginas para o termo "`seleção brasileira`".

O problema das invasões fica mais aparente quando se faz uma busca por termos que costumam ser utilizados por invasores quando efetuam uma invasão a um *site* e querem deixar um registro da ação. Assim, uma busca

e hora de início e término de sua utilização: `http://www.senado.gov.br/sf/atividade/materia/getHTML.asp?t=13748`.

[2]Projeto do Deputado Federal Carlos Bezerra `http://www.camara.gov.br/internet/sileg/Prop_Detalhe.asp?id=393544`.

[3]Substitutivo do Senador Eduardo Azeredo: `http://webthes.senado.gov.br/sil/Comissoes/Permanentes/CCJ/Pareceres/PLC2008061889.rtf`.

[4]Uma avaliação mais crítica desse substitutivo pode ser verificada em várias mensagens no *blog* do Sérgio Amadeu: `http://samadeu.blogspot.com/`.

por "`página hackeada`", por exemplo, retorna 10.900.000 páginas, e "`site hackeado`" retorna 73.800.000 páginas. A gravidade do problema fica mais transparente quando se verifica que boa parte das primeiras respostas dessas buscas são de: a) páginas alteradas (*defaced*) e que ainda não foram restauradas; b) repositórios de cópias de páginas alteradas ou c) alertas sobre páginas de instituições importantes que foram alteradas. A alteração da página em si trás grandes transtornos à instituição e sua imagem: o invasor pode desabilitar serviços *web* durante o processo, além de inserir mensagens ofensivas à instituição em si, como ilustrado na Figura 3.1.

Figura 3.1: *Site* Alterado após Invasão

Se os números do Google a esse respeito impressionam, é importante ressaltar que isso é apenas a parte visível do problema, uma vez que a maior parte dos ataques não são relatadas ou documentadas. Por causa do dano à imagem, as instituições ocultam ou relatam poucas informações sobre ataques efetuados, bem sucedidos ou não. Isso principalmente para evitar a fuga de clientes que se sentiriam receosos de informar ou enviar informações sensíveis para empresas que tiveram sistemas computacionais invadidos.

Danos à imagem da instituição não são os únicos riscos a que sistemas computacionais está sujeito e em geral não são o maior dano causado. São cada vez mais noticiados ataques, invasões e fraudes eletrônicas, envolvendo alvos os mais variados possíveis. Os prejuízos causados por problemas de segurança computacional são significativos, indo desde a fuga de clientes até inoperância da empresa por falhas em seus servidores. Os problemas afetam

inclusive usuários que não acessam a Internet e que são afetados indiretamente, ao terem dados furtados em sistemas bancários ou de empresas.

Além disso, boa parte das invasões é realizada não apenas como objetivo final, mas como uma ponte para um ataque maior em larga escala. Não muito tempo atrás vários provedores brasileiros, como UOL, ficaram indisponíveis após um ataque de negação de serviços distribuídos. O mesmo ocorreu com várias empresas no exterior, incluindo a SCO e até mesmo alguns *sites* da Microsoft, como uma retaliação de ativistas *hackers* por causa de processo movido pela SCO contra alguns usuários Linux. Os invasores conseguiram, através de vírus, invadir diversas estações de usuários e, a partir dessas máquinas, dispararam automaticamente ataques aos alvos verdadeiros. Mais recentemente, foram amplamente noticiados os ataques ao serviço de banda larga Speedy, da Telefônica, que causou problemas de acesso por vários para usuários do serviço no Estado de São Paulo[5].

O usuário comum é também alvo de diversos outros tipos de ataques, mais diretos, inclusive. Aplicativos para captura do que é digitado via teclado tem sido utilizados para captura de dados sensíveis, como *logins* e senhas, inclusive de acesso a instituições bancárias. Alguns desses aplicativos, em versão mais sofisticada, capturam não apenas o teclado, mas também fazem captura de telas ou de movimento de *mouses* ao usuário acessar *sites* pré-definidos, numa tentativa de burlar mecanismos de segurança adotados por alguns bancos, como a digitação da senha via *mouse* em um teclado gráfico.

Também tem sido noticiado há pouco mais de dois ano na mídia eletrônica, em diversos *blogs* e *sites*, sobre seqüestro de dados de usuários comuns. Invasores, após conseguirem acesso aos arquivos dos usuários, criptografariam esses dados e enviariam aos mesmos a mensagem para efetuarem depósito caso quisessem ter acesso a seus arquivos novamente. Aparentemente, esse tipo de ataque não tem sido lucrativo, uma vez que diminuiu o volume de notícias a esses respeito, mas o risco em potencial existe e pode ser crítico para quem possui dados valiosos. Em novas versões do ataque, o sequestro é efetuado de forma mais elaborada, através de vírus. O usuário é informado que seus dados foram infectados e é solicitado a instalar um anti-vírus específico para aquele problema. Entretanto, esse anti-vírus remove o vírus de apenas

[5]Entre as notícias sobre o assunto, pode-se citar `http://idgnow.uol.com.br/telecom/2009/04/09/pane-do-speedy-telefonica-diz-que-foi-alvo-de-acoes-externas/`. Também merece destaque `http://idgnow.uol.com.br/seguranca/2009/04/09/telefonica-foi-alvo-de-crackers-avaliam-especialistas/`.

um arquivo, exigindo pagamento para continuar o processo[6]. Esse tipo de ataque ocorre principalmente por que a grande maioria dos usuários não possui o hábito de efetuar cópias de segurança de seus arquivos[7].

Um elemento recente que agrava os problemas de Segurança Computacional é o fato que a computação torna-se cada vez mais ubíqua, presente em diversos tipos de equipamentos, celulares, *consoles* de jogos, *players* multimídia, televisores digitais, entre outros. Além disso, a popularização da Internet faz com que os problemas cheguem a diversos locais e a diversos tipos de usuários. As oportunidades trazidas pela computação e rede ubíquas são, portanto, afetadas severamente pelos riscos de segurança inerentes a esse meio.

As invasões em si são motivadas por diversos fatores, indo de ativismo cibernético, passando por espírito de desafio e jogo aos objetivos diretos de causar dano a alguém ou alguma empresa. Vários grupos de *hackers* organizam-se em grupos e costumam disparar ataques contra alvos específicos, em defesa de alguma causa, seja política, ecológica ou mesmo social[8]. Após os ataques de 11 de setembro, por exemplo, vários grupos de *hackers* americanos dispararam ataques contra *sites* islâmicos com ligações com grupos terroristas.

Boa parte das invasões são realizadas com o objetivo puro e simples da invasão em si, para divulgar o nome do invasor na comunidade *underground*. Quanto mais importante o alvo, maior o destaque obtido pelo invasor, o que de certa maneira torna-o mais respeitado por seus pares, especialmente entre aqueles que tem objetivos similares, de invadir para se auto-promover. Um motivo também bastante comum nesses grupos é o difamatório: invadir para denegrir uma instituição, um grupo ou um usuário específico.

Como exemplo de ataque com finalidades difamatórias, na Figura 3.2 é apresentado um trecho de uma revista digital (*e-zine*) divulgada principalmente por *e-mail* de um grupo de *hackers* que invadem servidores relacionados a pessoas conhecidas na área de segurança, com o objetivo explícito

[6]Um exemplo de notícia relacionada a esse tipo de ataque pode ser verificada em `http://computerworld.uol.com.br/seguranca/2009/03/27/malware-sequestra-arquivos-de-internautas-e-pede-resgate-em-dinheiro/`.

[7]Em consulta com alunos das disciplina de Segurança Computacional e Administração de Serviços de Redes de Computadores, no curso de Graduação em Ciência da Computação da UFLA, uma minoria, menos que 20% dos alunos tinha feito *backup* de seus arquivos mais importantes há menos que um mês. A grande maioria desses alunos não tinha um *backup* feito há menos de seis meses.

[8]Esse é o caso, por exemplo do cDc (*cult of Dead cow*), que defende causas ligadas à privacidade na rede. Esse grupo tornou-se famoso em 1998 pela construção do Back Orifice, um cavalo de tróia que, segundo o grupo, tinha o propósito de mostrar as falhas do Microsoft Windows 98.

de denegrir a imagem do alvo em questão[9]. Além da declaração citada na Figura 3.2, uma seção da revista dedica-se a denegrir um outro especialista, que teve servidor invadido, *e-mails* interceptados e dados pessoais divulgados na *e-zine*. Também há seções de ataques bem-sucedidos a empresas e especialistas de segurança, incluindo-se o relato de falhas de segurança afetando o grupo de resposta a incidentes da RNP.

```
PR0J3KT M4YH3M BR4Z1L DECL4R4 GU3RR4 A XXXXX XXXX

NóS d0 i sh0t the white hat / pr0j3kt m4yh3m br4z1l p0r mEio deSt4 c4Rta
d3cl4r4m0s 0f1c14lm3nt3 gUeRRa a Xxxxx Xxxx, da Xxxx e ex Xxxxxx
```

Figura 3.2: Declaração de Guerra de Grupo *Hacker* contra Especialista de Segurança

Esse espírito de desafio e jogo é bastante presente na comunidade *hacker*, mas também integram o grupo dos invasores aqueles com objetivos diretamente criminais, envolvendo principalmente estelionato, fraude e furto. Quadrilhas de falsificadores de cartões de crédito, por exemplo, contratam invasores para conseguirem dados de cartões de créditos válidos. Estelionatários criam páginas falsificadas de diversas instituições e enviam mensagens para usuários desavisados solicitando recadastramento e apontando o *site* falso para essa tarefa[10].

Cabe ressaltar que, parte da comunidade técnica não gosta quando o termo *hacker* é associado a atividades ilícitas, preferindo nesse caso que seja utilizado o termo *cracker*. Alguns ainda preferem o uso dos termos *black hacker*, para os mal intencionados, e *white hacker*, para os que usam técnicas de invasão com finalidades de verificação de segurança. Este trabalho prefere o termo português invasor, sem a discussão romântica em torno do significado da palavra *hacker*.

Se por um lado a motivação dos invasores é grande, do outro lado, não existe ainda uma movimentação forte por parte das empresas em prol da segurança de TI (Tecnologia da Informação). Isso pode ser verificado com relativa facilidade ao se acompanhar *sites* e *blogs* de notícias sobre tecnologia. No momento de escrita deste texto, o *site* brasileiro da Computerworld[11] destacava duas notícias que comprovam essa situação:

[9]Cópia dessa versão da e-zine pode ser encontrada em `http://www.milw0rm.com/papers/244`. Omitimos aqui neste trabalho o nome do alvo e as empresas ligadas a ele.

[10]Em alguns casos, esse ataque é associado a ataques a servidores DNS, desviando o tráfego legítimo para um servidor falso, como pode ser verificado em `http://info.abril.com.br/noticias/tecnologia-pessoal/virtua-e-suspeito-de-cair-em-golpe-de-dns-15042009-36.shl`.

[11]Computerworld: `http://computerworld.uol.com.br/`.

1. Pesquisa da Secure Computing, empresa de renome na área, aponta que mais de 50% dos sistemas de empresas de utilidades, como água, energia, óleo e gás, são vulneráveis a ataques. Além disso, mais da metade dessas empresas já sofreram incidentes *online* e 14% dos responsáveis por TI nessas empresas acreditam que os incidentes se repetirão em 2009[12].

2. Invasores lançaram um ataque mundial na *web*, atingindo mais de 10 mil servidores espalhados pelo mundo em dois dias. De acordo com a Kaspersky, outra empresa bastante conhecida na área de segurança, os invasores alteraram os *sites* invadidos para redirecionar os usuários para um dos seus seis servidores maliciosos. Se o computador do usuário estivesse utilizando um navegador sem atualizações de segurança, seu computador recebia vários programas maliciosos[13].

Outro dado que comprova essa situação é o resultado de uma varredura recente, feita por especialistas em segurança[14], em redes *wireless* no Rio de Janeiro. Entre os dados obtidos com essa varredura, que localizou 5.000 redes, destaca-se o fato que 30% das redes estavam totalmente abertas, sem qualquer uso de criptografia para a conexão de clientes *wireless*. Uma parte significativa, 40% das redes, utilizavam-se de criptografia WEP, que é sabidamente deficiente, possibilitando a invasão em aproximadamente 1 hora em uma rede com tráfego. Apenas 30% das redes detectadas utilizavam-se de WPA ou WPA2, que ainda são considerados protocolos seguros para controle de conexão às redes[15].

Uma observação importante é que cada vez mais invasões podem ser efetuadas sem um conhecimento técnico avançado. Inclusive, uma parte das invasões é efetuada por *script kiddies*, usuários sem conhecimento avançado de Redes de Computadores, mas que utilizam ferramentas automatizadas para disparar ataques. Em geral, utilizam ferramentas de verificação de análise

[12]Noticiado em `http://computerworld.uol.com.br/seguranca/2008/11/11/mais-de-50-dos-sistemas-de-empresas-de-utilidades-sao-vulneraveis-a-ataques/`.

[13]Noticiado em `http://computerworld.uol.com.br/seguranca/2008/11/10/mais-de-10-mil-servidores-foram-atingidos-em-ataque-em-massa-pela-web/`.

[14]Agradeço a Denilson V. Martins pela permissão para publicação dos dados na tese, antes de qualquer divulgação formal dos resultados. A varredura faz parte dos esforços de um grupo de pesquisadores, do qual faço parte, para avaliação da segurança de redes *wireless* nas capitais de RJ, MG e SP. Espera-se obter resultados muito semelhantes nas duas outras capitais.

[15]Notícia recente, publicada em `http://computerworld.uol.com.br/seguranca/2008/11/07/criptografia-wpa-para-wifi-e-quebrada-por-pesquisador-de-seguranca/`, informa que o pesquisador que quebrou o padrão WEP, Erik Tews, já encontrou uma maneira de atacar parcialmente o WPA, permanecendo apenas o WPA2 sem notícias de falhas.

de vulnerabilidades, como o Nessus[16], Figura 3.3, e ferramentas de invasão, como o Metasploit[17] , Figura 3.4.

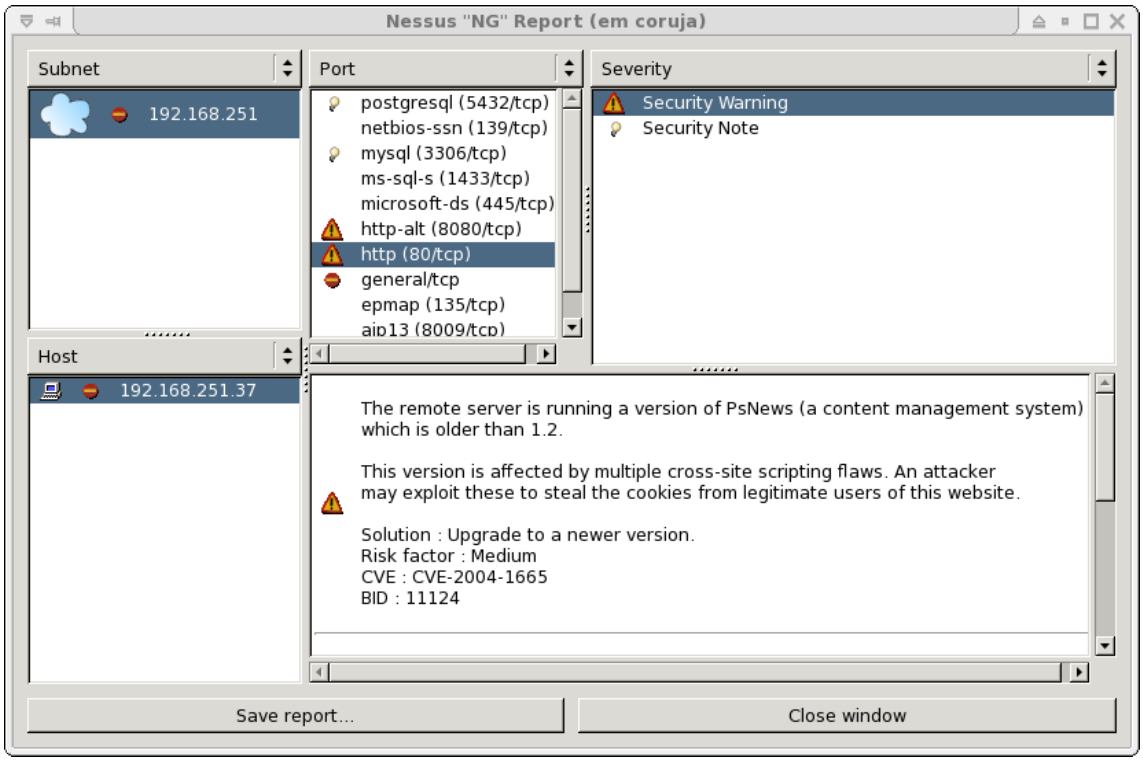


Figura 3.3: Análise de Vulnerabilidades com Nessus

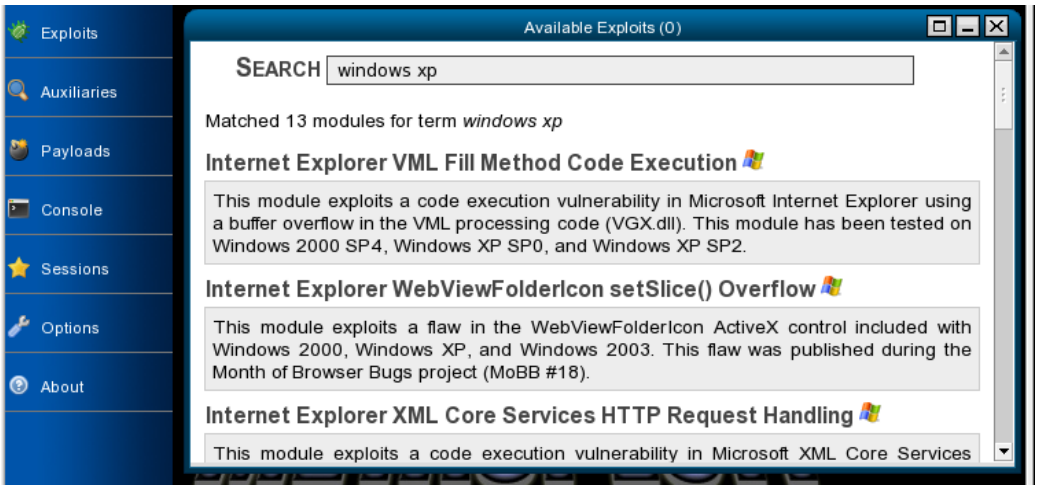


Figura 3.4: Metasploit Framework

É interessante destacar que apesar das facilidades cada vez mais crescente para invasão dos sistemas, não existe uma contrapartida equivalente por parte do pessoal de TI no que diz respeito à implementação de técnicas de Segurança Computacional. Em diferentes oportunidades, foi possível verificar esse tipo

[16]Nessus: `http://www.nessus.org/`.
[17]Metasploit Framework: `http://www.metasploit.com/`.

de problema com professores e alunos da área de TI: uso de senhas fracas em suas contas pessoais; uso de serviços não-criptografados para transferência de dados sensíveis; ausência de política adequada de *backup* pessoal.

Uma rápida varredura na rede do DCC/UFLA na época deste trabalho permitiu verificar a existência de vários computadores com aplicativos com falhas de segurança, passíveis de invasão, como exemplificado na Figura 3.3. Poderia-se pensar que isso é descuido dentro da própria área, mas pesquisa recente no Reino Unido mostrou que aproximadamente 10% dos usuários utilizam uma senha presente em uma lista de 100 palavras[18]. Não é incomum visitar empresas e encontrar senhas pregadas no monitor em etiquetas amarelas, ou anotadas em papel debaixo do teclado. Isso aponta para uma necessidade grande dos profissionais de TI se preocuparem mais com a segurança de seus usuários.

3.3 Vulnerabilidades

Um sistema computacional está sujeito a diversos riscos, que são amplificados à medida que aumentam o número de conexões locais ou remotas a esse sistema. Em termos gerais, define-se *vulnerabilidade* como sendo qualquer fraqueza ou falha em um sistema que permita a um invasor violar a integridade do sistema. O termo também pode ser aplicado para deficiências do sistema que impeçam sua rápida recuperação em caso de problemas nos equipamentos computacionais.

Vulnerabilidades podem surgir em diversos elementos num sistema computacional, destacando-se principalmente: uso de senhas fracas pelos usuários, sistemas computacionais mal projetados e *bugs* em *software*. O uso de senhas fracas facilita a descoberta das mesmas pelos invasores usando aplicativos que testam automaticamente uma grande quantidade de senhas em pouco tempo de execução. Uma vez conseguido o acesso como usuário comum, os invasores passam a explorar outras falhas do sistema em busca de acesso privilegiado.

Infelizmente, a maioria das empresas e projetistas de *software* não tem adotado em nível adequado metodologias para desenvolvimento seguro de aplicações. Várias aplicações contêm falhas críticas de segurança por causa de um projeto falho, seja para autenticar usuários ou validar dados de entrada

[18]Ver resultados em `http://www.modernlifeisrubbish.co.uk/article/top-10-most-common-passwords`.

em uma aplicação. A Microsoft, por exemplo, só mais recentemente apontou segurança como prioridade no desenvolvimento de aplicações[19]. A maioria das empresas prefere enfocar usabilidade e funcionalidades a aterem-se para a questão da segurança dos aplicativos.

Essa situação também pode ser verificada na área acadêmica: a maior parte dos textos acadêmicos sobre Engenharia de *Software*, por exemplo, não aborda o desenvolvimento seguro de aplicações, ou não faz um nível de detalhes adequado. E alguns dos que incluem essa abordagem, só o fazem de maneira mais profunda em novas edições. O resultado é toda uma geração de profissionais sem uma visão mínima de metodologias de desenvolvimento seguro de aplicações. Até que essa situação se modifique na academia e no mercado, vários tipos de ataques, como injeção de código SQL em páginas *Web*, serão comuns na Internet.

Como se não bastasse a ausência de uma cultura de programação segura no desenvolvimento de *software*, esse processo é suscetível a falhas, como qualquer atividade humana. E mesmo o uso de metodologias adequadas não é capaz de garantir o desenvolvimento de um aplicativo completamente seguro. Dessa maneira, mesmo que o aplicativo trate adequadamente entradas e permissões, por exemplo, ele pode possuir falhas (*bugs*) que poderiam ser exploradas por um invasor, fazendo com que o aplicativo execute ações não previstas ou forneça acesso privilegiado ao atacante.

Em geral, boa parte das vulnerabilidades de um *software* tornam-se públicas após descobertas por especialistas da área de segurança ou mesmo *hackers* "bem intencionados". Assim, existem repositórios públicos sobre vulnerabilidades[20] alguns inclusive com provas de conceito: aplicativos que exploram a vulnerabilidade, mostrando sua gravidade. Se esses repositórios auxiliam e muito o especialista de segurança a manter seus *sites* seguros, por outro lado oferece ao possível invasor informações preciosas sobre a fragilidade de um determinado tipo de sistema computacional.

Inclusive, é importante destacar que diversas ferramentas utilizadas pelos invasores, como Nessus ou Metasploit, são na verdade ferramentas de segurança, utilizadas por especialistas da área para verificar e validar a segurança em ambientes computacionais. Assim, por exemplo, um *scanner* como

[19]Ver, por exemplo, `http://www1.folha.uol.com.br/folha/informatica/ult124u14480.shtml` e `http://www.microsoft.com/brasil/corporativo/security/sdl.mspx`.

[20]Merecem destaque, entre esses repositórios o CVE (Common Vulnerabilities and Exposure), disponível em `http://cve.mitre.org/`, e o repositório da SecurityFocus, disponível em `http://www.securityfocus.com/vulnerabilities`.

o Nessus pode ser utilizado para verificar em que pontos um sistema está vulnerável. Um programa de ataque de senha pode ser utilizado para checar se os usuários não estão utilizando senhas fáceis, o que facilitaria uma invasão.

A relação um tanto quanto dúbia entre invasão de sistemas e segurança computacional fica evidente, inclusive, com o fato do Pentest (*Penetration Test* - Teste de Intrusão) ser uma das metodologias utilizadas cada vez mais em testes de segurança computacional. Essa metodologia, descrita em (HERZOG, 2008), consiste no uso de técnicas e ferramentas de invasão para avaliar a qualidade da segurança computacional de um dado sistema. E cada vez mais empresas tem contratado esse tipo de serviço, uma vez que é mais interessante receber um relatório com as falhas encontradas que uma invasão real com danos efetivos.

Em alguns casos, a vulnerabilidade afetada é descoberta não em *software*, mas em um protocolo, o que faz com que todos os aplicativos que implementem aquele protocolo estejam vulneráveis. Recentemente, dois casos desses foram largamente noticiados na mídia da área de TI. O primeiro caso refere-se a uma falha no DNS, que permitia o envenenamento de *cache*[21], fazendo com que servidores DNS respondessem erroneamente às consultas. Invasores poderiam aproveitar dessa falha, por exemplo, para que os usuários de um servidor DNS recebessem endereços incorretos, acessando páginas forjadas. É importante destacar que apesar dessa falha do DNS ter vindo a público em julho/2008, ainda existiam, no momento de escrita desta tese, servidores DNS, inclusive de grandes provedores, que não foram atualizados para evitarem essa falha[22] .

O segundo caso de falha em protocolo noticiado pela mídia de TI foi uma vulnerabilidade recentemente descoberta por dois pesquisadores no protocolo e que permitiria executar com facilidade ataques de negação de serviço[23]. O ataque possibilitado por essa vulnerabilidade teria o impacto de fazer com com que diversos equipamentos de rede (incluindo servidores) fiquem sem co-

[21]Ver alerta em português no *site* da RNP: `http://www.rnp.br/cais/alertas/2008/uscert-vu8000113.html`.

[22]Como noticiado em `http://info.abril.com.br/aberto/infonews/112008/14112008-47.shl`, em novembro de 2008, um quarto dos servidores DNS ainda continua vulnerável e 24% dos profissionais entrevistados não sabiam como corrigir a falha em seus servidores. Além disso, eu mesmo pude verificar, em final de outubro de 2008, que alguns servidores DNS de um provedor banda larga não conseguiam encontrar diversos endereços, por problemas de envenenamento de *cache*. Ao utilizar um servidor DNS alternativo e mais seguro, os endereços puderam ser resolvidos normalmente.

[23]Ver detalhes em `http://arstechnica.com/news.ars/post/20081002-researchers-disclose-deadly-cross-platform-tcpip-flaws.html` e `http://www.computerworld.com.pt/site/content/view/5896/52/`, por exemplo.

nexão, exigindo a reinicialização dos mesmos para voltarem a funcionar corretamente. Os detalhes dessa falha ainda não foram divulgados a público, até o momento de escrita desta tese, mas já teriam sido confirmadas por empresas da área de segurança.

De certa maneira, este tipo de vulnerabilidade ocorre porque a maior parte dos protocolos, especialmente os protocolos básicos da Internet, foram criados em um período em que o controle simples de acesso, por endereço IP por exemplo, era suficiente para garantir a segurança dos serviços. O uso crescente da Internet e novos usos desses protocolos trouxeram à tona problemas não previstos durante o projeto desses padrões. Como eles são adotados em larga escala, é extremamente complexo a correção dessas vulnerabilidades, que exigiriam mudanças significativas nesses protocolos ou mesmo a sua substituição. O IPv6, por exemplo, já tem mais de década de criação e sua adoção ainda é extremamente tímida, o que mostra a inércia na substituição desse tipo de padrão pelo mercado de TI.

Para que se tenha uma idéia da fragilidade dos protocolos básicos da Internet, em agosto de 2008, o CPNI (Centre for Protection of the National Infrastructure), órgão do governo britânico que se ocupa de segurança digital, apresentou a público um relatório apontando fragilidades nas especificações do TCP/IP[24]. O relatório aponta, entre outros riscos, que qualquer sistema construído tendo por base o TCP/IP pode reencarnar falhas de segurança que já foram corrigidas no passado (GONT, 2008). O relatório apresenta algumas melhorias de segurança que poderiam ser implementadas, mas alerta para a necessidade de uma rediscussão dos protocolos sob a ótica da segurança.

3.4 Ataques e Riscos

Como existem diversos tipos de vulnerabilidades, também são inúmeros os tipos de ataques a que um serviço computacional pode estar sujeito. Isso torna-se mais crítico em *servidores*, computadores que disponibilizam, de forma contínua ou periodicamente, um ou mais serviços para vários outros computadores (denominados *clientes* nesse contexto). Como um servidor encontra-se geralmente acessível via internet e em tempo integral, ele é mais suscetível a ataques de todos os tipos e formas que uma estação de trabalho.

[24] O relatório foi noticiado em vários *sites*, entre eles: `http://www.securecomputing.net.au/News/120418,uk-government-blast-tcpip-security.aspx` e `http://info.abril.com.br/blog/virusebugs/20081003_listar.shtml?116172`.

Dependendo de sua configuração, entretanto, é possível torná-lo tão ou mais seguro que estações de trabalho com acesso mínimo à internet. Com a finalidade de melhor situar o leitor, segue-se uma relação dos principais tipos de riscos, ataques e termos correlatos:

***Footprinting*:** Por *footprinting* entende-se a tarefa de coletar informações sobre um sistema alvo. Essa coleta é feita por vias tradicionais e públicas, como uso do comando `finger`, leitura de *páginas* do *site* para obter dados interessantes, consultas a *sites* de busca, etc. Geralmente, o invasor irá verificar quem é o responsável pela administração do sistema, uma vez que invadida a conta desse usuário é possível obter dados mais significativos. Na Figura 3.5, por exemplo, é apresentada uma varredura básica em uma rede, com um alvo em potencial. Nesse caso, a máquina foi identificada tendo como sistema operacional o Windows XP, com o *Service Pack* 2 instalado[25]. Como já foi lançado o *Service Pack* 3, um invasor mais experiente saberia quais as falhas existentes, facilitando suas tentativas de intrusão.

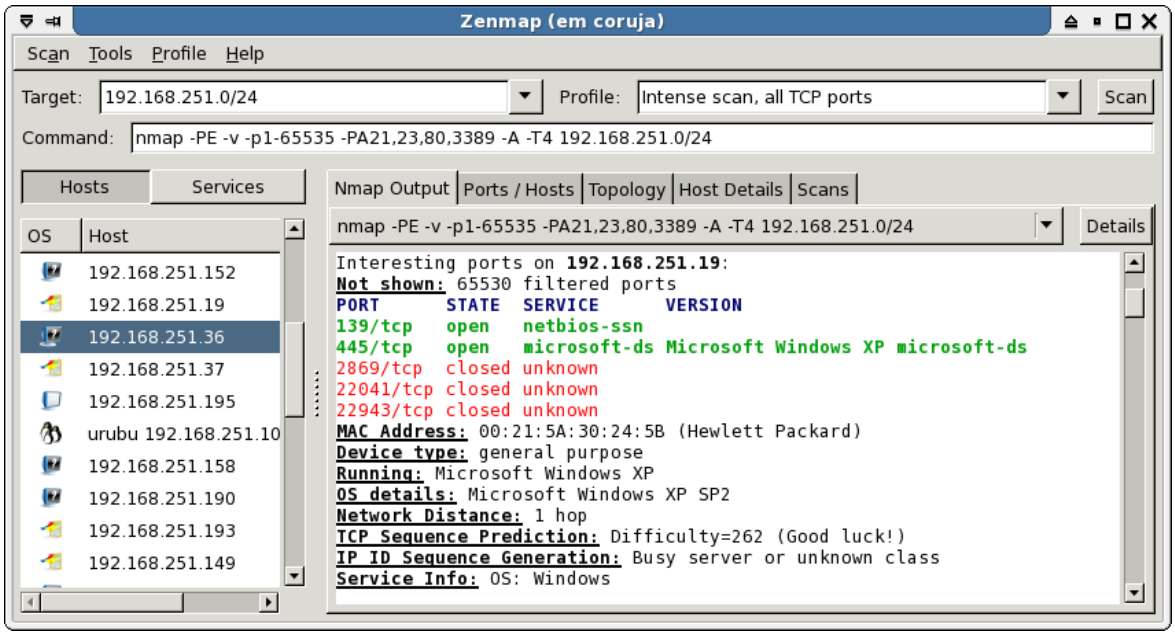


Figura 3.5: Obtenção de Dados via Rede através do Nmap

***Scanning* ou Varredura:** Um *scanner* é um utilitário que verifica vulnerabilidades. Pode ser um *scanner* de sistema, quando checa vulnerabilidades na máquina local (erros no `/etc/passwd`, permissão incorreta de arquivos, etc.), ou pode ser um *scanner* de rede, quando faz varredura de portas de redes, verificando quais estão abertas e, principalmente, quais

[25] A Microsoft lança atualizações de seus aplicativos através de pacotes, os *Service Packs*

estão mais vulneráveis. O objetivo principal desse tipo de ataque é descobrir falhas de segurança devido a *bugs* em serviços de rede ou ausência de proteção para serviços internos. Entre os aplicativos utilizados para essa tarefa, destacam-se Nessus, apresentado na Figura 3.3, OpenVAS[26], Nikto[27] e SARA[28].

***Sniffers*:** Principalmente dentro de uma rede física, onde é facilitada, um ataque muito utilizado é a espionagem eletrônica, com o uso de *sniffer*. Um *sniffer* é um aplicativo que fica "escutando" todos os pacotes de dados que trafegam por uma dada placa de rede. É importante observar que, em várias topologias de rede, um pacote passa por várias placas entre a origem e o destino. Além disso, cabe observar que, para interceptar mensagens destinadas à outras máquinas, a estação deve colocar sua interface de rede em "modo promíscuo"[29]. Esse ataque objetiva principalmente a captura de senhas de usuários internos, uma vez que isso facilita ao invasor a entrada no sistema para detecção de vulnerabilidades. Outro objetivo desse ataque é a captura de informações confidenciais em trânsito na rede. Existem vários aplicativos com essa funcionalidade, destacando-se o Ettercap[30] e o Wireshark[31], ilustrado na Figura 3.6.

***Spoofing*:** Por *spoofing* entende-se a tarefa de fazer uma máquina se passar por outra, forjando, por exemplo, pacotes IPs. Em geral, o usuário irá tentar bloquear o envio de pacotes de dados de uma máquina, tentando se passar por ela. Também podem ser forjados dados de DNS, na tentativa de iludir o usuário, fazendo-o acessar uma página falsa como se autêntica.

***Denial of Service* (DoS):** Pouca atenção tinha sido dado aos ataques de negação de serviço até que servidores importantes, como Amazon, Yahoo e mesmo UOL foram afetados por esses ataques, ficando inativos durante um determinado período de tempo. Como pode ser subentendido, o DoS é um ataque que busca desabilitar um serviço ou mesmo um servidor inteiro. Ultimamente, tem sido muito utilizado o DDoS (*Distributed DoS*), em que um atacante utiliza várias máquinas "zumbis" para enviar

[26] OpenVAS: `http://www.openvas.org/`.
[27] Nikto: `http://www.cirt.net/nikto2`.
[28] SARA: `http://www-arc.com/sara/`.
[29] Por "modo promíscuo" entende-se a configuração de uma interface de rede em que ela captura não apenas os pacotes de rede direcionados a ela, mas também os destinados a outras estações em um mesmo segmento de rede.
[30] Ettercap: `http://ettercap.sourceforge.net/`.
[31] Wireshark: `http://www.wireshark.org/`.

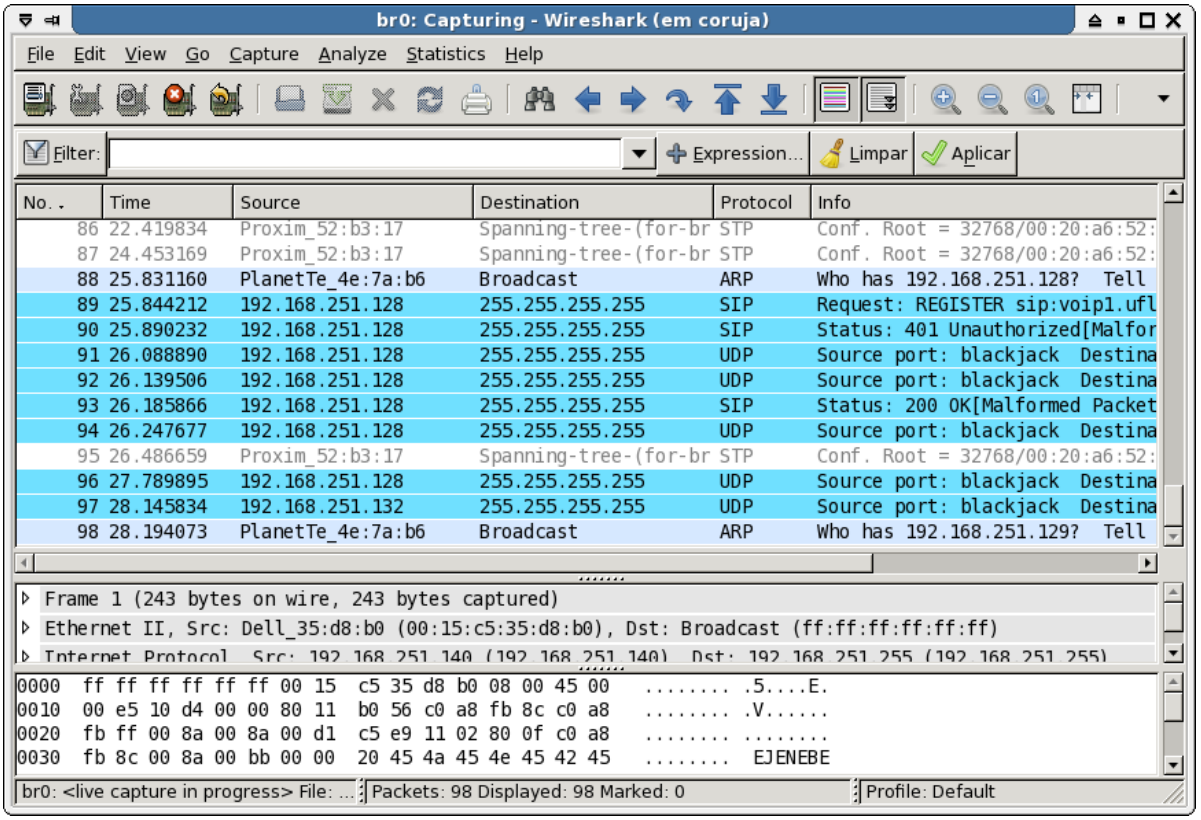


Figura 3.6: Captura de Pacotes da Rede com Wireshark

inúmeras requisições, ao mesmo tempo e de forma sincronizada, a um dado servidor, caso ocorrido em passado recente com a empresa SCO. Isso acaba por ou consumir grande parte da largura de banda de rede ou sobrecarregar um dado *daemon*, derrubando-o. Mais recentemente, pode-se comentar o caso do vírus do Apache, que fazia com que um dado servidor Web ficasse enviando grande quantidade de dados pela rede, conduzindo o tráfego de rede a uma lentidão insuportável.

Testes reais com DDoS foram efetuados por este pesquisador durante a realização da disciplina de Arquitetura de Sistemas Distribuídos, utilizando uma arquitetura similar à apresentada na Figura 3.7. Os testes realizados fizeram uso de apenas 5 máquinas zumbis disparando vários pedidos simultâneos de conexão a um servidor definido pela máquina mestre. Os resultados obtidos foram relativamente assustadores, pois com algumas poucas máquinas zumbis, foi possível bloquear o acesso da rede do CPDEE, na UFMG, aos servidores da UFLA. Na verdade, é possível que o ataque tenha bloqueado também o acesso de outros clientes, mas não houve a intenção de verificar isso durante a realização do trabalho, uma vez que os resultados já eram suficientes para mostrar a gravidade do problema.

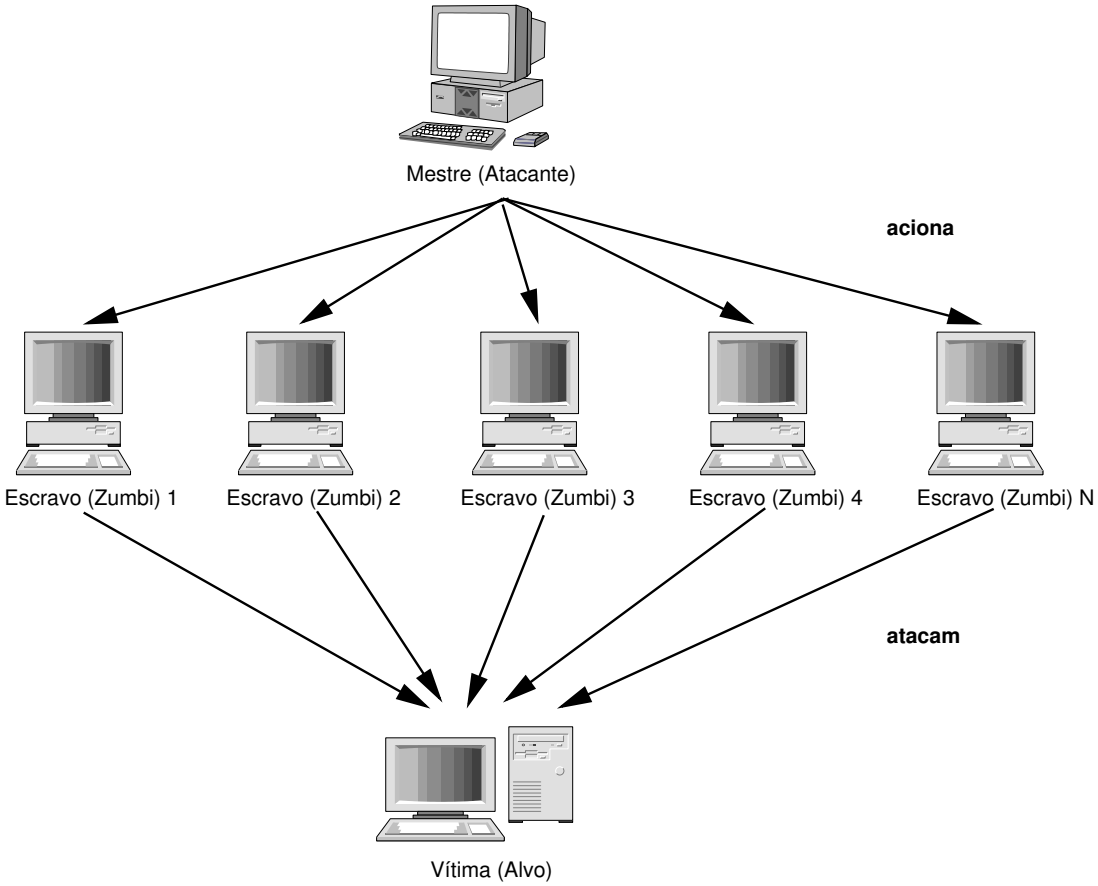


Figura 3.7: Estrutura de Ataque DDoS

Código Malicioso: Código malicioso, ou *malware*, como o nome já sugere, é um *software* criado com finalidades mal intencionadas. Nessa categoria incluem-se os códigos não autorizados (geralmente contidos dentro de um programa legítimo) que efetuam ações desconhecidas e não desejadas pelo usuário. Também são incluídos nessa categoria os programas legítimos que foram alterados para efetuar ações não desejadas pelo usuário. Outros tipos de código malicioso são aqueles que destroem dados sem a intenção do usuário. Dentre os tipos de códigos maliciosos que merecem destaque constam:

- **Cavalos de Tróia**: Um cavalo de tróia, também conhecido como *trojan horse* ou *trojan*, é um programa de computador alterado com finalidades ilícitas. Por exemplo, um atacante poderia substituir o `/bin/login` para não só autenticar usuários, como também armazenar essas senhas em um arquivo oculto.
- **Vírus**: Vírus são semelhantes a *trojans*, dado que efetuam ação não desejada pelo usuário. Uma das diferenças reside no fato que o vírus, uma vez ativado, irá infectar outros arquivos locais. A grosso modo,

vírus não podem infectar máquinas externas sem o auxílio de uma pessoa.

- **Vermes**: Um verme é um programa que pode infectar tanto a máquina local, quanto máquinas remotas, geralmente utilizando falhas de protocolos, serviços ou aplicativos.

Como observado por (HATCH; LEE; KURTZ, 2002), a maior parte dos programas perniciosos existentes são híbridos dessas três categorias. Como exemplo, tem-se o Melissa que era um cavalo de tróia (se passava por um *e-mail* que o usuário estivesse esperando), um vírus (infectava todos os arquivos locais de processamento de texto) e um verme (usava uma falha do Outlook para propagar-se a todos os contatos na agenda de endereços do usuário). No senso comum, vírus e verme são geralmente tomados com um único significado, sendo usado apenas o termo "vírus". Por exemplo, o vírus do Apache era, a bem da verdade, um verme (se propagava a outras máquinas usando o Apache), mas como não infectava outros arquivos no servidor, não poderia ser considerado necessariamente um vírus.

Um tipo de *trojan* extremamente perigoso são os *rootkits*. Como o nome sugere, um *rootkit* é um aplicativo (ou um conjunto de aplicativos) com o objetivo de garantir poderes de superusuário (`root`) ao invasor. Geralmente consistem de aplicativos alterados a funcionar de forma especial pelo usuário ou versões alteradas do próprio *kernel* do sistema operacional. Outro tipo, com difusão mais recente, são os *keyloggers*, que gravam tudo o que é digitado no teclado, enviando um relatório para o invasor.

***Exploits*:** *Exploits* são programas criados para explorar falhas, advindas principalmente de *bugs* nos *daemons* de serviços. Entre as falhas mais exploradas, encontram-se *buffer overflow*, que consiste em estourar o *buffer* de entrada de um servidor, forçando-o a estourar sua memória, devolvendo um *shell* para o invasor[32].

Ataques de Senhas: Esse tipo de ataque consiste em tentar descobrir a senha de um ou mais usuários por força bruta ou usando técnicas heurísticas. Em geral, o invasor tenta obter uma cópia das senhas e efetuar um ataque de dicionário: utilizando variações de palavras em uma dada lista (o dicionário), tenta-se confrontar a senha do usuário com essas variações até descobrir uma que permita o acesso.

[32]Um exemplo prático desse tipo de ataque pode ser verificado em `http://www.cic.unb.br/docentes/pedro/trabs/buffer_overflow.htm`.

Phishing: Recentemente tem-se difundido o uso de mensagens falsas, simulando uma comunicação eletrônica oficial, com o intuito de obter informações sigilosas, como senhas e números de cartão de crédito. Usuários são induzidos a acessarem um *site*, também falso ou a enviarem os dados para um determinado endereço de *e-mail* ou número telefônico. Essa é a manifestação digital mais recente de Engenharia Social, um conjunto de técnicas com a finalidade de obter acesso e informações importantes ou sigilosas em organizações ou sistemas, através da enganação ou exploração da confiança de pessoas[33]. O termo *Phishing* advém da tentativa dos criminosos em "pescar" (em inglês "*fish*") informações confidenciais dos usuários.

SPAM: SPAM[34] é o termo usado para correspondência eletrônica não solicitada, que geralmente é enviada a um grande número de pessoas, possuindo na maior parte dos casos algum caráter apelativo. Quando tem finalidades comerciais, esse tipo de mensagem também é referenciada como UCE (do inglês *Unsolicited Commercial E-mail*). Além de perturbar o usuário, que tem que apagar várias dessas mensagens ao dia, o SPAM causa prejuízos, tanto na queda de produtividade, como na sobrecarga da rede e dos servidores das empresas.

Além das mensagens de propaganda não autorizadas, também constituem prática de SPAM: *hoaxes*, estórias falsas destinadas a criar algum tipo de alarme; correntes, que prometem algum benefício a quem ajudar na propagação da mensagem; golpes (também conhecidos como SCAMs), com ofertas de oportunidades enganosas ou produtos que prometem falsos resultados. SPAMs também são enviados com o intuito de difusão de *malware*, bem como *phishing*.

Intrusão de Sistemas: Por intrusão de sistema compreende-se o acesso não-autorizado a um sistema computacional, seja ele um servidor ou apenas um determinado serviço. Essa intrusão pode-se dar de diversas formas e com diferentes objetivos, podendo ser a ação de um *hacker* em busca de apenas colocar seus conhecimentos à prova ou *crackers* com intenção explicitamente ilícita.

[33] Mais detalhes sobre Engenharia Social podem ser encontrados no Capítulo 10 de (MITNICK; SIMON, 2006).

[34] O termo SPAM surgiu de uma esquete de humor do programa inglês Monty Phyton. Numa das esquetes, personagens repetiam ao extremo a marca de um enlatado (Spam), causando incômodo a outros. A grafia mais adequada para isso seria, portanto, "spam". Esse texto mantém a grafia em maiúscula, por seu uso relativamente habitual em textos sobre o problema.

Como já comentado, às vezes, a invasão é feita apenas em busca de fama, principalmente por adolescentes, mas não somente. Nesse caso, intenciona-se deixar uma "marca" da invasão, o que é feita geralmente pela adulteração (*defacement*) das páginas de um determinado *site*. Entretanto, dependendo o invasor e o nível de acesso, informações sigilosas podem ser roubadas ou alteradas, bem como podem ser iniciados processos com vistas à obtenção de benefícios ilícitos. Cabe destacar que não só esse invasor pode ser alguém interno à instituição, como o próprio ataque poderia estar sendo feito a partir dos recursos e privilégios internos.

É importante ressaltar que, se até pouco atrás os alvos preferenciais eram os servidores de médias e grandes empresas, hoje mesmo o usuário final é um ualvo potencial. Máquinas de usuários comuns podem ser invadidas para seqüestro de dados, um ataque recente na internet, mas principalmente para serem utilizadas como ponto de partida para ataques a outros locais. O que se objetiva, nesse caso, é dificultar o rastreamento de um determinado ataque a um alvo mais promissor. A invasão à máquina de um usuário comum é feita geralmente com o uso de *malware*, difundido em boa parte dos casos com o uso de SPAM ou *phishing*.

As vulnerabilidades e falhas de segurança existentes tornam as redes de computadores um "terreno" relativamente perigoso, onde dados confidenciais precisam ser protegidos. Isso exige a tomada de uma série de medidas por parte dos administradores de TI, algumas das quais são apresentadas na seção a seguir.

3.5 Política de Segurança

O termo *Segurança Computacional* e seus correlatos, como *Segurança da Informação*, *Segurança de TI*, ou *Segurança em Redes de Computadores*, são muito utilizados atualmente, mas sem uma consciência exata a que se referem. Isso ocorre principalmente porque existe uma consciência da necessidade de Segurança Computacional, mas sem uma real avaliação do que isso implica. Assim, é preciso estar atento a qual significado os termos estão sendo utilizados. Dada a relatividade dos termos, é preciso refletir sobre quais ítens devem ser levados em conta ao abordar essa questão. Em geral, é possível destacar os seguintes elementos de um ambiente computacional, sob o ponto de vista da segurança:

Confiança: É possível confiar na disponibilidade do sistema? Os dados armazenados vão estar acessíveis quando forem necessários? Os mecanismos de *backups* são suficientes para garantir que as informações armazenadas possam ser recuperadas com facilidade em caso de problemas?

Integridade: Os dados recuperados são confiáveis? Como garantir que as informações não foram alteradas na fonte ou no tráfego de dados? Como garantir que o que foi acessado é idêntico ao que foi armazenado?

Confidencialidade: Como certificar que os dados só podem ser acessados por quem de direito? Como garantir a privacidade dos usuários e dos dados? Como impedir a espionagem de informações?

Custo/Benefício: Qual o custo das soluções de segurança adequadas ao ambiente? Qual abordagem possibilita melhor aproveitamento dos recursos? Qual solução otimiza os investimentos?

Essas questões irão chamar o administrador para a necessidade de refletir a respeito: "*o que é segurança computacional em meu ambiente?*". Dessa maneira, o elemento inicial mais importante para garantir segurança em uma rede de computadores é o estabelecimento de uma *política de segurança*. Enquanto a instituição não define o que é mais importante para o estabelecimento da segurança computacional em seu ambiente, as atitudes são tomadas sem a devida consciência e sem garantir um mínimo desejável de segurança:

> Uma Política de Segurança incorpora os resultados de uma análise de risco em um plano que providencia procedimentos para gerenciar um ambiente computacional. Em particular, ela fornece ao administrador do sistema linhas operacionais para o ambiente, tais como regras para o gerenciamento de contas de usuários, procedimentos de instalação de sistemas ...
>
> (MANN; MITCHELL, 2000, p.14)

> Uma política de segurança é um conjunto de leis, regras e práticas que regulam como uma organização gerencia, protege e distribui suas informações e recursos. Um dado sistema é considerado seguro em relação a uma política de segurança, caso garanta o cumprimento das leis, regras e práticas definidas nessa política.
>
> (SOARES; LEMOS; COLCHER, 1995, p.450)

É a política de segurança que vai deixar claro o que deve ter acesso restrito e o que pode ser liberado sem problemas. É ela que define quais são os ítens que precisam ser preservados, bem como quais são as pessoas que têm acesso a determinados recursos. Sem uma política de segurança clara, não se sabe o que vai se proteger, nem porque ou qual a melhor forma.

Uma boa política de segurança irá definir, por exemplo, como será estabelecida a política de uso. Essa política de uso é responsável por deixar claro o que o usuário pode e não pode fazer com determinados recursos. Em geral, tal política é um documento a ser assinado pelo usuário em questão. Uma política de segurança deve possuir as seguintes características:

- ser implementável através de procedimentos de administração de sistema, regras de uso, ou outros métodos apropriados;
- ser reforçada com ferramentas de segurança e sanções;
- definir claramente as áreas de responsabilidade para usuários e administradores do sistema.

Dessa maneira, atentando-se para essas diretrizes, é possível elaborar uma política de segurança válida para o ambiente em questão. Além disso, é imprescindível que a política de segurança atente-se para os seguintes ítens:

Segurança física: Como os equipamentos da instituição em questão serão protegidos? Como garantir a integridade física dos dados? Como garantir que não haverá acesso físico a dados que deveriam estar protegidos? Esse ítem é mais importante do que se imagina: de nada adianta senhas no *hardware* ou bem elaboradas, se qualquer funcionário recém demitido pode roubar o disco rígido do servidor para vender ao concorrente.

Segurança lógica: Como garantir integridade lógica dos dados? Como os dados da instituição em questão serão protegidos? Como garantir que não haverá acesso lógico indevido a dados? Quais são os dados mais importantes? Os casos de invasões tem-se tornado cada vez mais freqüentes e é imprescindível estar atento a isso.

Privacidade: O que fazer para proteger a privacidade dos usuários? A instituição irá proteger essa privacidade? Observe que a instituição deve estar atenta aos aspectos legais dessa questão.

Legalidade de *Software*: Como garantir um bom uso dos recursos, impedindo a pirataria? A responsabilidade pelos aplicativos instalados é de quem? da empresa? do usuário? Isso precisa estar claro em algum documento da política de segurança.

Além disso, é imprescindível que o administrador pratique vigilância e perseverança, assumindo sempre que os mal-intencionados sabem mais que ele. A adoção de uma política de segurança irá, na maioria das vezes, implicar em perda de performance ou conveniência do usuário. Alguns serviços devem ser evitados e isso gera problemas para usuários mais inexperientes. Além disso, o tráfego criptografado ocupa maior largura de banda, diminuindo a velocidade de transmissão de dados. Assim, deve-se pesar os prós e contras de qualquer atitude envolvendo segurança antes de implementá-la.

Em geral, a política de segurança da maior parte das instituições consiste das normas de uso dos recursos e uma política de uso. A política de uso é um documento que deixa claro o que o usuário pode e não pode fazer com os recursos dentro da instituição. Para garantir validade jurídica desse documento, é recomendável a sua validação pela assinatura do usuário ou em contrato coletivo de trabalho. Obviamente, dependendo do tamanho da empresa em questão, a política de segurança será baseada em normas formais, como a NBR ISO/IEC 27002 (ABNT, 2005).

É importante destacar que uma política de segurança deve estipular uma série de práticas para sua adequada implementação, incluindo uma série de diretrizes técnicas. Por exemplo, a política de segurança deve definir como serão efetuadas as cópias de segurança e com qual periodicidade. Além disso, qual estratégia a ser adotada no controle de acesso aos dados e serviços, destacando-se o uso de *firewalls* para filtrar o acesso aos recursos. A política também deve apontar quais mecanismos adotar para a transferência segura de dados em trânsito, o que é feito geralmente com o uso de criptografia.

3.6 Detecção de Intrusão em Redes de Computadores

Vulnerabilidades são descobertas com freqüência e é possível falar com absoluta tranqüilidade que não existem servidores 99% seguros. O que se pode pretender é um servidor que ofereça tanta dificuldade que ele desesti-

mule os invasores. Mas mesmo com esse nível de dificuldade, não é possível confiar cegamente no sistema. Dessa maneira, o administrador deve estar utilizando ferramentas de detecção e prevenção de intrusos para monitorar o sistema de sua responsabilidade, com o objetivo de impedir que ataques em fase inicial consigam chegar a um nível indesejado de intrusão no sistema.

Parte do serviço de prevenção de intrusos é feito com uma implementação de uma política de segurança adequada. Obviamente, essa política deve estar baseada em serviços criptográficos, uma correta configuração de serviços e *firewall*, entre outros. Dessa maneira, a dificuldade gerada servirá como uma prevenção adequada de intrusos. Mas isso não é suficiente, exigindo o uso de mecanismos que permitam alertar ou mesmo bloquear tentativas de ataque. O processo de detecção de intrusos envolve inúmeras estratégias. Geralmente são utilizadas ferramentas IDS (*Intrusion Detection System* - Sistema de Detecção de Intrusos). É importante notar que esse termo pode ser usado de várias maneiras, de forma mais ampla ou mais restrita.

Em uma abordagem mais restrita, IDS refere-se apenas aos aplicativos capazes de alertar quando uma tentativa de invasão encontra-se em ação. Nesse sentido, constituem-se principalmente em programas de monitoramento de conexões de rede, como o Snort[35]. Em uma visão mais ampla, também são IDS as ferramentas utilizadas para monitorar a integridade do sistema. Nesse caso, também podem ser definidos claramente como IDS os verificadores de integridade de arquivos, como o AIDE[36] ou o Tripwire[37]:

> Técnicas de Detecção de Intrusos se aproximam bastante daquelas usadas em *Firewalls* e sistemas de *Log*, e o seu objetivo principal é reagir a uma invasão (ou suspeita de invasão) no menor intervalo de tempo possível. Isto pode ser feito, por exemplo, monitorando-se continuamente o tráfego de rede, à procura de qualquer anomalia, ou então analisando-se continuamente as últimas entradas dos arquivos de *log*, à procura de ações suspeitas.
>
> (WEBER, 2000)

Em (NAKAMURA; GEUS, 2002) é proposta uma classificação de ferramentas IDS nas seguintes categorias:

HIDS – *Host-Based Intrusion Detection System*: monitora o sistema utilizando informações locais, como arquivos de *logs* ou agentes de auditoria.

[35]Snort: `http://www.snort.org/`.
[36]AIDE: `http://www.cs.tut.fi/~rammer/aide.html`.
[37]Tripwire: `http://www.tripwire.com/`.

O HIDS pode ser capaz de monitorar acessos e alterações em arquivos e processos no sistema, entre outros aspectos. Exemplos de IDS nessa categoria são o Tripwire, o AIDE, OSSEC [38] e as Sentrytools[39].

NIDS – *Network-Based Intrusion Detection System*: monitora o tráfego de rede, geralmente utilizando a interface da rede em modo promíscuo, como se fosse um *sniffer*. Exemplos de IDS nessa categoria são o Snort e o AAFID[40].

Hybrid IDS – *Hybrid Intrusion Detection System*: utiliza monitoração do sistema e da rede ao mesmo tempo para prevenir ataques. Com isso tem-se o melhor dos sistemas HIDS e NIDS. Um exemplo de IDS nessa categoria é o Prelude[41].

Como comentado em (KIM *et al.*, 2007), HIDS tem a vantagem de verificar se um ataque foi bem sucedido, bem como detectar ataques locais e exploração de privilégios[42]. Entretanto, esses sistemas são difíceis de manter e configurar, uma vez que precisam ser adaptados a cada máquina onde são instalados. NIDS, por outro lado podem ser centralizados em um roteador da rede, por exemplo e exigem menos esforço para sua configuração e manutenção. Entretanto, são incapazes de detectar vários tipos de ataques, inclusive os encriptados.

Entre as ferramentas IDS, merecem destaque aquelas que trabalham com mecanismos ativos de detecção de intrusão. Esse processo refere-se principalmente ao uso de ferramentas que monitoram o sistema ou, principalmente, a rede, efetuando ações pré-estabelecidas tão logo algo estranho seja detectado. A filosofia, de certa forma, é extremamente simples: o IDS analisa continuamente o sistema ou a rede e tão logo reconheça um padrão estranho, algum mecanismo de alerta ou de defesa é acionado, dependendo do caso.

Nesse sentido, é possível dizer que sistemas IDS funcionam de forma semelhante aos sistemas anti-vírus atuais, que continuamente ficam analisando arquivos inseridos no computador ou que chegam via rede. Uma tentativa de invasão, assim como um vírus, pode ser detectada por um padrão. Não será de estranhar se, num futuro próximo, as empresas desenvolvedoras de anti-

[38] OSSEC: `http://www.ossec.net/`.
[39] Sentrytools: `http://sourceforge.net/projects/sentrytools/`.
[40] AAFID: `http://www.cerias.purdue.edu/about/history/coast/projects/aafid.php`.
[41] Prelude: `http://www.prelude-ids.com/`.
[42] Quando um usuário tenta obter mais acesso do que o permitido.

vírus acabem por inserir ferramentas IDS em seus produtos ou transformar seus produtos em IDS.

Uma invasão geralmente deixa rastros. Talvez, inclusive, seja possível dizer que, da mesma forma que não existe um sistema totalmente seguro, não existe uma invasão perfeita. Assim, a verificação periódica dos arquivos de registros pode evitar surpresas extremamente desagradáveis, ao mostrar a tentativa de invasão desde o seu início.

Um esclarecimento a ser feito é que, em um sistema medianamente seguro, uma invasão é um procedimento relativamente demorado. Assim, o leitor deve excluir de sua imaginação a imagem romântica de um *hacker* que consegue penetrar em um sistema em poucos minutos. A menos que o sistema seja uma peneira de vulnerabilidades, uma invasão irá exigir esforço e paciência do intruso, que terá que fazer inúmeras tentativas para conseguir seu intento.

Caso haja uma verificação periódica dos *logs*, por exemplo, uma tentativa de invasão pode ser bloqueada em seu início. Além disso, os arquivos de registros podem indicar falhas em serviços, o que poderia comprometer não só a segurança, mas a qualidade do sistema. Outro motivo para a verificação periódica dos *logs* é a possibilidade de verificação de ações anormais no sistema, como *logins* fora do padrão ou tentativas de execução de aplicações restritas.

Outro mecanismo poderoso para a verificação da intrusão é a monitoração dos processos de sistema, em busca de anomalias. Em geral, tentativas de invasão sobrecarregam o número de processos do sistema, ou fazem com que determinados processos consumam mais recursos, como memória e CPU, comparados a uma execução normal. Isso ocorre porque em geral o invasor envia uma grande quantidade de dados ou de pedidos de conexão, em busca de fragilidades no sistema.

Um problema parecido ao detecção de intrusos, a detecção de SPAM consiste na análise da mensagem, incluindo seu texto e cabeçalho, com a finalidade de determinar a probabilidade da mesma enquadrar-se nessa categoria. Em geral, esse processo é implementado como filtro, seja em um dos serviços de *e-mail*[43], seja no cliente do usuário.

Deve-se destacar que o processo de reconhecimento, tanto de intruso como de SPAM, pode gerar falsos positivos ou falsos negativos, como a maioria dos

[43]Como comentado em (UCHÔA, 2005), o processo de envio e recebimento de *e-mail* envolve mais de um tipo de servidor. Em geral, inclui: servidor SMTP (para envio e recebimento das mensagens) e servidor IMAP ou POP ou Webmail (para leitura das mensagens pelo usuário). Em geral, filtros anti-SPAM, quando instalados no servidor, são configurados para funcionar em conjunto com o servidor SMTP.

sistemas de reconhecimento de padrão. Tomando-se SPAM como exemplo, tem-se como falsos negativos as mensagens que são caracterizadamente SPAM e que seriam entregues ao usuário como se fossem mensagens legítimas. Por outro lado, o sistema anti-SPAM poderia detectar como sendo SPAM uma mensagem importante para o usuário, evitando (ou atrapalhando) a entrega de uma mensagem legítima. Como pode ser inferido desse exemplo, em geral tem-se maior aceitação para falsos negativos que para falsos positivos, tanto na detecção de intrusos como na detecção de SPAM.

Os IDSs atuais geralmente são construídos ou baseados em detecção de anomalia ou detecção de mal uso, como apontado em (BIERMANN; CLOETE; VENTER, 2001). Quando baseados em detecção de anomalia, o IDS irá tomar por base que um ataque será diferente da atividade normal, e o invasor irá exibir um padrão de comportamento diferente do usuário legítimo. Por outro lado, a abordagem de detecção de mal uso é baseada num repositório de técnicas conhecidas de intrusão – invasões são descobertas a partir da comparação de dados do sistema com aqueles armazenados no repositório. Ainda em (BIERMANN; CLOETE; VENTER, 2001), é possível verificar uma comparação de várias técnicas de IDSs com relação ao tipo de funcionamento e alguns outros critérios comparativos. Esse artigo, entretanto, não inclui SIAs entre as abordagens adotadas no desenvolvimento de IDS.

Como comentado em (AICKELIN; GREENSMITH; TWYCROSS, 2004), sistemas baseados em detecção de mal-uso irão apresentar um número muito baixo de falsos positivos, mas serão incapazes de detectar novos tipos de ataques, ou ataques ofuscados. Assim, esses sistemas geram um grande número de falsos negativos. Por outro lado, sistemas baseados em detecção de anomalias são capazes de detectar novos tipos de ataques, mas costumam gerar um grande número de falsos positivos. Isso deve-se principalmente ao fato que o uso legítimo de sistemas computacionais é dinâmico, modificando-se com o passar do tempo. Obviamente, tratar essas questões é uma grande dificuldade no projeto e desenvolvimento de um IDS.

3.7 Comentários Finais

USUÁRIOS de computadores tem-se deparado cada vez mais com problemas sérios de segurança em redes de computadores. Dessa maneira, tem crescido os esforços e estudos na área de Segurança Computacional, com vistas a atacar esses e outros problemas que impedem um uso adequado dos

recursos computacionais. Este capítulo teve o objetivo de apresentar portanto, os conceitos básicos na área de Segurança Computacional, bem como os principais tipos de riscos e ataques existentes nas redes de comunicação de dados.

O capítulo foi produzido com o objetivo de conscientizar o leitor dos problemas envolvendo segurança computacional e a necessidade de busca de soluções para a área. Uma visão mais completa sobre o assunto pode ser encontrada em (NAKAMURA; GEUS, 2002), (SCHNEIER, 2001) e (CHESWICK; BELLOVIN; RUBIN, 2005). Uma introdução ao assunto em uma linguagem bem acessível ao usuário leigo é propiciada pela Cartilha de Segurança na Internet[44], um livro produzido pelo CERT.br (Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil). Além do livro, alguns vídeos sobre o assunto estão disponibilizados. Detalhes sobre algumas vulnerabilidades em redes TCP/IP, bem como mecanismos de exploração dessas falhas podem ser verificadas em (MELO, 2004). Em (NEY, 2006), são apresentadas diversas técnicas para verificação de vulnerabilidades utilizando-se do Nmap. Técnicas clássicas para detecção de intrusos em redes de computadores podem ser encontradas em (NORTHCUTT; NOVAK; MCLACHLAN, 2001).

É importante esclarecer que este capítulo foi produzido principalmente a partir da experiência adquirida pelo autor em administração de laboratórios e uso de diversos sistemas operacionais. Essa experiência em administração de sistemas foi enriquecida pela leitura de diversos materiais, destacando-se, além dos textos citados: (NEMETH *et al.*, 1995), (NEMETH *et al.*, 2001), (NEMETH; SNYDER; HEIN, 2002),(STANFIELD; SMITH, 2001), (WIRZENIUS *et al.*, 2002), (FRAMPTON, 1999), (MANN; MITCHELL, 2000), (ANONYMOUS, 2000),(KIRCH; DAWSON, 2002), (BAUTTS; DAWSON; PURDY, 2005), (HATCH; LEE; KURTZ, 2002), (SCHNEIER, 1996) e (NIC BR SECURITY OFFICE, 2003). A essas referências devem ser acrescidos vários *HOWTOs*[45] disponibilizados pela *The Linux Documentation Project*[46]. Entre esses *HOWTOs*, destacam-se (FENZI, 2004), (BURGISS, 2002a) e (BURGISS, 2002b).

[44]Cartilha de Segurança na Internet: `http://cartilha.cert.br/`.

[45]Um *HOWTO* é um pequeno guia que ensina um usuário a configurar um serviço ou fazer uma dada tarefa.

[46]*The Linux Documentation Project*: `http://www.tldp.org/`

CAPÍTULO 4

Imunologia e Imunoinformática

(...) Mas, como a realidade pensada não é a dita mas a pensada,
Assim a mesma dita realidade existe, não o ser pensada.
Assim tudo o que existe, simplesmente existe.
O resto é uma espécie de sono que temos,
Uma velhice que nos acompanha desde a infância da doença.

Fernando Pessoa

O peito, o corpo, é sempre uno, mas as almas que nele residem não são nem duas, nem cinco, mas incontáveis, o homem é um bulbo formado por cem folhas, um tecido urdido com muitos fios.

Herman Hesse, O Lobo da Estepe

4.1 Comentários Iniciais

ESTE trabalho tem por objetivo principal a utilização de algoritmos inspirados no funcionamento do sistema imune para aplicação em problemas de Segurança Computacional. Assim, este capítulo tem o propósito de apresentar conceitos básicos que auxiliem na compreensão da proposta do trabalho de pesquisa e desenvolvimento. Dessa maneira, a Seção 4.2 e a Seção 4.3 apresentam um conjunto mínimo de conceitos do sistema imune, com uma breve discussão sobre visões acerca do seu funcionamento.

Na parte final deste capítulo, na Seção 4.5, é apresentada a Imunoinformática, uso de técnicas de Bioinformática em Imunologia. Na Seção 4.6 são apresentados conceitos básicos sobre sistemas complexos e autômatos celulares, conceitos esses fundamentais para compreensão da área de modelagem do sistema imune, apresentada na Seção 4.7. O objetivo inicial de apresentar esses conceitos é principalmente ampliar a visão do leitor sobre as possibilidades de relacionamento entre Informática e Biologia. A motivação maior, entretanto, é o fato que alguns pesquisadores de algoritmos imunoinspirados também trabalham com modelagem do sistema imune e uma visão mais geral sobre o assunto irá permitir uma melhor compreensão da área e tipo de pesquisa utilizadas neste trabalho.

O foco de interesse desta pesquisa, entretanto, recai sobre a área de Sistemas Imunes Artificiais, um tipo de algoritmo imunoinspirado, e que serão discutidos no Capítulo 5. Essa área possui vários pontos de contato com a Imunoinformática e consiste na aplicação de conceitos imunológicos na busca de solução de problemas de Engenharia. Optou-se neste texto por uma apresentação inicial dos conceitos de Imunoinformática antes de Sistemas Imunes Artificiais, uma vez que isso permite uma apresentação dos conceitos indo do sistema biológico ao simulacro *in silico*, o que de certa forma traduz o espírito do trabalho apresentado neste texto.

4.2 O Sistema Imune

O objetivo desta seção é apresentar sumariamente alguns conceitos básicos de Imunologia aplicáveis ao desenvolvimento deste trabalho. Não se pretende aqui expor uma visão geral ou completa sobre o sistema imune, mas apenas destacar pontos-chave para o desenvolvimento da pesquisa. Para mais detalhes sobre o sistema imune e seu funcionamento, recomenda-se a leitura de (JANEWAY JR., 2001), (JANEWAY *et al.*, 2001), (CALICH; VAZ, 2001) e (VAZ; FARIA, 1993).

A Imunologia é o ramo da Biologia que estuda todos os aspectos do sistema imune nos diversos organismos, tratando, entre outros assuntos, do funcionamento fisiológico do sistema imune nos estados de saúde e doença do organismo. O termo é derivado do latim "*immunis*", cujo significado literal é "isento de encargo", aplicado ao sujeito livre de impostos, de encargos pesados, e a pessoas protegidas ou beneficiadas em relação às demais. No

contexto biológico, imunidade refere-se, obviamente, aos indivíduos isentos ou protegidos contra doenças.

A Imunologia é uma ciência relativamente recente, sendo que sua origem é atribuída a Edward Jenner, que descobriu em 1798 que a *vaccinia* ou *cowpox*, o agente infeccioso da varíola bovina, quando injetada no organismo humano, proporcionava imunidade à varíola (JENNER, 1988). A varíola é uma doença frequentemente fatal, mas encontra-se atualmente extinta, graças à vacinação em massa. O termo 'vacinação', inclusive, é devido a Jenner, utilizado para batizar o processo de inocular indivíduos são com amostras atenuadas ou mortas de agentes causadores de doenças, com vistas à proteção futura contra enfermidades.

Como apontado em (DE CASTRO, 2001), Jenner desconhecia os agentes infecciosos que causavam as doenças, tarefa que coube a pesquisadores do final do século XIX, destacando-se Robert Koch e Louis Pasteur. A Koch, por exemplo, deve-se a descoberta de que as doenças infecciosas eram causadas por microorganismos patogênicos, em 1876, fato depois confirmado por Pasteur em 1878. Atualmente são conhecidos quatro grande categorias de microorganismos causadores de doenças: vírus, bactérias, fungos e parasitas. As descobertas do final do século XIX possibilitaram um grande desenvolvimento na Imunologia, com uso da vacinação em várias outras doenças. Pasteur, por exemplo, desenvolveu com sucesso a vacina anti-rábica e vacina para a cólera aviária.

Os triunfos práticos obtidos com a vacinação levaram à busca dos mescanismos envolvidos no processo de imunização, uma vez que desconhecia-se os princípios de funcionamento do sistema imune. Em 1890, Emil von Behring e Shibasburo Kitasato demonstram que a proteção obtida nos processos de vacinação deviam-se ao surgimento de fatores de proteção, os anticorpos, no soro dos indivíduos vacinados. Essa data marca o início da teoria humoral da imunidade, uma vez que mostrou-se que os anticorpos eram capazes de se ligarem aos agentes infecciosos, neutralizando-os. Por causa da descoberta sobre a produção de anticorpos, Emil von Behring recebeu o primeiro prêmio Nobel de Medicina, em 1901.

Por volta da mesma época, Elie Metchnikoff propõe a teoria celular da imunidade, apontando que algumas células dos organismos eram capazes de "comer" microorganismos. Tais células foram denominadas de fagócitos e Elie propôs que elas eram o principal mecanismo de defesa do corpo contra os microorganismos. Como comentado em (DE CASTRO, 2001), essa foi a primeira

grande controvérsia em Imunologia, uma vez que iniciou-se uma disputa entre as teorias humoral e celular da imunidade, conflito que só foi resolvido em 1904 quando Almorth Wright e Joseph Denys demostram que os anticorpos eram capazes de se ligarem a bactérias, promovendo a destrução das mesmas por fagócitos.

Um pouco antes, em 1900, Paul Erlich apresenta um trabalho (ERLICH, 1988) sobre a formação dos anticorpos, formulando uma teoria, cuja principal premissa era a de que a superfície dos leucócitos é coberta com diversas cadeias laterais ou receptores, que formariam ligações químicas com os antígenos encontrados. De acordo com sua teoria, dado qualquer antígeno, moléculas capazes de serem reconhecidas pelo sistema imune, ao menos um receptor seria capaz de o reconhecer e se ligar a ele. A informação para a produção dos anticorpos seria providenciada pelos genes do animal. Ainda de acordo com sua teoria, o contato com um dado antígeno seria responsável por selecionar e estimular uma célula a sintetizar os receptores particulares como anticorpos (DE CASTRO, 2001). Cabe ressaltar que o prêmio Nobel de Medicina de 1908 foi dividido entre Metchnikoff e Erlich, graças às contribuições de ambos à Imunologia.

A idéia de seleção de anticorpos a partir dos antígenos foi obscurecida nas décadas de 1910 a 1950, sendo que nessa época o principal avanço deu-se na descoberta do sistema complemento, por Jules Bordet, que recebeu o prêmio Nobel de Medicina em 1919 por este fato. O sistema complemento é composto por um conjunto de proteínas que atuam no sistema imune de diversas maneiras, principalmente opsonizando os patógenos e induzir uma série de respostas inflamatórias que auxiliam no combate à infecção. Podem também ligar-se a diversas toxinas, neutralizando a ação das mesmas. Em diversas situações atuam em conjuntos com os anticorpos, sendo ativados por esses para desencadear outras ações do sistema imune.

Em 1946, George Snell e Peter A. Gorer identificam o Complexo de Histocompatibilidade Principal, MHC (*Major Histocompatibility Complex*), em camundongos. Snell recebeu o Nobel de Medicina em 1980, junto com Baruj Benacerraf e Jean Dausset por suas descobertas referentes a estruturas, determinadas geneticamente na superfície celular, que regulam reações imunológicas. O MHC é uma região genômica responsável por proteínas expressas nas células de superfícies na maioria dos vertebrados, tendo por principal função a apresentação de antígenos próprios (framentos peptídicos da própria célula) e não-próprios (fragmentos de microorganismos) para as células T. O nome desses genes e das proteínas associadas advém do fato que o MHC está

intimamente ligado ao processo de rejeição ou aceitação, a compatibilidade, de órgãos transplantados.

O seletivismo em Imunologia foi reativada nos anos 1950 por Niels Jerne, que afirmou que o organismo apresenta uma população diversificada de anticorpos naturais, mesmo na ausência de antígenos. O antígeno se combinaria a anticorpos em circulação que possuissem estrutura complementar a esse antígeno, fomentando a seleção dos mesmos. Por essas e outras teorias e contribuições Jerne merecidamente recebeu o Nobel de Medicina em 1984. Sua especial contribuição à Imunologia vem do início da década de 1970, com sua teoria de rede idiotípica (JERNE, 1973; JERNE, 1974), segundo a qual os antígenos próprios, incluíndo os próprios anticorpos, afetam a diversidade e regulação da resposta imunológica.

Mas foi com McFarlane Burnet que o seletivismo tomou força em Imunologia, com sua teoria de seleção e expansão clonal (BURNET, 1959). Burnet recebeu um Nobel de Medicina em 1960, junto com Peter Medawar pelo trabalho de ambos em tolerância imunológica. A teoria ou hipótese de seleção clonal estipula que cada linfócito expressa em sua superfície um único tipo de anticorpo e que esses seriam selecionados por estímulação de antígenos: uma vez que a célula se ligasse a um dado antígeno, ela iria se proliferar clonalmente e secretar anticorpos no soro do indivíduo.

É importante aqui destacar que a teoria de seleção clonal de Burnet, bem como a teoria da rede idiotípica de Jerne são a base da maior parte dos algoritmos imunoinspirados em uso atualmente pela comunidade acadêmica. Mais recentemente, também tem sido desenvolvidos algoritmos e sistemas imunes artificiais inspirados na visão de Polly Matzinger sobre o funcionamento do sistema imune, conhecida como Modelo ou Teoria do Perigo (MATZINGER, 1994). Poucos trabalhos se apóiam em outras descobertas mais recentes da Imunologia, como a descoberta da estrutura e geração da diversidade das moléculas de anticorpos por Susumo Tonegawa, que recebeu o Nobel de Medicina em 1987 por este fato.

Poucos trabalhos computacionalmente inspirados na Imunologia também utilizam das descobertas mais recentes sobre o papel de componentes inatos do sistema imune, como o MHC e os *Toll Like Receptors*. Sobre os Toll Like Receptors, é importante destacar que, em 1997 ficou demonstrado que, quando ligados, os mesmos induzem a ativação de certos genes necessários ao início da resposta imune (MEDZHITOV; PRESTON-HURLBURT; JANEWAY, 1997). De certa maneira, isso ressaltou a importância de mecanismos não-

adaptativos como elementos ativadores dos mecanismos adaptativos dos sitema imune.

O MHC, por sua vez, está presente no processo de apresentação de antígenos, uma vez que algumas células do sistema imune não se ligam diretamente aos antígenos, mas apenas através daqueles que lhe são "apresentados" por outras células do organismo. Isso ressalta o papel da interação dos componentes do sistema imune, bem como a complexidade de seu funcionamento. No caso de apresentação de antígenos a células T por células B, o "*antígeno precisa inicialmente ser internalizado e processado por células B específicas e então apresentado a células T em uma maneira restrita por MHC (...)*" (LANZAVECCHIA, 1985). Isso ressalta a complexidade do sistema imune, que envolve um grande conjunto de elementos que interagem entre si e com outros elementos do organismo, bem como elementos externos, como os microorganismos.

Nesse sentido, o *sistema imune* é geralmente compreendido como um mecanismo de defesa do organismo contra agentes infecciosos, os *patógenos*. Nesses termos, *antígenos*, que são componentes dos patógenos, interagem com o sistema imune e desencadeiam um conjunto de atividades que resultam na produção e elevação do nível de anticorpos que reagem com esses antígenos. Os *anticorpos*, por sua vez, produtos da resposta imunológica, inativam o patógeno marcando-o para sua eliminação pelo organismo. Nessa visão, portanto, o patógeno "ataca" e o sistema imune "contra-ataca".

Essa abordagem não é consensual na comunidade dos pesquisadores em Imunologia, sendo que, inclusive, existem diversas opiniões e discussões sobre o funcionamento do sistema imune. Por exemplo, o título de um artigo relativamente recente de um conhecido pesquisador na área de imunidade inata tem por título "*How the immune system works to protect the host from infection: A personal view*" (JANEWAY JR., 2001). Além disso, uma das publicações mais conhecidas na área é a "Current Opinion in Immunology". Muitos processos imunológicos não são bem compreendidos, e existe muita discordância entre os imunologistas em muitos dos princípios básicos do funcionamento do sistema imune (ANDREWS; TIMMIS, 2005).

Entre outros aspectos, a visão do sistema imune como mecanismo de defesa do organismo centra-se na descrição do sistema sobre seus efeitos nas infecções e não sobre uma perspectiva sistêmica global a partir das relações entre seus componentes e desses componentes com o meio em que se insere, o próprio organismo. Dessa maneira, a visão de defesa do organismo é uma

abordagem pontual e restrita do sistema imune, apesar de ainda ser, em maior ou menor grau, predominante entre os pesquisadores da área. Essa visão tem se focado em estudar os componentes do sistema e seus efeitos, ao invés de uma abordagem mais global e funcional do sistema imune, incluindo a relação entre seus componentes. Como comentado em (VAZ; FARIA, 1993, p.1),

> Há um desconhecimento bastante difundido sobre os mecanismos de defesa imunológica. (...) Os mecanismos básicos de operação do sistema imune não são conhecidos, embora conheçamos minuciosamente a maioria de seus componentes e sub-componentes.

Apesar de possuir mais de uma década de existência, essa afirmação continua válida, mesmo com todo o amadurecimento do conhecimento biológico nesse mesmo período. Essa situação ocorre porque existe um conhecimento relativamente satisfatório sobre os componentes do sistema imune, mas pouco ainda sobre a interação entre seus elementos, e desses elementos com patógenos ou mesmo com o próprio organismo. Ainda destacando (VAZ; FARIA, 1993, p.1), também continua atual o trecho:

> Desde que não entendemos como o sistema opera, não sabemos como intervir previsivelmente na maioria das atividades imunológicas: as vacinas em uso humano e veterinário foram todas desenvolvidas empiricamente, milhares de outras falharam no estágio experimental. Não sabemos porque umas funcionam, outras não.
>
> A Imunologia não pode prometer vacinas à sociedade porque não sabe como fabricá-las. Os progressos recentemente obtidos sobre mecanismos celulares e moleculares não alteram este panorama.

Como pode ser deduzido a partir dessa afirmação, atualmente tem-se muito mais conhecimento sobre os componentes que sobre os mecanismos e formas de operação do sistema imune. Ao contrário de outras ciências biológicas, a Imunologia, ao invés de partir do estudo da estrutura e funcionamento de um sistema, surgiu como uma forma de validação de resultados obtidos pela Medicina, mais especificamente no final do século XIX (VAZ; FARIA, 1993). Isso levou ao estudo dos mecanismos de resistência a infecções, denominado imunidade.

No que se refere aos seus papéis anti-infecciosos, o sistema imune atua por meio de mecanismos inatos e adaptativos, sendo por isso descrito como sistema imune inato e sistema imune adaptativo. A imunidade inata é composta principalmente por barreiras epiteliais, fagócitos, mastócitos, células

NK (*Natural Killer*), células dendríticas, sistema do complemento, receptores celulares tipo Toll, etc. Em geral, funcionam ou como barreira mecânica à penetração do patógeno (caso da pele), ou por atuação de células que efetuam o reconhecimento de padrões associados a diversos tipos de patógenos (PAMP - *Pathogen-Associated Molecular Patterns*), eliminando ou iniciando o processo de eliminação desse patógeno. Esse reconhecimento é feito através dos PRR (*Pattern Recognition Receptors*), sendo os receptores tipo Toll (TLRs - *Toll Like Receptors*) os mais importantes. É importante ressaltar aqui que esses padrões são pré-definidos, no sentido em que o sistema inato não tem condições de se modificar para reconhecer padrões não-conservados.

A imunidade adaptativa, por sua vez, é caracterizada pela ativação, multiplicação e diferenciação dos linfócitos, células com especificidade restrita, mas com um repertório bastante amplo (CALICH; VAZ, 2001). Comparados aos fagócitos, por exemplo, os linfócitos apresentam, do ponto de vista imunológico, duas diferenças significativas. A primeira é que cada linfócito reconhece menos padrões que um fagócito, possuindo receptores que interagem com menos estruturas, sendo portanto muito mais específico. A segunda diferença é que essa alta especificidade do linfócito é compensada por seu amplo repertório, uma vez que existe uma grande variabilidade nos receptores antigênicos expressos, ao contrário dos fagócitos, nos quais os receptores não costumam variar de uma célula para outra.

Existem duas grandes classes de linfócitos: as células B e as células T. As células B expressam uma proteína globular em suas superfícies, o BCR (*B Cell Receptor* - Receptor da Célula B), também conhecidos como imunoglobulinas de membrana. Quando ativados, os linfócitos B diferenciam-se em plasmócitos, secretando essas imunoglobulinas que circulam como anticorpos no organismo. Dessa maneira, anticorpos são apenas BCRs secretados por células que, em sua fase inicial, expressaram essas imunoglobulinas sobre a sua superfície. Uma imunoglobulina possui um sítio de ligação, responsável pelo reconhecimento de antígenos específicos, como ilustrado na Figura 4.1.

A grande variabilidade das imunoglobulinas expressas, seja em membrana ou secretadas, ocorre devido a processos de recombinação e mutação de DNA durante a formação da célula B. Esses processos garantem ao organismo a existência de uma grande diversidade de BCRs, mas cada tipo expresso geralmente em poucas células. Em condições propícias, o processo de ligação ao antígeno, entretanto, estimula a célula B a se dividir, formando uma linhagem clonal expandida dessa linhagem. Durante esse processo de clonagem, também podem ser formadas células B de memória, capazes de sobreviver por um

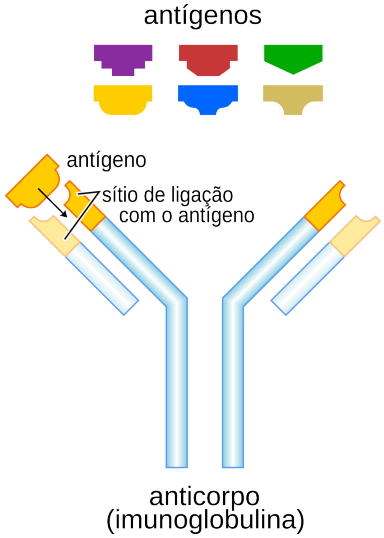


Figura 4.1: Ligação de Anticorpo a um Antígeno

longo tempo e possibilitar uma rápida resposta a uma segunda exposição do organismo ao mesmo antígeno.

A segunda grande classe de linfócitos é formada pelas células T, que quando ativadas, são geralmente encontrados sob a forma de células T citotóxicas, ou células T auxiliares. Os linfócitos T expressam o TCR (*T Cell Receptor* - Receptor da Célula T), que possuem algumas similaridades com os BCRs, como por exemplo o uso de recombinação de DNA durante o processo de formação da célula. Por outro lado, esse receptor possui características próprias, pois é adaptado para reconhecer antígenos apresentados por outras células do organismo, destacando-se as células dendríticas. Os linfócitos obviamente expressam outras moléculas de superfície, sendo que inclusive a presença das glicoproteínas CD4 e CD8 são utilizados para definir o tipo de célula T. Nesse caso, CD é abreviatura de *Cluster of Differentiation* ou alternativamente *Cluster of Designation*, um protocolo para identificação de células de superfície presentes em células brancas do sangue.

As células T citotóxicas apresentam o receptor CD8, sendo por isso também denominadas de CD8+, produzindo toxinas que atacam células tumorais ou infectadas, induzindo-as à apoptose, morte celular programada. Essa indução também pode ocorrer mediante a interação celular do linfócito T com a célula-alvo. As células T auxiliares, por sua vez, são CD4+ e, quando ativadas, dividem-se rapidamente e secretam citocinas, para regulação da resposta imune. Essa regulação pode se dar, por exemplo, pela ativação de outras células do sistema imune, como linfócitos B e macrófagos. É importante observar que com a maior parte dos antígenos, inclusive, as células B dependem da

ação de uma célula T auxiliar para poderem se ativar e diferenciarem-se em plasmócitos.

Além das células T citotóxicas e células T auxiliares, existem duas outras classes de linfócitos T de grande importância para o correto funcionamento do sistema imune: células T de memória e células T regulatórias. As células T de memória são definidas por características semelhantes às células B de memória: capacidade de sobreviver por um longo tempo e possibilidade de rápida resposta a novas exposições do antígeno ativador. As células T regulatórias, em geral CD4+, são também conhecidas como células T supressoras e auxiliam no processo de regulação do sistema imune, sendo essenciais para a manutenção a tolerância imunológica. Para que se tenha uma idéia da importância dessa classe, deficiências no funcionamento dessa classe de células em geral levam ao surgimento de doenças autoimunes[1].

Como comentado em (JANEWAY JR., 2001), a imunidade inata é anterior, evolutivamente, à imunidade adaptativa. Com raras excessões, todos os vertebrados possuem imunidade adaptativa baseada em linfócitos e anticorpos[2]. As células do sistema imune, inato ou adaptativo, são produzidas na medula óssea e passam por uma ou mais etapas de maturação, como ilustrado de forma simplificada na Figura 4.2. Cabe destacar que, no caso dos linfócitos T, a medula óssea é responsável pela produção de seus precursores celulares, sendo que o linfócito T completa sua formação no timo, onde passa a expressar todo um conjunto característico de receptores moleculares. O linfócito B, por sua vez, sai da medula óssea já apresentando os receptores característicos de sua classe.

Além disso, é importante observar que o sistema imune inato participa não apenas como um primeiro contato do patógeno com o organismo, mas também como elemento essencial para ativação e regulação do sistema imune adaptativo. Mais detalhes podem ser verificados em vários trabalhos de pesquisadores da imunidade inata, especialmente Ruslan Medzhitov e Charles Janeway Jr. Em especial, para melhor compreensão da imunidade inata, recomenda-se a leitura de (KOPP; MEDZHITOV, 2003), (MEDZHITOV, 2001), (MEDZHITOV; JANEWAY JR., 2000) e (JANEWAY JR.; MEDZHITOV, 2002). Uma visão evolucionária do sistema imune inato é apresentada em (MUSHEGIAN; MEDZHITOV,

[1]Em termos simplificados, pode-se dizer que uma doença autoimune é caracterizada por uma resposta do sistema imune contra o próprio organismo. Ou seja, o organismo ataca suas próprias células, através de um sistema imune desregulado

[2]Todos os organismos providos de maxilares, mesmo peixes cartilaginosos, possuem tecido linfóide organizado e receptores de célula T e imunoglobulinas (JANEWAY JR., 2001). Entretanto, em vertebrados sem maxilares, como a lampréia, não se observa um sistema imune adaptativo baseado em moléculas padrões do sistema imune adaptativo.

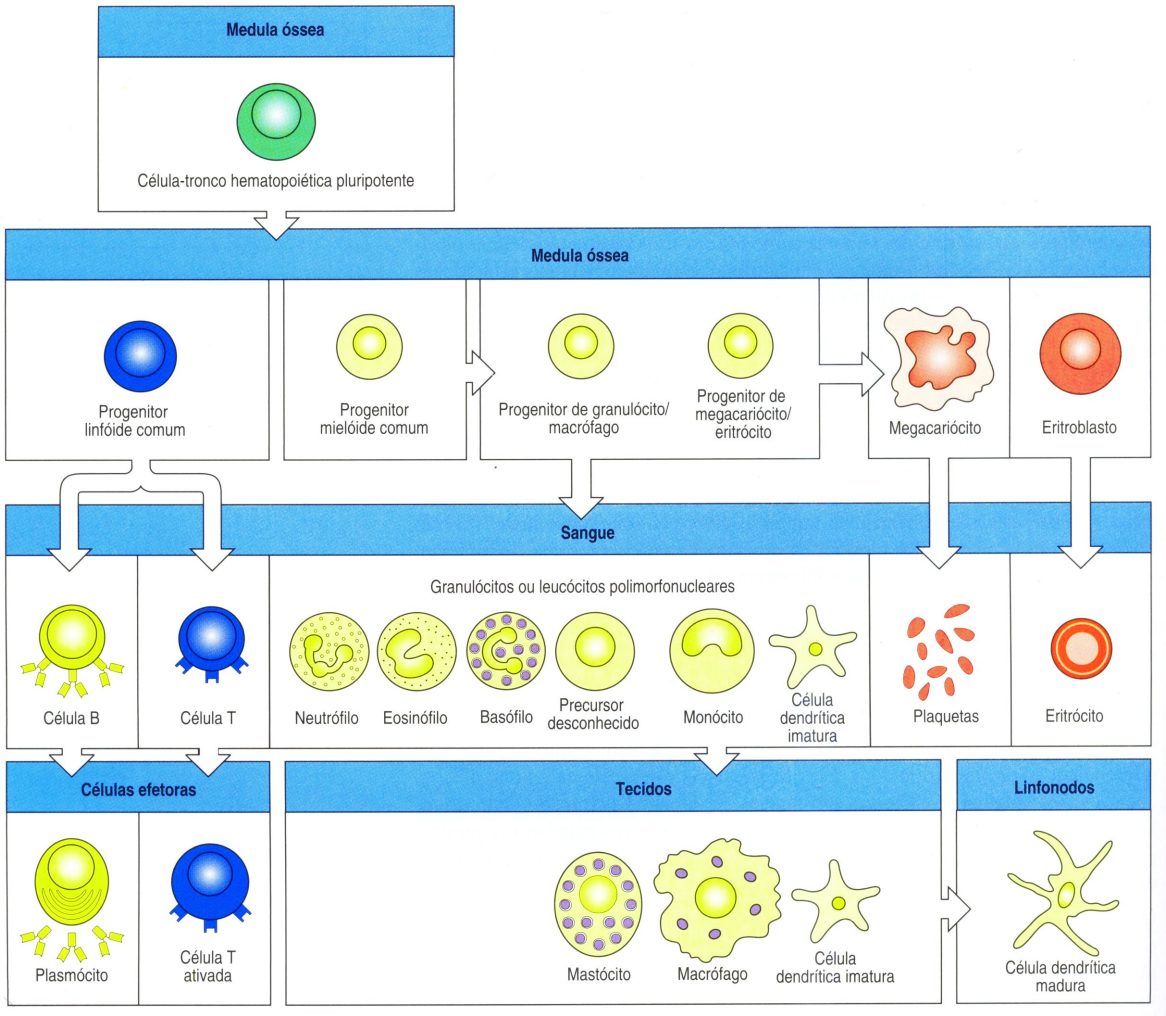


Figura 4.2: Geração e Maturação de Células do Sistema Imune (JANEWAY *et al.*, 2001)

2001). A regulação do sistema imune adaptativo pelo sistema imune inato, por sua vez, é abordada em (MEDZHITOV; JANEWAY JR., 1998), (JANEWAY JR., 2001), (BARTON; MEDZHITOV, 2002) e (PASARE; MEDZHITOV, 2004).

O sistema imune adaptativo, por outro lado, possui esse nome por poder interagir com epítopos, componentes dos antígenos que não se configuram como padrões moleculares (PAMPs) e para os quais não há receptores do sistema imune inato que os identificam. O sistema imune adaptativo é capaz de se adaptar ao "reconhecimento" de novas moléculas, que não são características conservadas de grupos de patógenos. A resposta do sistema imune adaptativo, entretanto, é muito mais lenta que a do sistema imune inato, o que aponta para uma dinâmica mais complexa no sistema adaptativo. Essa dinâmica ocorre principalmente mediante a ação de dois mecanismos:

Imunidade Humoral: formada por linfócitos B e plasmócitos. Os plasmócitos são resultados da maturação de linfócitos B e responsáveis pela produção de anticorpos.

Imunidade Mediada Por Célula: formada por linfócitos T, incluindo células T efetoras (linfócito T auxiliar CD4 ou linfócito T citotóxico[3] CD8) e linfócitos T regulatórios.

O processo de ativação das células maduras do sistema imune adaptativo implica em estimulação e a existência de um contexto ativador. De acordo com (RUSSO, 2001), as células do sistema imune podem ser estimuladas tanto por contato com outras células como por meio de citocinas. Entretanto, uma superestimulação ou mesmo a estimulação em um momento ou local não propício pode induzir, não uma ativação, mas uma inibição da célula imune ou até mesmo sua morte celular programada, a apoptose.

É possível inferir, a partir das observações anteriores, que a resposta imune é relativamente complexa, passando por todo um mecanismo de ativação celular e envolvendo diversos elementos do sistema imune. Essa resposta pode se manifestar de diferentes maneiras, seja pela ação de células atuando diretamente no patógeno, seja por meio indireto através da ação de citocinas ou anticorpos, por exemplo. A resposta imunológica envolve duas características extremamente importantes do sistema imune, que são a *especificidade* e a *memória*. Como comentado em (ABRAHAMSOHN, 2001, p.31),

> A especificidade é a capacidade de reconhecer e reagir a determinada molécula e a memória é a capacidade de voltar a reconhecer e reagir rapidamente a esta mesma molécula, quando esta for reintroduzida no organismo.

Como já comentado, a especificidade de cada linfócito é restrita e variável, porém o repertório reconhecido pelos linfócitos é amplo, uma vez que existem bilhões deles no sistema imune. Com os componentes do sistema imune inato, a situação é inversa: os receptores não variam de célula para célula em uma mesma família, e cada célula possui receptores para padrões moleculares que interagem com diversas estruturas. Assim, a especificidade das células do sistema imune inato é ampla, porém seu repertório é extremamente limitado, muito mais restrito que o dos linfócitos. Sobre o processo de reconhecimento pelo sistema imune adaptativo, detalha (ABRAHAMSOHN, 2001, p.31):

[3] O linfócito T citotóxico é também chamado de CTL, do inglês *Cytotoxic T Lymphocyte*.

> As moléculas que são reconhecidas pelo sistema imune são chamadas de *antígenos*. O reconhecimento de um antígeno ocorre pela interação química, não covalente, entre segmentos de molécula de antígeno, chamados de *determinantes antigênicos* ou *epítopos*, com regiões especializadas de glicoproteínas existentes na superfície dos linfócitos. Estas moléculas são chamadas de *receptores para antígenos* e são proteínas integrais de membrana.

Uma vez reconhecido o antígeno, que pode ser inclusive um componente do próprio organismo, é iniciado um processo de resposta imunológica, possível de se manifestar de diferentes maneiras. No caso de um linfócito B, por exemplo, satisfeitas as condições de ativação, ocorreria sua diferenciação em plasmócito e desencadeamento da produção e secreção de anticorpos. Esses anticorpos, por sua vez, podem, entre outras possibilidades, iniciar um processo de opsonização, "marcando" microorganismos para que eles sejam fagocitados. Nesse caso, tem-se uma resposta humoral do sistema imune.

A resposta imunológica poderia se dar também por meio de células T produzindo citocinas que induzirão células do sistema imune inato, como os macrófagos, para fagocitarem partículas que apresentarem o antígeno reconhecido. Esse processo constitui outro mecanismo da imunidade adaptativa, que é a imunidade mediada por células. Nesse caso, a ação se dá por meio de secreções moleculares solúveis (as *citocinas*) que atuam nas células da vizinhança e que possuam receptores para elas. Dependendo do tipo de célula T emissora, a citocina terá papel fundamental em vários processos, destacando-se:

- ativação de células B ou macrófagos, no caso de linfócitos T auxiliares CD4;
- indução da apoptose de células com antígenos reconhecidos pelos receptores de linfócitos T citotóxicos CD8;
- regulação da resposta imune na região, no caso de linfócitos T regulatórios.

O processo de ativação da imunidade mediada por células envolve o MHC (Complexo de Histocompatibilidade Principal – *Major Histocompatibility Complex*), receptor celular das células apresentadoras de antígenos (APCs), responsável pela apresentação de antígenos aos linfócitos T. Essa apresentação ocorre principalmente através do contato dos receptores de antígenos dos linfócitos T com o MHC de células dendríticas. Existem dois tipos de MHC: as

moléculas do MHC de classe I, portadoras de peptídios intracelulares, são reconhecidas por células T citotóxicas; já as moléculas de classe II, portadoras de antígenos extracelulares, são reconhecidas por células T auxiliares.

Obviamente, o estudo da resposta imune adaptativa é extremamente complexo e é onde pairam as maiores dúvidas e questões. Para uma exposição mais completa do sistema imune adaptativo, recomenda-se a leitura de (VAZ *et al.*, 2003), (JANEWAY JR., 2001), (JANEWAY *et al.*, 2001) e (CALICH; VAZ, 2001). Para uma melhor compreensão dos elementos do sistema imune, apresenta-se na Tabela 4.1, adaptada de (TWYCROSS; AICKELIN, 2005), uma comparação entre os sistemas imune adaptativo e inato.

Tabela 4.1: Comparação entre os sistemas imune inato e adaptativo

Propriedade	**Sistema Imune Inato**	**Sistema Imune Adaptativo**
Células	Células Dendríticas, Macrófagos, Mastócitos, Células NK	Células T, Células B
Receptores	Codificados pelo genoma, possuindo especificidade fixa, sem rearranjamento	Codificados em segmentos gênicos, com rearranjo somático
Distribuição dos Receptores	não-clonal	clonal
Reconhecimento	Padrões moleculares conservados, selecionados evolutivamente	Detalhes de estruturas moleculares, selecionados durante a vida do individuo
Resposta	Citocinas, Quimiocinas	Expansão clonal, anticorpos, citocinas
Tempo de Ação	Ativação imediata	Ativação lenta
Organismos	Vertebrados e invertebrados	Somente vertebrados

Percebe-se, a partir da Tabela 4.1, que o sistema imune inato constitui-se numa estrutura direcionada ao coletivo, à espécie (os padrões moleculares), enquanto o sistema imune adaptativo possui mais características direcionadas ao indivíduo (o antígeno). Dessa maneira, o sistema imune adaptativo é mais detalhista que o sistema imune inato, uma vez que o primeiro precisa ter um maior "conhecimento" do organismo que faz parte.

Nesse sentido, uma das questões predominantes no estudo do sistema imune, e de especial interesse neste trabalho, é a discriminação entre antígenos componentes do organismo (posto que este também é constituído de antígenos) e o patógeno (fonte de antígenos que ativam o sistema imune). Dito de outra forma, torna-se importante verificar como o sistema imune trata o que é próprio e o que não é próprio do organismo (*self*/*nonself*). Como comentado em (JANEWAY JR., 2001), o sistema imune é, em essência, auto-referencial: ele é selecionado e ativado por moléculas próprias. Essa última afirmação

contradiz a visão simplista de um sistema imune como um elemento de defesa do organismo, respondendo a elementos invasores. Não se pode esquecer que o sistema imune não possui vontade própria, ou mesmo intenção de defender o organismo. Como ressaltado em (VAZ *et al.*, 2003), para o sistema imune não existe o estranho e a percepção de *fora/dentro*, o que torna a discriminação de *próprio/não-próprio* pelo sistema um pseudo-problema. Essa discussão é retomada mais à frente, na Seção 4.4.

À parte dos debates sobre o funcionamento do sistema imune, muito progresso tem ocorrido recentemente na Imunologia. Entretanto, ainda continuam em aberto várias questões, por exemplo:

- como se dá a interação antígeno $\times$ anticorpo e anticorpo $\times$ anticorpo?
- como atuam as vacinas?
- como surgem as doenças do sistema imune (ex.: alergias, HIV, etc.)?
- como é a relação do sistema imune com um órgão transplantado?
- como o sistema imune "atua" na defesa do organismo?
- como o sistema imune "aprende"?

Essas questões não possuem ainda respostas satisfatórias, apesar do grande volume de pesquisa existente nos temas relacionados. Existe um relativo consenso que as respostas a essas indagações passam por uma análise da interação dos componentes do sistema imune entre si e com o organismo. Dessa forma, pode-se considerar bastante adequado o termo "Imunobiologia", que é preferido por alguns autores, e.g. (JANEWAY *et al.*, 2001).

Além disso, paralelo ao processo de resposta imune, ocorre a formação de uma memória imunológica, com diferenciação de linfócitos em *células de memória*. Essa memória será responsável por uma resposta mais rápida e eficiente ao mesmo antígeno quando de uma nova apresentação. Entretanto, como seria de se esperar, esse processo não é tão simples como se poderia pensar, o que será observado mais à frente, na Seção 4.3.

4.3 Rede e Memória Imunológica

DENTRO do contexto de interação, é importante ressaltar duas visões que objetivam propiciar melhor compreensão do funcionamento do sistema

imune: a compreensão do sistema imune como um sistema cognitivo e o conceito de rede idiotípica, uma rede imunológica formada pelos idiotipos dos anticorpos. O conceito de rede idiotípica foi proposto por Niel Kaj Jerne (JERNE, 1973), (JERNE, 1974). Essa teoria, como descrito em (JERNE, 1973) foi proposta tendo por base três dicotomias:

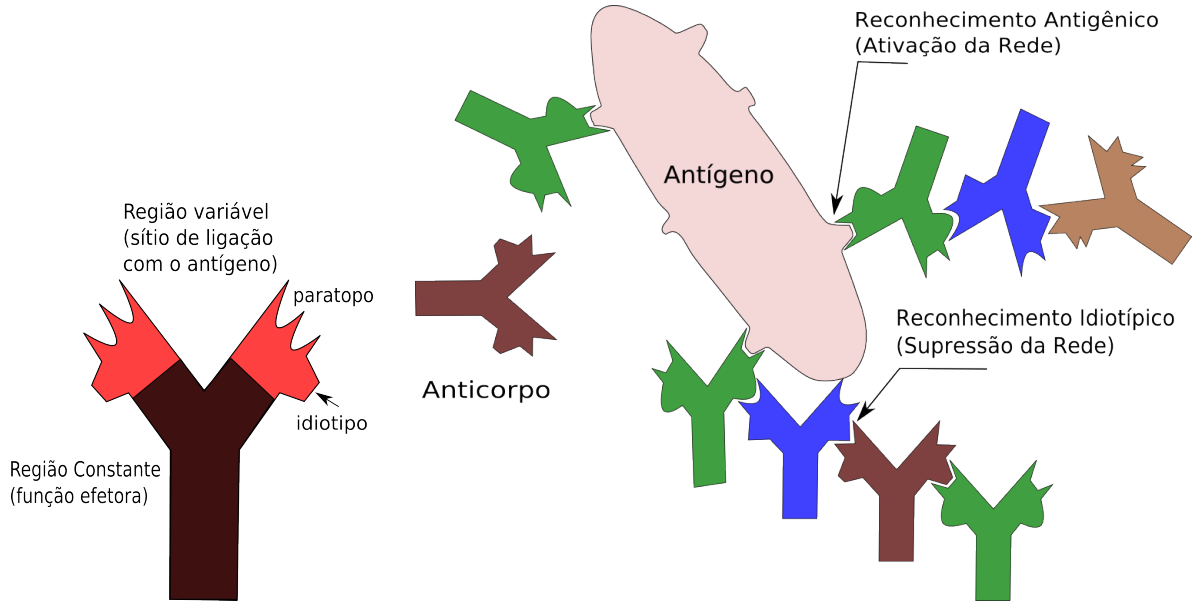
1. a existência de dois tipos de linfócitos, T e B, em parte sinérgicos, em parte antagônicos;
2. a dualidade (estimulação ou inibição) da resposta potencial de um linfócito quando seus receptores reconhecem um epítopo;
3. moléculas de anticorpo podem reconhecer anticorpos, mas também serem reconhecidas.

A visão do sistema imune como um sistema cognitivo é apresentada de forma detalhada em (VARELA *et al.*, 1988). De acordo com esse trabalho, os sistemas imune e neural são as mais poderosas formas de cognição biológica dentro do reino animal. Essa visão é em grande parte baseada na teoria de rede idiotípica. Em (JERNE, 1974), inclusive, Niel Jerne especulava que poderiam existir similaridades no conjunto de genes que governavam a expressão e regulação dos sistemas nervoso e imune.

Como comentado por de Castro e Timmis (2002c), a teoria da rede idiotípica propõe que o sistema imune tem um comportamento dinâmico, mesmo em ausência de estímulo externo. As células e moléculas do sistema imune fariam parte de uma rede regulada com capacidade para reconhecimento intrínseco. Assim, por exemplo, para cada anticorpo disponível, existiria no organismo um anticorpo com capacidade de reconhecê-lo. Dessa maneira, o funcionamento do sistema imune residiria na busca de um equilíbrio dinâmico.

O funcionamento da rede idiotípica fica melhor ilustrado com a Figura 4.3. Nessa Figura, encontra-se a representação de um anticorpo (Figura 4.3(a)), destacando-se nele duas regiões: i) o paratopo, localizado na região variável do anticorpo e que é responsável pela ligação ao epítopo do antígeno; ii) o idiotipo, um conjunto de epítopos do anticorpo e que pode ser reconhecido por outros anticorpos. Assim, na Figura 4.3(b), tem-se uma situação em que o reconhecimento idiotípico acaba por regular o funcionamento da rede como um todo. As bases da teoria da rede idiotípica foram lançadas em (JERNE, 1974), que estabeleceu que, para um dado sistema imune, “qualquer idiotipo

pode ser reconhecido por um conjunto de paratopos e qualquer paratopo pode reconhecer um conjunto de idiotipos".



(a) Anticorpo (b) Rede Idiotípica

Figura 4.3: Exemplo de Rede Idiotípica

Nesse sentido, como apontado por Jerne (1974), uma teoria completamente baseada em rede do sistema imune necessita que os resultados esperados das várias interações dentro da rede sejam definidos com bastante clareza e precisão. Para esse autor, o sistema imune pode ser visto como uma rede funcional com um comportamento auto-supressivo, mas aberta a estímulos externos, o que desencadearia a resposta imunológica.

É interessante observar que o conceito de rede idiotípica pode ser utilizado para explicar a formação da memória imunológica. Nesse caso, pressupõe-se que a memória mais que nos elementos reside nas interações entre eles. Essa já era uma questão levantada em (COUTINHO, 1989, p.22):

> É a memória uma propriedade da rede? Sem dúvida, a dinâmica da rede leva a uma memória de curta duração (...) e existe evidência que a rede 'relembra' pequenas modificações nas concentrações de componentes por até alguns meses.

De uma forma mais incisiva, em (STEWART; VARELA, 1989) é proposto a partir de resultados obtidos com simulação que a memória é uma propriedade de toda a rede imunológica e não pode ser atribuída a um único componente como uma célula de memória.

Nesse sentido, é importante destacar os trabalhos de Antonio Lanzavecchia, que recentemente tem investigado o processo de manutenção da memória serológica. Esse pesquisador possui diversos trabalhos ligados à questão da memória imune. Entre os trabalhos sobre células T de memória, pode-se destacar (SALLUSTO; LANZAVECCHIA, 2001), (LANZAVECCHIA; SALLUSTO, 2002), (LANZAVECCHIA; SALLUSTO, 2005) e principalmente (SALLUSTO; GEGINAT; LANZAVECCHIA, 2004). Em (LANZAVECCHIA; SALLUSTO, 2000) e (LANZAVECCHIA; SALLUSTO, 2001) são apresentados mecanismos de regulação de células T através da ação de células dendríticas.

Para este trabalho, entretanto, tem especial interesse três artigos desse pesquisador e sua equipe. Em (TRAGGIAI; PUZONE; LANZAVECCHIA, 2003), é discutido o processo de manutenção da memória serológica. Como comentado nesse trabalho, duas hipóteses foram levantadas na literatura sobre a manutenção do nível de anticorpos no sangue, mesmo após "cessado" o estímulo da resposta imune:

- a produção de anticorpos é mantida por plasmócitos de longa vida que sobrevivem em nichos apropriados (MANZ; RADBRUCH, 2002), (MANZ *et al.*, 2005);
- antígenos persistentes ou de reação cruzada estimulam continuamente células B a se proliferarem e diferenciarem em plasmócitos de vida curta (OCHSENBEIN *et al.*, 2000), (ZINKERNAGEL, 2002).

Entretanto, em (BERNASCONI; TRAGGIAI; LANZAVECCHIA, 2002) é proposto que mesmo na ausência de antígenos específicos, células B de memória se diferenciam e proliferam em resposta a estímulos policlonais derivados de micróbios ou células T ativas. Como mostrado em (BERNASCONI; ONAI; LANZAVECCHIA, 2003), os receptores do tipo Toll desempenham importante papel nesse processo. Ainda conforme (BERNASCONI; TRAGGIAI; LANZAVECCHIA, 2002), as células B de memória possuem dois modos de resposta:

- em um modo dependente de antígeno, um conjunto de células B de memória procede a uma expansão massiva e diferenciação em plasmócitos;
- em um modo policlonal, as células B de memória em um dado microambiente respondem a estímulos policlonais, procedendo a uma proliferação e diferenciação contínua, mantendo o nível de anticorpos relativamente estável.

Como verificado por esse pesquisador, em experimentos *in vitro*, foi possível verificar situações em que células B de memória respondiam aumentando a produção de anticorpos em até dez vezes, mesmo não sendo específico para aquele antígeno. Um resultado relativamente interessante, apontado por Bernasconi, Onai e Lanzavecchia (2003), é que existem evidências para um sinergismo entre receptores de células B e receptores do tipo Toll. Alguns antígenos contendo ativadores policlonais podem atuar como uma ponte física entre esses receptores.

Esses resultados vêm ao encontro da observação apresentada em (VAZ *et al.*, 2003) de que a imunopatologia surge mais a partir de desarranjos na conectividade dos linfócitos que de respostas imuno-específicas induzidas por antígenos. Mais à frente, nesse mesmo artigo, os autores comentam:

> Portanto, um sistema imune precisa ser definido não apenas baseado na presença de linfócitos, mas também de acordo com a maneira que mudanças ocorrem nas relações entre esses linfócitos e entre os linfócitos e o organismo enquanto o sistema opera.
>
> (VAZ *et al.*, 2003, p.14)

Tomando-se como referência, então, que o meio no qual o sistema imune opera é o organismo do qual ele é componente, a modelagem do comportamento do sistema imune implica em uma rede idiotípica não-trivial, com relações entre elementos dos sistemas imunes adaptativo e inato. Além disso, a modelagem de outros componentes do organismo pode ser necessária, dependendo do estudo que se pretenda fazer.

4.4 A questão "*Próprio × Não-Próprio''*

Em uma visão mais tradicional, como já comentado, o sistema imune é responsável por proteger o corpo de seus invasores, sem lesar o organismo no entanto. Nessa visão, cabe ao sistema imune a responsabilidade de discriminar o próprio (*self*), o organismo, e o não-próprio (*nonself*), o invasor, o estranho. Como o sistema imune em alguns momentos ataca células próprias e é tolerante para vários organismos, essa visão foi adaptada: o sistema imune teria responsabilidade discriminar entre algum próprio e algum não-próprio.

Para Ruslan Medzhitov e Charles Janeway Jr., o sistema imune inato desenvolveu várias estratégias para a discriminação "*próprio/não-próprio*". Como exposto em (MEDZHITOV; JANEWAY JR., 2002, p.298),

> O sistema imune inato dos animais vertebrados utiliza três estratégias de reconhecimento imune que podem ser descritas em termos de reconhecimento de "não-próprio microbial", "ausência de próprio" e "próprio induzido ou alterado".

De acordo com esses pesquisadores, a base para o reconhecimento microbiano reside na capacidade do organismo em reconhecer produtos conservados do metabolismo desses microorganismos, os PAMPs, permitindo a distinção entre o próprio não-infeccioso e o não-próprio infeccioso. O reconhecimento da ausência de próprio residiria na detecção de marcadores do próprio normal, especialmente o MHC. Por fim, o reconhecimento de próprio induzido ou alterado seria baseado na detecção de marcadores de próprio anormal, induzidos por infecção e transformação celular. Em síntese, esse modelo apóia-se no sistema imune inato para distinguir entre próprio e não-próprio.

Entretanto, como apontado por Matzinger, em (MATZINGER, 2002), esse modelo criou novos problemas ao resolver antigos, não conseguindo explicar, entre outros aspectos, porque vírus estimulam imunidade, porque transplantes são rejeitados e o que induz autominunidade. Assim, para resolver alguns desses problemas, o modelo original de Medzhitov e Janeway foi alterado em diversos pontos, mas ainda não consegue resolver algumas das questões fundamentais em Imunologia.

Por sua vez, Polly Matzinger propõe uma visão do sistema imune em uma outra camada: para essa pesquisadora, o sistema imune distinguiria não entre próprio e não-próprio, mas entre perigo e ausência de perigo. Esse modelo, denominado *Modelo do Perigo* (*Danger Model*), encontra-se descrito de forma sintética em (MATZINGER, 2001) e (MATZINGER, 2002) e de forma bastante abrangente em (MATZINGER, 1994). Como exposto em (MATZINGER, 1994), o sistema imune discriminaria entre algum próprio de algum não-próprio, existindo quatro classes de estruturas reconhecidas por ele:

Próprio visível: estruturas para as quais o sistema imune é tolerante;

Não-próprio visível: estruturas para as quais o sistema imune responde normalmente;

Próprio invisível: estruturas do organismo para as quais o sistema imune não é tolerante, mas para as quais ele normalmente não responde;

Estruturas não imunogênicas: que o sistema imune simplesmente ignora, como o silicone.

Em (MATZINGER, 1994), é feita ainda a distinção entre ativação e engatilhamento. A ativação ocorreria com células em repouso, induzindo uma mudança de estado, requerendo co-estimulação. O engatilhamento, por sua vez, seria a indução de células efetoras a dispararem seus grânulos contra determinados alvos, ou indução de células auxiliares para envio de sinais de ativação para células B e macrófagos. Apesar desses dois processos serem conseqüências de algum evento externo, existe uma diferença no processo de sinalização de células em repouso ou efetoras. Uma visão geral desse processo pode ser visualizada no algoritmo apresentado na Figura 4.4.

Como observado em (MATZINGER, 1994), as regras apresentadas na Figura 4.4 são uma grande simplificação do processo, mas podem ser usadas para descrever um sistema imune. Recomenda-se a leitura desse texto para maior detalhamento do algoritmo apresentado. Cabe destacar que os modelos mais recentes utilizam o princípio de tolerância, ao invés de morte, para o caso em que células recebam o sinal 1, antigênico, na ausência do sinal 2, co-estimulatório. É importante observar aqui que não foi possível verificar, ainda, na literatura a existência de implementações baseadas no algoritmo apresentado na Figura 4.4, que foi utilizado por Matzinger para explanação de sua teoria. Isso ocorre apesar de existirem diversas aplicações computacionais baseadas no Modelo do Perigo, o que será visto no Capítulo 5.

Como apontado em (GREENSMITH; AICKELIN; CAYZER, 2005), o Modelo do Perigo sustenta que o funcionamento do sistema imune é guiado pela detecção de sinais endógenos, ao invés de exógenos. Os sinais endógenos surgiriam como resultado de dano ou *stress* nos tecidos celulares, e as células dendríticas possuem um papel de destaque nesse modelo, por sua capacidade de combinar sinais endógenos e exógenos. Isso pode ser verificado claramente na Figura 4.5, extraída de (MATZINGER, 2001).

Pode ser inferido da Figura 4.5, que um elemento central no Modelo do Perigo é a morte celular, o que é apontado em (GREENSMITH; AICKELIN; CAYZER, 2005) e outros textos sobre essa teoria. A morte pré-programada da célula, *apoptose* é parte do ciclo vital da célula e, sem ela, não seria possível o controle do crescimento do corpo. No início da apoptose, todo o material do núcleo é fragmentado de forma ordenada, enzimas digestivas são secretadas

```
I. Regras para linfócitos
   A. Regras para timócitos
      1. Morra se receber o sinal um (o recebido pelo receptor
         linfocitário) na ausência do sinal dois.
      2. Seja incapaz de receber sinal dois de qualquer fonte.
   B. Regras para células T virgens
      1. Morra se receber o sinal um na ausência do sinal dois.
      2. Responda para segundos sinais apenas se forem disparados por
         APCs profissionais (e.g.: células dendríticas).
      3. Circule de nódulo linfático a nódulo linfático ou para o
         baço.
   C. Regras para células T maduras
      1. Morra se receber o sinal um na ausência do sinal dois.
      2. Seja capaz de responder a segundos sinais disparados por
         células B e macrófagos, além de células dendríticas.
      3. Circule nos tecidos, bem como de nódulo linfático a
         nódulo linfático.
   D. Regras para células B
      1. Morra se receber o sinal um na ausência do sinal dois.
      2. Receba sinal dois apenas de células T experientes ou
         efetoras.
   E. Regras para células T ou B efetoras
      1. Ignore sinal dois. Execute sua função quando receber sinal um,
         independente da ausência ou presença do sinal dois.
      2. Morra ou reverta para um estado de repouso após um período
         relativamente razoável de tempo.
II. Regras para APCs
   A. Regras para APCs profissionais
      1. Torne-se ativada na presença de destruição de tecidos.
      2. Expresse sinais co-estimulatórios que podem ser recebidos
         por células T auxiliares virgens ou experientes.
      3. Regule sinais co-estimulatórios para precursores de CTL,
         ao receber sinais adequados advindos de células T auxiliares.
      4. Capture antígenos aleatoriamente do ambiente e apresente-os
         para células T.
   B. Regras para células B
      1. Expresse sinais co-estimulatórios que podem ser recebidos
         por células T maduras, mas não por células T virgens.
      2. Capture antígenos de seu ambiente, não aleatoriamente,
         ligando-os a seu receptor, concentre-os e apresente-os
         para células T.
```

Figura 4.4: Algoritmo de Funcionamento do Sistema Imune, de acordo com o Modelo do Perigo – Fonte: (MATZINGER, 1994)

internamente e novas moléculas são expressas na superfície celular. Dessa maneira, a célula é ingerida por macrófagos, com a membrana ainda intacta. Acredita-se que as citocinas resultantes da apoptose sejam anti-inflamatórias.

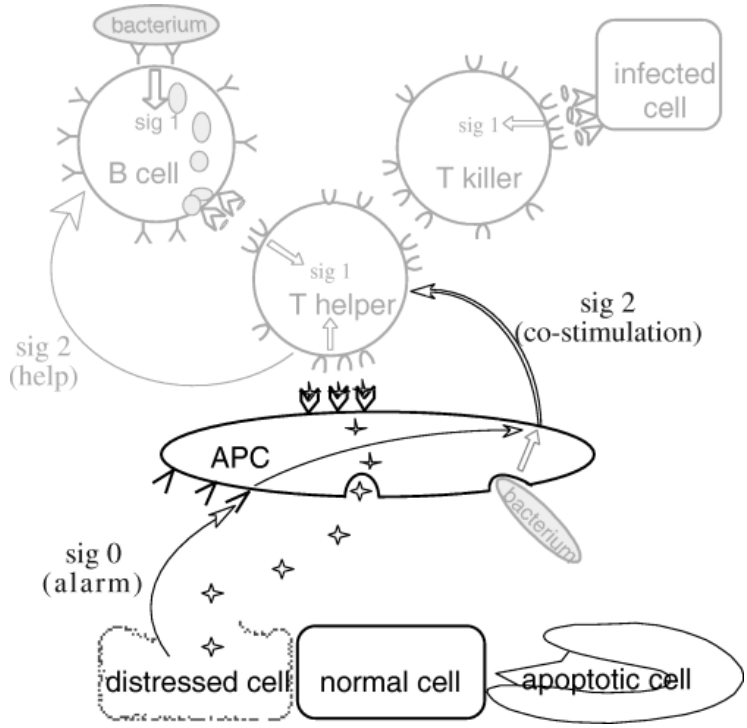


Figura 4.5: Sinais no Modelo do Perigo – Fonte: (MATZINGER, 2001)

Por outro lado, a *necrose* consiste na morte celular induzida por *stress* (infecção, dano, radiação, etc.). Nesse caso, não existe um cuidado no empacotamento dos conteúdos celulares internos ou preservação da membrana. A célula acaba perdendo a integridade da membrana e explode, liberando seu conteúdo, incluindo citocinas pró-inflamatórias. De acordo com o Modelo do Perigo, a diferença entre essas duas formas de morte celular são cruciais para destacar se um elemento exógeno deve ou não ser abordado pelo sistema imune, mais que o reconhecimento de PAMPs.

Partindo-se de uma visão mais abrangente do sistema imune, é possível destacar inúmeros méritos no Modelo do Perigo. Ele retira o foco dos elementos exógenos ao organismo, dando ao próprio organismo a responsabilidade por seu controle e ativação. Entretanto, ainda continua, mesmo que disfarçada, a visão do sistema imune como um mecanismo de defesa: se agora não contra patógenos, mas contra o perigo, predominando uma visão maquineísta de seu funcionamento. Além disso, como apontado em (AICKELIN; DASGUPTA, 2006), a teoria não é completa, e existem dúvidas sobre seu comportamento e estrutura. Entre outros pontos, a natureza de um sinal de perigo não é clara, e a teoria compartilha problemas comuns a modelos baseados em discriminação "*próprio* × *não-próprio*".

Partindo de uma visão mais abrangente, é possível apontar que no sistema imune predomina mais o diálogo entre seus componentes que a discriminação "*próprio* × *não-próprio*" ou entre "*perigo* × *ausência de perigo*". Como argumentos para essa proposição, o imunologista Irun R. Cohen apresenta um conjunto de observações em (COHEN, 2000):

1. É alto o grau de envolvimento do sistema imune no processo de manutenção do corpo (reparo de tecidos, regeneração de células, etc.), que depende em muito de processos inflamatórios regulados.

2. Células T autoimunes contribuem para a manutenção do corpo e possuem papel importante na manutenção do sistema nervoso central.

3. Autoimunidade pode ser ativada espontaneamente por tecidos danificados.

4. Indivíduos saudáveis possuem naturalmente auto-anticorpos e células T autoimunes.

5. Autoimunidade natural para um antígeno próprio pode regular respostas imunes a antígenos não-próprios que possuam alguma ligação com o antígeno próprio.

6. O mecanismo de autoimunidade é, em si, auto-regulado, para prevenir doenças autoimunes.

7. Antígenos tumorais são, em sua maior parte, antígenos próprios; e imunidade tumoral é em sua maior parte autoimunidade.

8. Doenças autoimunes freqüentemente envolve a ativação desregulada da autoimunidade natural.

Como argumentos adicionais, Cohen destaca que os antígenos em si são apenas moléculas ou fragmentos de moléculas que são reconhecidos pelos receptores de antígenos. O sistema, portanto, discrimina com base em antígenos, mas não com base em uma dicotomia próprio/não-próprio. Assim, o sistema imune não seria responsável por emitir vereditos sobre o que é ou não para ser destruído, mas por manter uma regulada inflamação corretiva a partir de um contínuo reconhecimento do estado dos tecidos:

> A tarefa do sistema imune é anexar o tipo de resposta inflamatória para as necessidades do momento: é necessário um novo tecido conectivo aqui?

> são necessários novos vasos sanguíneos ali? Um osso fraturado precisa de reparo? Como poderíamos melhor livrar o corpo de células velhas, anormais ou infectadas? O problema não é somente produzir a resposta apropriada, mas orquestrar o espectro de respostas dinamicamente no tempo de acordo com as necessidades mutáveis dos tecidos. Inflamação mantém o bem-estar do corpo e autoimunidade ajuda a fazer os ajustes finos do processo inflamatório.
>
> (COHEN, 2000, p.216)

Nessa visão a defesa contra patógenos é compreendida apenas como um caso especial de manutenção do corpo. Dessa maneira, dicotomias simples, como "*próprio*×*não-próprio*" ou "*perigo*×*não-perigo*", ocultam a profundidade da manutenção imune. Para manter a homeostase do corpo, o sistema imune precisa adotar diferentes soluções para diferentes problemas, selecionando e regulando um repertório de respostas inflamatórias. Mais ainda: o sistema imune é um sistema complicado porque esse processo regulatório requer um julgamento também complicado.

4.5 Imunoinformática

RECENTEMENTE tem-se assistido a um grande avanço da área de Bioinformática. Cada vez mais, surgem grupos e eventos nacionais e internacionais ligados ao assunto. Como comentado em (KITANO, 2002), a Bioinformática possui dois ramos distintos: 1) descoberta de conhecimento e mineração de dados em dados experimentais e 2) análise baseada em simulação, que testa hipóteses biológicas em ambientes de silício. Como comentado em (NUSSENZVEIG, 2003b), aos *in vivo* e *in vitro* dos experimentos em Biologia, acrescenta-se o *in silico*.

Esse potencial não passou despercebido na Imunologia. O crescimento explosivo em biotecnologia combinado a avanços fundamentais na tecnologia da informação tem potencial para radicalmente transformar a Imunologia (PETROVSKY; BRUSIC, 2002). Isso fomentou a recente criação da Imunoinformática, uma subárea da Bioinformática, focada na aplicação de técnicas computacionais em problemas associados à resposta imunológica. Entre as áreas da Imunoinformática, uma que tem recebido bastante atenção são os bancos de dados especializados em dados imunológicos. Pode-se citar, por exemplo:

- HIV Immunology Database (`http://hiv-web.lanl.gov/`);
- International ImMunoGeneTics (IMGT – `http://imgt.cines.fr/`);
- Swine Major Histocompatibility Complex (`http://sdmc.krdl.org.sg/`);
- MHCPEP (`http://wehih.wehi.eud.au/mhcpep/`).

Entretanto, persistem ainda várias questões sobre os bancos de dados imunológicos, entre eles a falta de padronização para armazenamento dos dados. Como observado em (PETROVSKY; BRUSIC, 2002), um dos objetivos do IMGT é contribuir para que isso ocorra. Existem também questões sobre a interpretação dos dados e mesmo sobre sua qualidade; nesse sentido, o IMGT e alguns outros são anotados.

Outra pendência na área de bancos de dados imunológicos é a escassez de ferramentas que facilitem a extração eficiente de dados e geração de conhecimento, com sério potencial para uso de técnicas de Mineração de Dados. Como apontado em (PETROVSKY; SILVA; BRUSIC, 2003), a habilidade para extrair e analisar informação útil dos variados bancos de dados em expansão é crucial não só para a pesquisa imunológica como para a prática clínica.

Ainda nas áreas da Imunoinformática, outro campo sobre o qual tem surgido bastante interesse recentemente é o estudo da apresentação de antígenos e desenvolvimento de vacinas. Em geral, as abordagens para esse tipo de pesquisa utilizam diversas técnicas, destacando-se:

Protein docking: utilizando-se de técnicas de otimização computacional e/ou modelagem tridimensional, busca-se modelar o acoplamento de moléculas do antígeno aos receptores de células T ou B.

Análise estatística: a partir de dados existentes sobre sistemas experimentais conhecidos, busca-se deduzir resultados para novos antígenos ou propostas de vacinas.

Modelos baseados em aprendizado de máquina: a partir de um grande repositório de dados já existentes, são desenvolvidas ferramentas que inferem propriedades e relações a partir desses dados, possibilitando a utilização desse conhecimento para indução de novas vacinas, por exemplo. Resultados significativos de aplicações de redes neurais artificiais na análise de rejeição de transplantes são apresentados em (PETROVSKY; SILVA; BRUSIC, 2003).

Além disso, também pode-se destacar o uso de Imunoinformática para estudo do comportamento do sistema imune, ou parte dele, em determinados contextos. Inúmeros trabalhos, utilizando-se as mais diversas técnicas, são propostos para estudo de infecções, autoimunidade, alergias, etc. Uma visão geral sobre essas possibilidades pode ser encontrada em (PETROVSKY; BRUSIC, 2002), (CHAKRABORTY; DUSTIN; SHAWN, 2003), (BRUSIC; PETROVSKY, 2003) e (PETROVSKY; SILVA; BRUSIC, 2003). Interessados em uma abordagem envolvendo genoma ou mapeamento e ligação de epítopos no MHC podem obter uma visão abrangente em (GROOT *et al.*, 2002). Uma publicação relativamente recente que merece bastante destaque é (BOCK; GOODE, 2003).

Uma outra visão do assunto, apresentada em (TARAKANOV; SKORMIN; SOKOLOVA, 2003) busca propor a criação da Imunocomputação, utilizando princípios de processamento de informação de proteínas e redes imunológicas para resolver problemas complexos. Nesse caso, a Imunocomputação levaria a um novo tipo de computador, o imunocomputador, com um *framework* matemático próprio, bem com o um novo tipo de *software* e *hardware*. O modelo proposto possui semelhanças em seus objetivos com aqueles estabelecidos pela Neurocomputação, que é o de utilizar inspiração biológica para a criação de novos mecanismos e modelos computacionais.

4.6 Sistemas Complexos

VÁRIOS trabalhos recentes de simulação e modelagem em Biologia tem-se apoiado sob fundamentos de Sistemas Complexos. Assim, esta seção tem o objetivo de apresentar uma visão geral dessa área, sem entrar em detalhes. Para uma visão mais abrangente, recomenda-se a leitura de (NUSSENZVEIG, 2003a), uma coletânea de vários artigos na área, a maior parte com enfoque biológico ou inspirado pela Biologia.

Entende-se um *sistema complexo* como um conjunto relativamente grande de elementos simples interagindo para produzir comportamento não-previsível. Por causa dessa interação, um sistema complexo apresenta as seguintes propriedades:

1. O todo é mais complexo que a soma das partes, apresentando características não dedutíveis a partir dos elementos isoladamente. Além disso, os elementos podem apresentar comportamento contraditório, inibindo ou estimulando outros elementos. Nesse caso, o sistema como um todo

tende a adotar "soluções de compromisso", que objetivem otimizar o máximo possível as restrições impostas pelos elementos.

2. O sistema incorpora um certo grau de aleatoriedade, dado que algumas de suas características são distribuídas ao acaso. Entretanto, o sistema adquire um grau de organização de forma espontânea, podendo apresentar atratores múltiplos, que são estados para os quais muitas configurações iniciais tendem a convergir após um tempo suficientemente longo. Em geral, o comportamento do todo expressa claramente não-linearidade nas relações entre seus elementos.

3. Como apontado em (OTTINO, 2004), em geral não existem leis que governam sistemas complexos, mas sim relações, inclusive as guiadas por uma lei de potência.

4. O sistema é dinâmico, em constante processo de desenvolvimento/adaptação como resposta das interações entre os elementos que o constituem e ao meio em que se insere. Nesse sentido, diz-se que o sistema é *adaptativo*, pois modifica suas características a partir da experiência e do funcionamento interno. Talvez por ser a mais importante das propriedades, também utiliza-se na literatura a expressão "sistemas complexos adaptativos".

Sistemas complexos podem ser identificados pelo que fazem e como podem ser analisados, exibindo organização sem uma autoridade organizadora central. E, em geral, decompor e analisar os elementos do sistema não fornece pistas para o comportamento do todo. Ecossistemas e a internet são exemplos reais de sistemas complexos. Exemplos clássicos de aplicações computacionais de sistemas complexos são o famoso "Jogo da Vida", uma aplicação clássica de autômatos celulares, discutida sob o ponto de vista da cognição em (BEER, 2004). Como comentado por Oliveira (2003), esse jogo consiste em um *software* que simula a competição pela sobrevivência numa população e reproduz muitos dos fenômenos observados naturalmente. Um autômato celular, por sua vez, pode ser definido como uma quádrupla (L,S,N,f), onde:

L**:** uma malha regular de células, os elementos do autômato celular. Em geral, adota-se um reticulado em uma dada dimensão 'd'.

S**:** um conjunto finito de estados do autômato celular, construído a partir dos estados individuais de cada célula e sua vizinhança.

N: um conjunto de índices que especifica uma regra de vizinhança para as células do autômato.

f: $S_n \rightarrow S_{n+1}$, uma função de transição entre os estados do autômato. Essa função irá atualizar cada célula do autômato, de acordo com o estado anterior da célula e o estado das células em sua vizinhança.

Mais detalhes e exemplos de aplicações de autômatos celulares podem ser encontrados em (OLIVEIRA, 2003), (BARBOSA, 2004) e (FERRUGEM *et al.*, 2003) entre outros vários textos que abordam o tema.

Como pode ser inferido a partir dos comentários anteriores, diversas são similaridades entre o sistema imune e sistemas complexos. Existem, diversos trabalhos que adotam essa abordagem na modelagem e simulação de processos imunológicos. Vários desses trabalhos utilizam, inclusive, autômatos celulares como base para a modelagem, o que será discutido na Seção 4.7 a seguir.

4.7 Modelagem e Simulação do Sistema Imune

Como apontado em (CHAKRABORTY; DUSTIN; SHAWN, 2003), a criação de modelos tem sido essencial em muitos avanços da Biologia. É destacado nesse artigo que o uso de modelos computacionais como complementos essenciais aos experimentos *in vivo* e *in vitro* não é um novo paradigma, mas a continuação de uma tradição iniciada por Burnet, Cohn e outros. Nessa visão, a Imunologia tem muito a ganhar com a união de dois grandes avanços das últimas cinco décadas: Biologia Celular e Molecular e simulação computacional de Sistemas Complexos. Obviamente esse é um grande desafio, como já apontado em (PETROVSKY; SILVA; BRUSIC, 2003, p.24):

> Dificuldades na aplicação de modelos computacionais surgem do fato que a maioria dos pesquisadores em Imunologia e especialmente clínicos têm apenas uma compreensão limitada de análise sofisticada de dados e sua aplicabilidade e limitações, enquanto a maior parte dos cientistas da computação carecem de compreensão da profundidade e complexidade de dados imunológicos.

Em (HOOD; PERLMUTTER, 2004), isso também é observado: "*a integração de disciplinas, de Matemática à Biologia Molecular é um dos aspectos mais*

desafiantes da abordagem de sistemas". Por Biologia sistêmica, os autores desse artigo entendem como uma abordagem analítica para as relações entre elementos de um sistema biológico, com o objetivo de compreender suas propriedades emergentes. Essa definição vem ao encontro do uso de sistemas complexos para compreensão de processos biológicos.

Várias são as abordagens encontradas na literatura para a modelagem do sistema imune, o que pode ser verificado em (SEGEL; COHEN, 2001) e (BOCK; GOODE, 2003). Uma grande parte dos trabalhos, se não a maioria, utiliza-se de equações diferenciais nessa tarefa. Nesses modelos, populações de antígenos e elementos do sistema imune são representados como variáveis contínuas em conjuntos de equações diferenciais ordinárias. Essa tem sido a abordagem preferencial de Antonio Coutinho, um dos grandes pesquisadores atuais em rede imune, e vários outros pesquisadores que atuaram com ele, como Francisco Varela, John Stewart e Jorge Carneiro.

Entre os trabalhos desses pesquisadores, utilizando essa abordagem, merecem destaque (CARNEIRO *et al.*, 1996a) e (CARNEIRO *et al.*, 1996b), onde é apresentado um modelo de rede imunológica com interação entre células B e T. Mas o esforço desses pesquisadores nessa tarefa vem desde a década de 80, o que pode ser confirmado, por exemplo, em (STEWART; VARELA, 1989). Obviamente esse tipo de modelagem não é exclusivo desse grupo. Por exemplo, um estudo apresentando o uso dessa abordagem para modelagem de migração linfocitária periférica é apresentada em (SRIKUSALANUKUL; BRUYNE; MCCULLAGH, 2000)

Entretanto, como comentado em (LOUZON *et al.*, 2003), apesar que sistemas biológicos dinâmicos são representados tradicionalmente utilizando equações diferenciais parciais, esse tipo de modelo é inadequado em muitas situações. Como comentado nesse texto, equações diferenciais parciais são úteis em sistemas que possuam as seguintes características:

Um número grande de partículas que interagem entre si: Apesar que o sistema imune possui grande número de células e moléculas, o número de interações pode ser relativamente pequeno, especialmente em estágios iniciais de infecção.

Uma taxa de difusão alta o suficiente para amortecer flutuações locais: A taxa de difusão pode ser baixa em vários casos e a taxa de difusão pode ser limitada fortemente por compartimentalização.

Um resultado imediato de cada interação, sem atraso para o resultado: A maioria dos processos biológicos requer a produção de novas moléculas ou divisão celular, que podem tomar horas ou mesmo dias. Portanto, existe sim atraso entre uma ação e seu resultado.

Além dessas questões, abordagens baseadas em equações diferenciais geralmente são mais adequadas a modelar o comportamento, ao invés do funcionamento interno dos sistemas. Com isso, corre-se mais riscos de incorporar subjetividade aos modelos, uma vez que são baseados na visão que o pesquisador tem sobre o processo. Em (KLEINSTEIN; SEIDEN, 2000) são apresentadas outras críticas ao uso de equações diferenciais, entre elas o fato de que em geral representam apenas o comportamento médio do sistema e envolvem não-linearidades o que torna sua solução analítica uma tarefa relativamente difícil.

Outro ponto a ser observado ainda é que esses modelos tendem, grosso modo, a funcionar como "caixas pretas", uma vez que dificilmente é possível destacar sua lógica interna. Na humilde opinião deste pesquisador, matemático por formação, essa situação acaba por dificultar a aceitabilidade desses modelos mais amplamente. Dizendo em uma forma hiperbólica, uma abordagem do sistema imune como uma calculadora avançada tende, em geral, a ser vista com desconfiança por muitos teóricos e pesquisadores do sistema imune. Isso também é apontado em (PETROVSKY; SILVA; BRUSIC, 2003, p.24):

> Não será fácil para imunologistas aceitar os resultados de caixas pretas tais como redes neurais artificiais ou outros sistemas de predição ou modelagem computacional (...) A única alternativa à frente portanto é desenvolver *frameworks* apropriados em Imunoinformática

O desenvolvimento desse *framework* passa obviamente por desenvolvimento de ferramentas que permitam aos imunologistas descrever processos biológicos, bem como compreender com facilidade o resultado de uma dada simulação. Esforços nesse sentido não são apenas importantes, mas também necessários em um contexto em que cada vez mais a Biologia se utiliza de métodos computacionais.

Obviamente, os esforços dos pesquisadores que utilizam a metodologia de equações diferenciais ou similares são extremamente importantes e possuem impacto significativo na comunidade científica. Os trabalhos de Coutinho, Varela e Stewart, por exemplo, são mais conhecidos pelas proposições teóricas sobre o sistema imune que por suas simulações *in silico*. Além disso, os

trabalhos desses pesquisadores buscam modelar principalmente a atividade intrínseca do sistema imune, independente do processo ativatório, o que não é de forma alguma simples de ser modelado por outros mecanismos.

Uma outra abordagem que tem crescido bastante nos últimos anos é a utilização de abordagens baseadas em autômatos celulares, inspiradas na visão do funcionamento sistema imune como um sistema complexo. Nesse sentido, merecem destaque os trabalhos desenvolvidos por Philip Seiden e Franco Celada, que apresentaram resultados com esse tipo de modelagem desde 1992 (SEIDEN; CELADA, 1992; CELADA; SEIDEN, 1992). Eles desenvolveram uma aplicação denominada IMMSIM[4].

Uma apresentação geral do IMMSIM, sua filosofia e resultados obtidos com ele podem ser verificados em (KLEINSTEIN; SEIDEN, 2000). Entre os trabalhos que utilizam esta ferramenta, podem ser citados (BERNASCHI; SUCCI, 2000) e (KOHLER *et al.*, 2001). Nesse último, inclusive, os autores propõem uma abordagem para o estudo do processo de vacinação e apontam que as respostas humorais e celulares do sistema imune, apesar de cooperarem para a cura do organismo infectado, mostram comportamento competitivo entre si.

Uma versão paralela do IMMSIM, utilizando-se de mecanismo de passagem de mensagens (MPI[5]) foi desenvolvida por Filippo Castiglione e encontra-se apresentada em (BERNASCHI; CASTIGLIONE; SUCCI., 1999) e (BERNASCHI; CASTIGLIONE, 2001). Essa versão, denominada PARIMM[6], foi utilizada, entre outros, na modelagem de vacina preventiva ao câncer (PAPPALARDO *et al.*,) e da resposta imune a tumores (CASTIGLIONE *et al.*, 2005).

No Brasil, a abordagem de autômatos celulares é utilizada principalmente por Américo Bernardes e Rita Zorzenon. Entre os trabalhos desses dois pesquisadores destacam-se (BERNARDES; SANTOS, 1997), (SANTOS; BERNARDES, 1998), (SANTOS; COPELLI, 2003) e (COPELLI; SANTOS; STARIOLO, 2003). Em geral, esses dois pesquisadores trabalham sob o ponto de vista de sistemas dinâmicos ou caóticos, utilizando-se de técnicas da geometria fractal, investigando o sistema imune sob a ótica de análise de pertubações.

Ainda com esse foco, um trabalho bastante interessante sobre Sistemas Imunes Artificiais é (GARRETT, 2003) que compara e discute diversos modelos baseados na rede imune sob o ponto de vista se um paratopo é ou não considerado um epítopo. Essa discussão é feita analisando-se as implicações

[4]IMMSIM: `http://www.cs.princeton.edu/immsim/software.html`.

[5]MPI: `http://www-unix.mcs.anl.gov/mpi/`.

[6]No *site* do pesquisador Filippo Castiglione (`http://www.iac.rm.cnr.it/~filippo/`) esse aplicativo é agora chamado de C-IMMSIM, por ser uma versão em C do IMMSIM.

no processo de seleção clonal e o autor, inclusive, estende essa discussão para alguns trabalhos de modelagem do sistema imune. Ao término do artigo, é proposto um autômato celular denominado GIN (*Generic Immune Network*) que tenta avançar de modelos mais simples de redes imunes na literatura. Entretanto, apesar do autor apresentá-lo como uma ferramenta para Sistemas Imunes Artificiais e modelagem do sistema imune, não foi mostrada nenhuma aplicação dessa abordagem.

Abordagens baseadas em autômatos celulares avançam na descrição dos elementos e relações do sistema imune, quando comparados aos modelos baseados em equações diferenciais. Entretanto, autômatos celulares possuem várias limitações, entre elas a ausência de movimento de seus elementos. Dessa maneira, as abordagens baseadas em autômatos acabam utilizando-se de mecanismos para simular essa movimentação. Em boa parte dos trabalhos, como os de Rita Zorzenon e Américo Bernardes, por exemplo, as células do autômato celular representam órgãos ou parte de órgãos do sistema imune.

Além disso, em outra parte significativa dos trabalhos, o funcionamento interno dos elementos do autômato é representado por equações diferenciais. Por exemplo, em (FURLAN; GAGLIARDI; ALVES, 2004) é proposto um estudo de memória imunológica baseado em um modelo de competição e co-evolução de parasitas e o sistema imune de um hospedeiro. É utilizado, para essa tarefa, um autômato celular probabilístico que utiliza equações diferenciais para modelar o processo de aquisição da memória imune e uma rede regulatória imune. De qualquer maneira, autômatos celulares têm apontado para direções promissoras no processo de modelagem do sistema imune e é uma área com bastante potencial de contribuição.

Uma proposta mais recente para a modelagem do sistema imune é usar uma abordagem baseada em sistemas de informação, o que é feito especialmente pela equipe de Stephanie Forrest. Nessa abordagem, o sistema imune é visto principalmente como um processador de informações, utilizando-se da metáfora de um computador biológico. Essa abordagem possui, como pode ser facilmente intuido, fortes relações com a área de Sistemas Imunes Artificiais, e em geral envolve uma visão estocástica do sistema imune. Detalhes dessa abordagem podem ser verificados em (FORREST; HOFMEYR, 2001), (CHAO *et al.*, 2003), (CHAO *et al.*, 2004) e (CHAO, 2004).

A abordagem de sistema de informação é interessante sob o ponto de vista que o sistema imune pode ser então modelado como um conjunto de entidades que processam e trocam informações entre si. Essa visão, porém, pode

ser aplicada em diversos níveis. Os trabalhos anteriores usando essa metodologia, por exemplo, procuram modelar o sistema imune a partir de seu comportamento, utilizando-se análise estatística nesse processo.

Uma abordagem que apresenta-se mais interessante para a modelagem do sistema imune é baseada em agentes. O sistema imune é visto como um conjunto distribuído de agentes que processam diferentes tipos de informação[7]. Como comentado em (WARRENDER; FORREST; SEGEL, 2001, p.329):

> O sistema imune é um exemplo natural interessante de um sistema multi-agente. Ele é composto de um grande número e variedade de componentes distribuídos através do corpo. Ações individuais de um vasto número de células, e suas interações com um número ainda maior de mediadores moleculares, determinam o curso de uma infecção.

Um exemplo de uso da abordagem baseada em agentes é apresentada em (GRILO; CAETANO; ROSA, 2001), que utiliza um autômato celular para modelar o ambiente físico e algoritmos genéticos no processo de adaptação e seleção. De certa maneira, os modelos baseados em agentes possuem grande similaridade com os autômatos celulares e, em geral, tentam ultrapassar os limites impostos por esses últimos. Dependendo, inclusive, da granularidade como os elementos do autômato são definidos, uma solução baseada em autômatos celulares pode ser considerada como utilizando abordagem baseada em agentes.

Entre os modelos baseados em agentes, merece destaque o CyCells[8], apresentado em (WARRENDER, 2004a). Em (WARRENDER, 2004b) é apresentado o uso desta ferramenta na modelagem de interações intercelulares no sistema imune periférico. Como comentado em (WARRENDER, 2004a), CyCells é um simulador tridimensional em tempo discreto para o estudo de interações celulares através de sinais extra-celulares, como pode ser visto na Figura 4.6.

Na opinião deste pesquisador, das propostas apresentadas, o CyCells é a ferramenta que atualmente utiliza a abordagem mais adequada ao atual contexto de conhecimento biológico e poder computacional. Esse tipo de abordagem tende a ser mais sistêmico e evita a modelagem de tautologias, pois resultados são obtidos mais a partir da evolução do sistema que de suposições iniciais. Óbvio que conceitos pré-estabelecidos podem ainda influenciar

[7] Uma conceituação acurada de sistema multi-agente pode ser encontrada em (YEPES; BARONE, 2003) e (GARCIA; SICHMAN, 2003).

[8] CyCells: `http://cs.unm.edu/~christy/simcode/`.

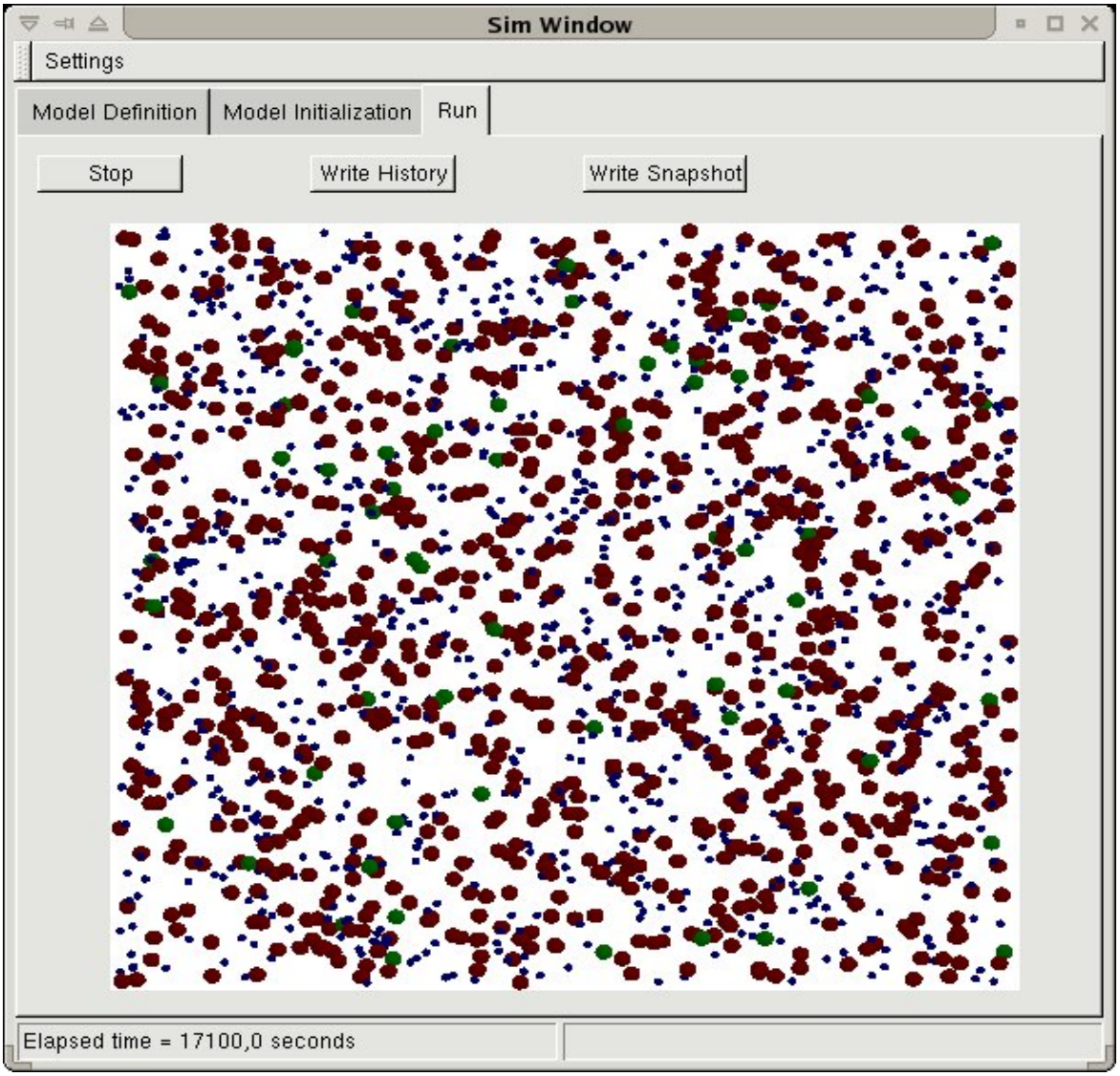


Figura 4.6: CyCells em Execução

fortemente o sistema, mas o impacto é menor que, por exemplo, utilizando-se equações diferenciais.

Entre as ações celulares projetadas no CyCells, encontram-se apopotse, diferenciação e divisão celular, influxo (entrada da célula em um compartimento), secreção molecular e movimentação. Condições podem ser estabelecidas para que essas ações ocorram, podendo ser definidas por meio de variáveis estocásticas, ou operações simples como “maior que” ou “menor que”. Condições podem ainda ser compostas logicamente, explicitando se é necessário que duas condições sejam atendidas integralmente ou apenas uma entre as duas.

Dadas essas características, o CyCells aponta-se como uma promissora ferramenta para modelagem do sistema imune. Entretanto, alguns problemas

detectados por este pesquisador tiram o brilho dessa ferramenta. Algumas das falhas são plenamente justificáveis pelo foco do trabalho de Christina Warrender, que foi a modelagem do sistema imune periférico, especialmente interação macrófago-patógeno (WARRENDER, 2004b).

Uma limitação do CyCells é que ele não possui mecanismos para reconhecimento de padrões, o que trás problemas para a representação da memória imune. Além disso, isso impossibilita a representação de especificidade dos linfócitos, mas permite de forma relativamente adequada a modelagem de macrófagos. Cabe ressaltar que mesmo no caso de modelagem de macrófagos, o CyCells não permite representar explicitamente a presença ou não de determinados receptores, não possibilitando esse nível de granularidade.

Outro problema, mais crítico sob o ponto de vista do trabalho ao qual o CyCells foi proposto, é que não existe nenhum controle efetivo sobre o processo de fagocitose, especificando-se apenas qual célula deve ser fagocitada por uma outra. Não existe aqui nenhum condicionamento, o que poderia ser implementado com relativa facilidade, dado que outras funções são executadas sob condições. Na opinião deste pesquisador, o CyCells deveria ao menos verificar em um dado atributo se a célula foi ativada, a partir da percepção do ambiente, antes de fagocitar uma dada outra. Entretanto, a fagocitose no CyCells é definida apenas pelo tipo celular e distância entre células.

Outro problema prático é que a saída do CyCells, seu arquivo de resultados, não é compatível com os arquivos de entrada (modelo e inicialização). Isso impede que se interrompa o processo, para posterior reinício, ou mesmo que se altere alguns elementos para que sejam verificadas o comportamento do sistema a partir de duas situações distintas (a inclusão de um patógeno em um momento intermediário, por exemplo). Esse tipo de procedimento é crucial para o estudo da memória imune, por exemplo, analisando-se as inúmeras possibilidades de comportamento do sistema.

Ainda assim, entretanto, os méritos do CyCells são grandes, principalmente por utilizar uma abordagem baseada na modelagem da interação celular, com os elementos representados como agentes do sistema. Comparado às alternativas existentes, essa abordagem permite um maior grau de detalhamento dos ítens modelados, possibilitando testes relativamente detalhados de hipóteses que sejam baseadas na interação entre os elementos. A área de modelagem e simulação do sistema imune deverá cada vez mais receber contribuições de abordagens similares, na opinião deste pesquisador.

4.8 Comentários Finais

Em um divertido artigo publicado na Cancer Cell, Lazebnik faz uma extrapolação sobre o uso das atuais abordagens para estudos biológicos aplicadas ao conserto de um rádio (LAZEBNIK, 2002). Após diversas suposições sobre os resultados desse uso, esse pesquisador conclui que a ausência de uma linguagem formal quantitativa para descrição de entidades biológica é uma séria deficiência na pesquisa biológica atual. Além disso ele aponta que:

> O poder computacional disponível e avanços na análise de sistemas complexos aumentam a esperança que é próxima a época em que a abordagem de sistemas irá mudar de uma ferramenta esotérica, considerada inútil por muitos biólogos experimentais, para uma abordagem básica e indispensável à Biologia.
>
> (LAZEBNIK, 2002, p.182)

Como apontado em (PETROVSKY; SILVA; BRUSIC, 2003), uma perspectiva interessante é o desenvolvimento *in silico* de modelos de sistemas integrais. A tecnologia disponível atualmente já permite vislumbrar a construção de um sistema imune virtual, e essa construção deverá ser um dos maiores desafios da próxima década. Esse fato também é apontado em (CHAKRABORTY; DUSTIN; SHAWN, 2003), "progresso futuro poderá envolver o uso de modelos computacionais acurados ('rato transgênico digital')". Como apontado nesse artigo, entretanto, os resultados de experimentos *in silico* dependem diretamente do modelo sendo simulado, exigindo-se cuidado na formulação desse modelo. Como ressaltado por (PETROVSKY; BRUSIC, 2002, p.248):

> Cuidados precisam ser tomados para que as ferramentas computacionais sejam adequadas e modelem apropriadamene os processos imunológicos. Preconceitos e tendências podem ser facilmente codificadas em ferramentas computacionais (...)

Essa afirmação é extremamente válida em Imunologia, uma vez que é uma ciência rica em hipóteses e mesmo opiniões. Conforme (KITANO, 2002), sistemas biológicos não devem ser encarados como sistemas complexos. Por sistemas complexos geralmente entende-se um conjunto grande de elementos simples interagindo para produzir comportamento complexo. Em um sistema biológico, isso não ocorre necessariamente: é possível que, dentre os elementos, haja algum com complexidade maior que os demais. Dessa maneira, para

esse autor, sistemas biológicos são melhor caracterizados como sistemas simbióticos.

Obviamente, essa discussão pode ser tomada apenas como uma questão de linguagem, uma vez que não existe na literatura uma definição padrão de sistemas complexos que seja adotada integralmente por vários pesquisadores. O que pretende-se aqui, ao apontar essa questão é que uma implementação computacional de simulação de fenômenos do sistema imune deve permitir riqueza de detalhes e vários níveis de complexidade na descrição dos elementos.

Nesse sentido, tomando-se os problemas e limitações subjacentes às propostas atuais de modelagem do sistema imune (ver Seção 4.7), este trabalho sustenta a hipótese de que a melhor abordagem atual são aquelas baseadas em agentes, em que agentes representam entidades do sistema imune. Como comentado em (WARRENDER, 2004b, p.11)

> Existe uma série de razões para representar o sistema imune como uma população discreta de células ao invés de variáveis contínuas em equações diferenciais. Em modelos baseados em indivíduos, cada elemento tem seu próprio estado que é atualizado de acordo com certas regras direcionando as interações com outros indivíduos do ambiente. Propriedades do modelo são portanto mais relacionadas às propriedades dos componentes do sistema modelado que se utilizasse equações diferenciais. Modelos discretos podem também representar explicitamente o alto grau de heterogeneidade que existe em sistemas multicelulares.

Por fim, é importante observar que a modelagem do sistema imune tem inspirado o desenvolvimento de modelos e algoritmos computacionais, como pode ser percebido, por exemplo em (FORREST; BEAUCHEMIN, 2007). Esse texto, inclusive, utiliza o nome Imunologia Computacional para designar o conjunto de trabalhos computacionais com funcionamento análogo, em maior ou menor escala, ao sistema imune natural. Assim, essa área incluiria não apenas os trabalhos de modelagem e simulação, mas também os Sistemas Imunes Artificiais, foco deste trabalho e tema do próximo capítulo.

CAPÍTULO
5

Sistemas Imune Artificiais

As relações entre os homens, o trabalho, a própria inteligência dependem, na verdade, da metamorfose incessante de dispositivos informacionais de todos os tipos. Escrita, leitura, visão, audição, criação, aprendizagem são capturados por uma informática cada vez mais avançada.

Pierre Lévy, As Tecnologias da Inteligência

Ao perder as flores
Com o templo se confunde
A cerejeira.

Buson

5.1 Comentários Iniciais

UMA área em Inteligência Computacional com profunda inspiração em Imunologia são os Sistemas Imunes Artificiais (SIAs), um tipo de Algoritmos Bioinspirados. Inclusive o termo "Algoritmos Imunoinspirados" possui uma maior adequação lingüística, uma vez que a princípio, os SIAs consistem na aplicação de conceitos e teorias imunológicas em problemas de Engenharia (otimização e/ou aprendizado de máquina). O termo "Algoritmos Imunoinspirados" propicia uma maior aproximação com a realidade, uma vez que nem

todo uso dos SIAs pode ser encarado facilmente como um "sistema imune". Entretanto, da mesma maneira que "Redes Neurais Artificiais", o termo "Sistemas Imunes Artificiais", apesar de não ser totalmente adequado, encontra-se fortemente estabelecido na comunidade e será usado neste texto.

Dado esse enfoque, o pesquisador Leandro Nunes de Castro propõe, inclusive, o termo Engenharia Imunológica, como sendo "(...) uma estrutura formal para o desenvolvimento de sistemas imunológicos artificiais" (DE CASTRO, 2001). Como comentado nesse mesmo texto, os Sistemas Imunológicos Artificiais surgiram a partir das tentativas de modelar e aplicar os princípios imunológicos no desenvolvimento de novas ferramentas computacionais.

SIAs surgiram em meados da década de 1980, sendo que o trabalho de Farmer, Packard e Perelson em redes imunes (FARMER; PACKARD; PERELSON, 1986) é considerado pioneiro na área. Entretanto, o estabelecimento de SIAs como uma área deu-se somente em meados da década de 1990. Nessa primeira etapa foram significativas as contribuições das equipes de Stephanie Forrest e Dipankar Dasgupta, com estudos em algoritmos de seleção negativa.

No final da década de 90, tornaram-se conhecidos os trabalhos envolvendo seleção clonal, destacando-se a contribuição dada por Leandro Nunes de Castro e Fernando Von Zuben, entre outros, cujos trabalhos inspiraram este projeto de pesquisa. Mais recentemente, uma certa inovação é apresentada por Uwe Aickelin, utilizando o Modelo do Perigo como base para novos algoritmos. É importante destacar que em 1998 é publicado o primeiro livro na área, editado por Dasgupta (DASGUPTA, 1998), e em 2002, sai o segundo livro, de autoria de Leandro Nunes e Timmis (DE CASTRO; TIMMIS, 2002b). SIAs também são assunto de um extenso capítulo de (DE CASTRO, 2006a), livro também de autoria de Leandro Nunes.

A filosofia subjacente aos SIAs é que muitos processos biológicos podem ser encarados, a grosso modo, como modelos naturais de otimização ou aprendizado. Usando-se essa simplificação, é possível, por exemplo, encarar o processo de defesa do organismo às doenças como uma "otimização" da resposta do organismo, em especial os anticorpos, aos antígenos a ele apresentados.

Originalmente SIAs foram criados apenas como abstrações de processos encontrados no sistema imune. Recentemente, entretanto, vários pesquisadores da área tornaram-se interessados no processo de modelagem do sistema imune e aplicação de SIAs a problemas imunológicos. Isso tem relação com o fato, ressaltado em (DE CASTRO; TIMMIS, 2003), que muitos pesquisadores sentem dificuldades em identificar a diferença entre SIAs e trabalhos em Imu-

nologia Teórica. Nesse texto, SIAs são conceituados como sistemas computacionais para solução de problemas, inspirados por Imunologia teórica, bem como funções, princípios e modelos imunológicos observados. Dessa maneira, o diferencial entre essas duas áreas encontra-se na aplicabilidade:

> Enquanto trabalhos em Imunologia Teórica têm usualmente por objetivo modelar e prover uma melhor visão de experimentos laboratoriais e o funcionamento do sistema imune, trabalhos em SIAs são aplicados para resolver problemas em Computação, Engenharia e outras áreas de pesquisa.
>
> (DE CASTRO; TIMMIS, 2003, p.8)

Como destacado por esses autores, SIAs, portanto, constituem-se como um paradigma de *soft computing*, de maneira similar a Algoritmos Genéticos, Redes Neurais e Lógica *Fuzzy*. Entre as características dos SIAs, destacam-se o controle distribuído, o reconhecimento de padrões e a diversidade.

Dada a amplitude de idéias em SIAs, existem aplicações dos mesmos em diversas áreas, como segurança da informação, detecção de falhas e outros, como apontado em (TARAKANOV; DASGUPTA, 2002). Uma visão geral das diversas aplicações de SIAs pode ser verificada em (DASGUPTA; ATTOH-OKINE, 1997), (FREITAS; TIMMIS, 2007) e (DE CASTRO; VON ZUBEN, 2000a). Uma abordagem mais detalhada sobre SIAs pode ser encontrada em (DE CASTRO; VON ZUBEN, 1999), (GREENSMITH, 2003), (DE CASTRO; VON ZUBEN, 2000a), (DE CASTRO; VON ZUBEN, 2002) e (HOFMEYR; FORREST, 2000). Entre trabalhos recentes que apresentam uma visão geral dos SIAs, recomenda-se a leitura de (DE CASTRO; TIMMIS, 2003), (DE CASTRO; TIMMIS, 2002c), (DE CASTRO, 2006b), (AICKELIN; DASGUPTA, 2006) e (DASGUPTA, 2006).

Entre inúmeras aplicações dos SIAs, podem ser citadas: reconhecimento de padrões, clusterização, aprendizado, otimização, etc. Como apontado em (ANDREWS; TIMMIS, 2005), a maior parte dos trabalhos envolvendo SIAs são desenvolvidos tomando-se por base a teoria de seleção clonal de Burnet ou a teoria de rede idiotípica de Jerne. Merecem destaque também os trabalhos utilizando seleção negativa e mais recentemente os inspirados no Modelo do Perigo. As próximas seções detalham as principais classes de algoritmos dos SIAs, bem como apresentam uma discussão sobre modelos híbridos e perspectivas futuras.

5.2 SIAs Inspirados na Teoria da Rede Idiotípica

Os algoritmos de rede idiotípica baseiam-se no processo de regulação imune propostos a partir das idéias iniciais apresentadas em (JERNE, 1974). Existe uma grande diversidade de algoritmos inspirados na teoria da rede idiotípica, o que pode ser verificado em (DE CASTRO; TIMMIS, 2003). Um dos algoritmos mais conhecidos, o aiNet (DE CASTRO; VON ZUBEN, 2000b), é apresentado na Figura 5.1. As características dessa classe de algoritmos os tornam adequados para problemas envolvendo clusterização e reconhecimento de padrões.

1. *Inicialização:* crie uma população inicial aleatória de anticorpos da rede;
2. *Apresentação antigênica:* para cada padrão antigênico, faça:
   - 2.1 *Seleção e expansão clonal:* para cada elemento da rede, determine sua afinidade com o o antígeno apresentado. Selecione um número de elementos de alta afinidade e reproduza-os (clonalmente) proporcionalmente à afinidade;
   - 2.2 *Maturação de afinidade:* aplique mutação a cada clone de forma proporcionalmente inversa à sua afinidade. Selecione novamente um número de clones com alta afinidade e coloque-os em um conjunto de memória clonal;
   - 2.3 *Interações clonais:* determine as interações da rede (afinidade) entre os elementos do conjunto de memória clonal;
   - 2.4 *Supressão clonal:* elimine os clones de memória cuja afinidade seja inferior a um dado limite pré-especificado.
   - 2.5 *Metadinâmica:* elimine todos os clones de memória cuja afinidade com o antígeno é inferior a um determinado limite;
   - 2.6 *Construção da Rede:* incorpore os clones restantes da memória clonal com a rede de anticorpos;
   - 2.7 *Interação da Rede:* determine a similaridade entre cada para dos anticorpos da rede;
   - 2.8 *Supressão da Rede:* elimine todos os anticorpos da rede cuja afinidade seja inferior a um dado limite;
3. Ciclo: repita o passo 2 até que uma dada condição de parada seja satisfeita.

Figura 5.1: aiNet – Fonte: (DE CASTRO; TIMMIS, 2003)

Como já comentado, (FARMER; PACKARD; PERELSON, 1986) é considerado o primeiro trabalho em SIAs. Nesse trabalho os autores desenvolvem um classificador a partir de um sistema dinâmico não-linear baseado em uma rede de anticorpos modelada por equações diferenciais. Destaca-se nesse trabalho algo defendido neste texto, que é a proximidade entre SIAs e modelagem

do sistema imune, pois apesar do foco desse trabalho ser o desenvolvimento de novos algoritmos de reconhecimento de padrões, os autores comentam que seu principal objetivo é a simulação do sistema imune, para uma maior compreensão do mesmo em organismo reais.

Entre os trabalhos que utilizam dessa abordagem, pode-se citar (KIM; CHO, 2004) e (KIM; LEE, 2004) que usam uma rede imune para otimização de controladores PID (*Proportional – Integral – Derivative*) de temperatura em termoelétricas. Em (TARAKANOV; DASGUPTA, 2002) é proposta uma arquitetura para um *immunochip*, baseado em rede imune, com apresentação de resultados utilizando simulação em *software*.

5.3 SIAs Inspirados na Teoria de Seleção Clonal

Como comentado em (ABBAS; LICHTMAN; POBER, 2003), cada indivíduo possui inúmeros anticorpos derivados clonalmente. Cada clone origina-se de um precursor único, capaz de reconhecer e responder um determinante antigênico distinto e, quando, o antígeno entra, seleciona um clone específico pré-existente, ativando-o. Isso vem a constituir a *hipótese ou teoria de seleção clonal* e, conforme apontado em (ABBAS; LICHTMAN; POBER, 2003), foi proposto por Niels Jerne em 1955 e mais claramente enunciado por Macfarlane Burnet em 1957 (BURNET, 1959).

Assim, seguindo-se esse raciocínio, quando um animal é exposto a um dado antígeno, o organismo irá, de certa maneira, ativar um determinado anticorpo, cuja produção é estimulada. Esse antígeno irá estimular o organismo a fazer com que determinadas células se proliferem e transformem-se em uma outra célula capaz de secretar anticorpos em altas taxas, como destacado em (DE CASTRO; VON ZUBEN, 2002). Mais detalhes sobre resposta imune podem ser encontrados em (JANEWAY *et al.*, 2001).

É importante enfatizar que, apesar de ser questionada por alguns pesquisadores, a teoria de seleção clonal é uma fonte de idéias para o desenvolvimento de Sistemas Imunes Artificiais e outros modelos evolutivos computacionais. Entre os textos apresentando as falhas do princípio de seleção clonal enquanto proposta de funcionamento do sistema imune, destacam-se (VAZ; FARIA, 1993) e (COUTINHO, 2003).

Entre os algoritmos baseados na teoria de seleção clonal, o mais conhecido e utilizado é com certeza o CLONALG (DE CASTRO; VON ZUBEN, 2002),

apresentado na Figura 5.2. Esse algoritmo tem sido aplicado na literatura em problemas de aprendizado de máquina ou otimização. No caso de resolução de problemas de otimização, o antígeno é a função a ser otimizada. Assim, o vetor de afinidades é determinado pelo grau em que cada anticorpo otimiza essa função. Além disso, a cada iteração, a família anterior de anticorpos é substituída por novos anticorpos.

1. Em um dado instante de tempo i, um antígeno Ag é apresentado a todos os anticorpos da população Ab.
2. Um vetor de afinidades f_i em relação aos anticorpos da população Ab é determinado.
3. Do conjunto Ab, um subconjunto Ab_i é selecionado, baseado na medida de afinidade f_i de cada anticorpo de Ab com relação ao antígeno Ag.
4. Os n indivíduos selecionados irão se proliferar, através de um processo de clonagem proporcional à afinidade do anticorpo com o antígeno: quanto maior a medida de afinidade, maior o número de clones de cada um dos n indivíduos. Após essa clonagem, é constituída uma população de clones Ab_c.
5. A população de clones Ab_c é submetida a um processo de maturação de afinidade, gerando uma nova população Ab_m, onde cada anticorpo de Ab_c irá sofrer uma mutação com taxa inversamente proporcional à sua afinidade: quanto maior a afinidade, menor a taxa de mutação.
6. Um novo vetor de afinidades f_m é calculado, agora para os anticorpos da população Ab_m.
7. A população Ab é substituída pelos melhores elementos de Ab e Ab_m.

Figura 5.2: CLONALG - Fonte: (DE CASTRO, 2001)

Em (LI *et al.*, 2004), por exemplo, é apresentado a otimização de projetos de controladores *fuzzy* baseado no princípio de seleção clonal. Um trabalho correlato para controladores neuro-*fuzzy* é apresentado em (ZUO; LI; BAN, 2003). Em (GUIMARÃES *et al.*, 2007) são apresentados dois algoritmos inspirados no princípio de seleção clonal, um baseado em representação de dados com representação de números reais (RCSA) e outro para utilização em otimização multiobjetivo (MCSA). Um maior detalhamento desses algoritmos, bem como a apresentação de alguns outros algoritmos inspirados no sistema imune para aplicação em otimização, pode ser encontrada em (BATISTA, 2009). Uso do RCSA em problema de otimização de *design* eletromagnético ou magnético pode ser verificado, respectivamente em (CAMPELO *et al.*, 2005) e (CAMPELO; NOGUCHI; IGARASHI, 2006). Um algoritmo baseado em seleção clonal para otimização de topologia 3D é apresentada em (CAMPELO; WATANABE; IGA-

RASHI, 2007). Aplicação de seleção clonal em problemas de expansão de rede de distribuição elétrica pode ser verificada em (CARRANO *et al.*, 2007).

Alguns testes realizados durante este projeto, aplicado na resolução do *Timetabling*[1], permitiram concluir que o CLONALG é um sério competidor para Algoritmos Genéticos clássicos, com as seguintes particularidades:

- CLONALG não utiliza o operador de *crossover* para trocar propriedades individuais;
- para compensar a ausência de mistura de propriedades individuais, o algoritmo utiliza uma taxa muito maior de mutação, permitindo um nível alto de variabilidade;
- outro efeito interessante é a distribuição da população em ótimos locais, com maior diversidade entre os elementos.

Neste sentido, é melhor utilizar o CLONALG para resolver problemas cuja representação torna o *crossover* um operador sem sentido, complexo ou sem utilidade. Outros testes e análises precisam ser executados para permitir respostas mais conclusivas sobre quando é melhor utilizar CLONALG ou Algoritmos Genéticos.

5.4 SIAs Inspirados na Teoria de Seleção Negativa de Células T

UMA terceira classe de algoritmos de SIAs é baseado no processo de seleção negativas de células T no timo. De acordo com essa hipótese, são eliminadas do organismo células T que reconheceriam o *self*. Dessa maneira, as células T que sobreviveriam seriam capazes de reconhecer apenas antígenos não-próprios. A partir dessa idéia, em (FORREST *et al.*, 1994), é proposto um algoritmo de seleção negativa, tendo-o aplicado na detecção de vírus de computadores. O algoritmo, relativamente simples, é apresentado na Figura 5.3.

É importante aqui comentar que Stephanie Forrest é professora da Universidade de Novo México e pesquisadora no Instituto de Santa Fe. Esse instituto é muito conhecido pelas diversas pesquisas na área de sistemas complexos.

[1] O *Timetabling* é conhecido problema de alocação de horários, que consiste em agendar o encontro de professores e estudantes para a realização de disciplinas em várias salas.

1. Gere um conjunto aleatório de *strings* R_0, determinando a afinidade de cada uma dessas *strings* com o conjunto protegido (self) S. Crie um conjunto de detectores R com as *strings* com afinidade superior a um dado limite, i.e., aquelas *strings* que não são ativadas pelo *self*.

2. Monitore os dados protegidos S, comparando-os com os detectores R. Se um detector for ativado, então ocorreu uma mudança no sistema, tendo sido detectado um elemento não-próprio.

Figura 5.3: Algoritmo de Seleção Negativa - Adaptado de (FORREST *et al.*, 1994)

Como seria de se esperar, os trabalhos de Forrest incluem não somente Sistemas Imunes Artificiais, mas também a modelagem do sistema imune. Seus esforços na área de Sistemas Imunes Artificiais vão principalmente na direção da construção de sistemas computacionais imunes, sob a ótica de Segurança Computacional. Isso pode ser visto, por exemplo, em (FORREST; HOFMEYR; SOMAYAJI, 1997) e (HOFMEYR; FORREST, 2000).

Outro pesquisador de SIAs que tem feito um uso freqüente dessa classe de algoritmos é Dipankar Dasgupta, que possui como característica fundamental o uso de algoritmos de seleção negativa utilizando uma representação em valores reais. Detalhes do algoritmo e sua implementação são apresentadas em (JI; DASGUPTA, 2005), incluindo-se resultados obtidos em reconhecimento de padrões.

Esse método foi criticado na literatura, por exemplo em (STIBOR *et al.*, 2005), que aponta que o algoritmo padrão é ineficiente, dado que um número muito grande de detectores gerados aleatoriamente precisam ser descartados, antes da obtenção de um bom conjunto de detectores. Para esses autores, portanto, esse método acaba se tornando uma busca aleatória simples. Apesar dessa ser uma opinião extrema sobre o assunto, é possível verificar uma certa tendência nessa direção para essa classe de algoritmo. Em (STIBOR; TIMMIS; ECKERT, 2005), são apresentados problemas no uso de seleção negativa baseada em valores reais na aplicação em conjuntos de dados com muitas dimensões.

Defensores desse método argumentam que os algoritmos dessa classe compartilham apenas alguns elementos centrais. Para (JI; DASGUPTA, 2006) e (JI; DASGUPTA, 2007), essa família de algoritmo é altamente flexível, permitindo o uso de diferentes estratégias dentro dela. Dessa maneira, muitos dos problemas apontados na literatura seriam, na opinião desses pesquisadores, derivados de mal uso do método. Mais ainda, para eles, as reais dificuldades

com seleção negativa existem também para outras classes, incluindo a decisão sobre um bom modelo de dados e o domínio da aplicação.

5.5 SIAs Inspirados no Modelo do Perigo

Em (GREENSMITH; AICKELIN; CAYZER, 2005) é apresentado um algoritmo de detecção de anomalias baseado no processo de maturação das células dendríticas, inspirado pelo Modelo do Perigo. O algoritmo, apresentado na Figura 5.4, baseia-se em células dendríticas imaturas que residem no tecido, onde coletam material antigênico e são expostas a sinais endógenos e exógenos. A partir da combinação desses materiais, são geradas células dendríticas maduras ou semi-maduras. Células maduras tem efeito ativador, enquanto as semi-maduras seriam supressoras, induzindo tolerância.

```
Crie um grupo de 100 células dendríticas
Para cada ítem de dados
  Pegue 10 células dendríticas do grupo
    Para cada célula dendrítica
      Adicione antígeno (etiqueta do dado) na lista de antígenos coletados
      Atualize a concentração dos sinais de entrada
      Calcule concentrações de citocinas de saída
      Atualize o total das citocinas de saída
      Se total de moléculas co-estimulatórias for maior que limite fuzzy
        Remova célula do grupo e migre-a
        Crie nova célula dendrítica

Para cada célula dendrítica que migrar
  Se concentração de semi for maior que concentração de maduras
    contextoDoAntígeno = semi
  Senão
    contextoDoAntígeno = maduro

Para cada antígeno que entrar no sistema
  Calcule o número de vezes apresentado como maduro ou semi
    Se semi > maduro
      Antígeno = benigno
    Senão
      Antígeno = maligno
```

Figura 5.4: Algoritmo de Células Dendríticas – Fonte: (GREENSMITH; AICKELIN; CAYZER, 2005)

Uma apresentação mais recente desse algoritmo pode ser encontrada em (GREENSMITH; AICKELIN; CAYZER, 2008). Entre as aplicações dessa abordagem, destacam-se principalmente detecção de padrões (WILSON; BIRKIN; AICKELIN, 2007), segurança computacional (GREENSMITH; AICKELIN, 2007) e detecção de anomalias (GREENSMITH; TWYCROSS; AICKELIN, 2006).

Cabe destacar que a literatura de SIAs baseados no Modelo do Perigo são quase que totalmente focados em imunidade inata. Isso ocorre apesar da criadora dessa teoria ter apresentado um algoritmo relativamente completo do sistema imune e que é passível de implementação, como pode ser verificado na Seção 4.4, Figura 4.4.

5.6 Comparação com Outras Abordagens e Modelos Híbridos

EXISTEM inúmeras pesquisas de modelagem de sistemas biológicos em computadores e a posterior utilização desses modelos para resolução de problemas diversos. Exemplos clássicos dessa abordagem são os Algoritmos Genéticos, apresentados em (MITCHELL, 1996), (HOLLAND, 1992) e (ZEBULUM; PACHECO; VELLASCO, 2002). Outro exemplo são as Redes Neurais Artificiais, apresentadas em (FAUSETT, 1994), (BRAGA; CARVALHO; LUDERMIR, 2000b) e (HAYKIN, 2001).

Sistemas Imunes Artificiais, portanto, pertencem a uma classe de algoritmos em computação denominada de algoritmos bioinspirados, possuindo similaridades com Redes Neurais Artificiais, mas principalmente com Computação Evolutiva e *Algoritmos de Enxames*. Esses algoritmos, por sua vez contribuem para o estabelecimento da *Soft Computing* (Computação Flexível), um paradigma em inteligência computacional baseado em modelos probabilísticos, incluindo além dos algoritmos bioinspirados: Lógica *Fuzzy* (ZADEH, 1965; ZADEH, 1979), Teoria de Conjuntos Aproximados (PAWLAK, 1991; PAWLAK, 1982; UCHÔA, 1998), Teoria de Dempster-Shafer (SHAFER, 1976), e outros métodos com filosofias similares.

A *Soft Computing* consiste no uso de abordagens híbridas, integradas ou não, para a solução de problemas diversos. De acordo com (ZADEH, 1994), o principal objetivo desse paradigma é "obter tratabilidade, robustez, baixo custo de solução e alto quociente de inteligência da máquina (MIQ – *Machine Intelligence Quotient*) através da exploração da tolerância a imprecisão e incerteza". Como apontado em (WANG; TAN, 1997), a Lógica *Fuzzy* possui grande destaque nessas abordagens. Assim, de forma diferente a métodos mais clássicos, a *Soft Computing* valoriza o uso de modelos que permitem a representação de incerteza. Mais detalhes sobre a *Soft Computing* podem ser verificados em (BONISSONE, 1997) e (NOVÁK, 1998).

Uma interessante discussão epistemológica sobre a definição de *Soft Computing* é apresentada em (DUBOIS; PRADE, 1998), que comenta que a definição atual dessa disciplina ainda não é satisfatória, pois *"cada um de seus componentes tem pouco em comum com o outro"*. Esses autores sugerem, portanto, uma visão mais ampla dessa área, como

> um campo dedicado a métodos de solução de problemas capazes de explorar simultaneamente dados numéricos e conhecimento humano, usando modelagem matemática e sistemas de raciocínio simbólico.
>
> (DUBOIS; PRADE, 1998, p.11)

Dentro desse contexto, uma apresentação geral de SIAs, Computação Evolutiva e Algoritmos de Enxames é apresentada em (DE CASTRO, 2002a). A Computação Evolutiva fundamenta-se em algoritmos que tentam mimetizar processos da evolução natural, sendo os Algoritmos Genéticos seus maiores representantes. Já a classe de Algoritmos de Enxames, da qual faz parte os Algoritmos de Colônias de Formigas, é fundamentada na propriedade de alguns sistemas apresentarem comportamento coletivo inteligente a partir de elementos simples com capacidade limitada. Em (DE CASTRO, 2002b), é apresentada uma comparação entre essas três classes de algoritmos, sendo que, como apontado nesse texto, a diferença entre SIAs e computação evolutiva é sutil. Isso é válido principalmente para os algoritmos baseados no princípio de seleção clonal, como já destacado na Seção 5.3.

Em (TARAKANOV; TARAKANOV, 2005), os autores compararam Algoritmos Genéticos com um algoritmo baseado em rede imunológica, obtendo melhores resultados com SIAs em uma série de tarefas. Obviamente, dada as características inexatas dessa classe de algoritmo, qualquer comparação desta natureza é incompleta e, em geral, tendenciosa.

Testes realizados por nossa equipe com Algoritmos Genéticos clássicos e CLONALG sugere que é melhor utilizar o CLONALG para resolver problemas cuja representação torna o *crossover* um operador sem sentido, complexo ou sem utilidade. Esse é o caso, por exemplo, do *Timetabling*, que consiste na elaboração de uma agenda entre participantes, de forma a satisfazer uma série de restrições. Um exemplo real do *Timetabling* é a alocação de horários por turmas e professores em escolas, por exemplo. Além das restrições de professores, existem uma série de outros fatores que tornam essa tarefa relativamente complexa: i) preferência de professores por certos dias e horários; ii) não pode haver choques de atividades entre os participantes, aulas do mesmo

professor no mesmo horário para turmas diferentes; iii) necessidade de otimizar as aulas em blocos, geralmente de duas aulas, evitando aulas esparsas, etc.

Em (TIMÓTEO, 2002), são apresentados mais detalhes sobre o *Timetabling*, bem como uma solução utilizando Algoritmos Genéticos. Neste trabalho, orientado por este pesquisador, uma das primeiras dificuldades encontradas foi justamente a definição do operador de *crossover*, uma vez que, a partir da representação escolhida para as possíveis soluções, o cruzamento de diferentes indivíduos levava a soluções inviáveis. Obviamente, isso pode ser um problema advindo da forma de representação adotada para o problema, mas aponta para dificuldades em situações semelhantes. Testes realizados por este pesquisador permitiu a obtenção de melhores resultados com o CLONALG que com Algoritmos Genéticos na resolução do *Timetabling* e eliminou a dificuldade de formulação de um *crossover* adaptado ao problema.

Além dos Algoritmos Genéticos, os SIAs possuem alguma proximidade com as Redes Neurais Artificiais, principalmente os inspirados na Teoria da Rede Imunológica. Nesse sentido a Tabela 5.1, extraída de (AICKELIN; DASGUPTA, 2006), apresenta uma visão geral comparativa desses algoritmos. Como apontada nesse texto, essa Tabela é uma simplificação grosseira, mas extremamente valiosa para que se possa situar os SIAs em um contexto mais amplo. Deve-se comentar ainda que essa comparação foi baseada em um Algoritmo Genético clássico, utilizado em otimização, e modelos clássicos de Redes Neurais Artificiais para classificação. Uma discussão mais detalhada da comparação de SIAs com esses métodos pode ser encontrada no Capítulo 6 de (DE CASTRO; TIMMIS, 2002b).

A partir da Tabela 5.1, é possível perceber que os SIAs agregam algumas vantagens e características existentes em Algoritmos Genéticos e Redes Neurais Artificiais. Como apontado em (AICKELIN; DASGUPTA, 2006), SIAs, assim como outros algoritmos evolucionários, são relativamente robustos com relação aos parâmetros de configuração, dado que sejam escolhidos em uma faixa sensível ao algoritmo. Em (DE CASTRO; TIMMIS, 2002c), que apresenta uma excelente comparação entre SIAs e Redes Neurais Artificiais, é apontado que SIAs e Redes Neurais Artificiais são altamente flexíveis e tolerantes a ruídos. SIAs também compartilham com essas classes de algoritmos a capacidade de generalização e a existência e o suporte para não-linearidade, características bastante desejáveis na resolução de diversos problemas.

	Algoritmos Genéticos	**Redes Neurais Artificiais**	**SIAs**
Componentes	*Strings* de cromossomos	Neurônios artificiais	*Strings* de atributos
Localização dos componentes	Dinâmica	Pré-definida	Dinâmica
Estrutura	Componentes discretos	Componentes em rede	Componentes discretos/em rede
Armazenamento de Conhecimento	*Strings* de cromossomos	Pesos das conexões	Concentração dos componentes / conexão da rede
Dinâmica	Evolução	Aprendizado	Evolução/Aprendizado
Meta-dinâmica	Recrutamento / eliminação de componentes	Construção / ajuste de conexões	Recrutamento / eliminação dos componentes
Interação Entre Componentes	Cruzamento	Conexões da rede	Reconhecimento / conexões da rede
Interação Com o Ambiente	*Fitness*	Estímulo externos	Reconhecimento / função objetivo
Forma de Ativação	Aglomeração / compartilhamento	Ativação de neurônio	Afinidade entre componentes

Tabela 5.1: Comparação entre SIAs, Algoritmos Genéticos e Redes Neurais Artificiais – Fonte: (AICKELIN; DASGUPTA, 2006)

A escolha entre uma dessas classes para a resolução de um dado problema pode ser relativamente complexa, dado o grande leque de aplicação dos mesmos, bem como o grande número de modelos existentes em cada classe. Podem ser determinantes nessa escolha a natureza do problema e o conhecimento do especialista sobre os métodos em si. Entre os critérios que podem direcionar a escolha para o uso de um SIA, encontra-se o fato de que, em geral, os algoritmos desse paradigma mesclam vantagens de algoritmos baseados em redes com algoritmos baseados em seleção e mutação.

Vários modelos híbridos têm sido propostos na literatura, sendo que a Tabela 5.2, adaptada de (DE CASTRO; TIMMIS, 2003), apresenta uma visão geral de modelos híbridos envolvendo SIAs, Computação Evolutiva (CE), Lógica *Fuzzy* (LF) e Redes Neurais Artificiais (RNAs). Esses modelos híbridos são interessantes no sentido em que uma abordagem pode auxiliar na resolução de problemas encontrados utilizando-se apenas um paradigma. Apesar disso, cabe destacar que ainda não existem abordagens híbridas que possam ser consideradas bem estabelecidas na literatura, ou seja: ainda não é possível detectar quais tipos de hibridizagem serão preferidos na solução de problemas ou mesmo se esses modelos ganharão algum tipo de destaque, como os já clássicos modelos neuro-*fuzzy*.

Um dos primeiros modelos híbridos foi apresentado em (DE CASTRO; VON ZUBEN, 2001b), que avaliou o uso do SAND para inicialização de pesos em

Integração	Resultados
SIAs $\longleftrightarrow$ RNAs	• SIAs sugeriram novos modelos, arquiteturas e algoritmos de aprendizados de RNAs • SIAs forneceram capacidades extras de memórias para RNAs • SIAs foram utilizados para desenvolvimento de novas técnicas de inicialização de RNAs
SIAs $\longleftrightarrow$ CE	• CE forneceu novas definições de repertórios iniciais para SIAs • CE foi utilizada para estudar a avaliação da codificação genética de SIAs • SIAs foram utilizados para aperfeiçoar a convergência em algoritmos genéticos • SIAs foram utilizados para controlar limites em algoritmos genéticos • SIAs foram utilizados para o desenvolvimento de algoritmos genéticos co-evolucionários • SIAs foram utilizados para promover e manter nichos, espécies e diversidade em algoritmos evolucionários • Foi proposta uma versão imunológica de programação genética
SIAs $\longleftrightarrow$ LF	• LF tem sido utilizada para modelar ligação imunológica em SIAs • O uso de LF permitiu o desenvolvimento de SIAs com maior apelo biológico • SIAs foram utilizados como alternativas para esquemas de clusterização *fuzzy* • SIAs podem ser utilizados para modelar seleção em LF

Tabela 5.2: Modelos Híbridos Envolvendo SIAs – Adaptado de (DE CASTRO; TIMMIS, 2003)

redes neurais não-recorrentes, tendo comparado-o a cinco outros métodos. SAND é um algoritmo de têmpera simulada inspirado pela diversidade do sistema imune, descrito em detalhes em (DE CASTRO, 2001). SAND e dois outros métodos obtiveram os melhores resultados na comparação, sendo que o SAND tinha como vantagem sobre esses dois outros métodos o fato de não precisar fazer uso de dados de treinamento para estimar o conjunto inicial de pesos.

Além desse trabalho, em (DE CASTRO; VON ZUBEN, 2001a), os mesmos autores apresentaram uma abordagem baseada em rede imunológica para ini-

cialização de centros de redes neurais RBF (*Radial Basis Function*). Mais recentemente, em (DE CASTRO; VON ZUBEN; DEUS JR., 2003), os autores propuseram um modelo de rede neural competitiva *booleana*, também baseando-se em rede imunológica.

Aparentemente, destacam-se na literatura o uso de modelos híbridos utilizando SIAs e Lógica *Fuzzy*. Isso em parte é justificado pela facilidade da aplicação de conceitos da Lógica *Fuzzy* a outros paradigmas. Em (NASAROUI; GONZALEZ; DASGUPTA, 2002), por exemplo, é apresentada uma aplicação baseada em rede imunológica, utilizando-se Lógica *Fuzzy* no processo de reconhecimento. A aplicação desenvolvida foi utilizada para a geração de *profiles* de navegação *Web*.

Na área de detecção de falhas, vários trabalhos tem utilizado SIAs ou modelos híbridos. Em (YAKUWA *et al.*, 2002), por exemplo, é apresentado um algoritmo baseado em redes neuro-*fuzzy* e redes imunes para diagnóstico de falhas em sistemas dinâmicos. Por sua vez, em (GÓMEZ; GONZÁLEZ; DASGUPTA, 2003) é proposto um algoritmo baseado em seleção negativa, onde conjuntos *fuzzy* são utilizados para representar *self* e *nonself*, para aplicação em detecção de anomalias.

Em (FÉLIX; USHIO, 2005), é apresentada uma proposta interessante para o cálculo de afinidades entre o paratopo de um linfócito e um epítopo de um dado antígeno utilizando-se de conceitos da Teoria de Conjuntos Aproximados. Com o uso dessas estratégias, os autores puderam generalizar os linfócitos existentes, permitindo o uso de um menor número dos mesmos. Apesar de não estar claro no trabalho, o método é baseado em Seleção Negativa.

5.7 Representação de Conhecimento em SIAs

A partir dos modelos apresentados nas subseções anteriores, deve estar claro ao futuro projetista de SIAs que ele deverá tomar diversas decisões ao adotá-los na solução de um determinado problema. Em um primeiro momento, ele deverá escolher entre os vários tipos de algoritmos existentes, procurando o modelo mais próximo de sua solução. Após isso, deverá preocupar-se com a representação do conhecimento necessário para a busca dessa solução.

Para problemas normais de classificação de padrões ou otimização, modelos inspirados na Teoria de Seleção Clonal ou no Modelo do Perigo podem

mostrar-se mais aptos. Já para problemas onde clusterização é uma característica desejável, torna-se interessante o uso de modelos baseados em rede idiotípica. Este pesquisador não recomenda o uso de modelos baseados na Teoria da Seleção Negativa, exceto em problemas onde exista uma clara distinção entre próprio e não-próprio, dadas as premissas teóricas sobre as quais esses algoritmos se apóiam bem como os problemas apontados na Seção 5.4.

Após a escolha do algoritmo, a próxima etapa será a definição de como o conhecimento sobre o problema será representado. Essa representação irá depender do algoritmo adotado e pode, inclusive, ser um dos parâmetros utilizados para escolha desse. Como apontado em (AICKELIN; DASGUPTA, 2006), quatro decisões precisam ser tomadas: forma de codificação dos dados, escolha da medida de afinidade, forma de seleção e mecanismo de mutação dos elementos.

Em geral, os dados em SIAs são representados como *strings* binárias ou de caracteres, mas várias outras formas podem ser verificadas na literatura. A representação desses dados será influente por sua vez na escolha da medida de afinidade entre antígeno e os receptores do sistema imune. Deve-se observar, que até para se facilitar nesse processo, em geral antígenos e receptores possuem representações relativamente similares. Cabe destacar também que a medida de afinidade em SIAs, a grosso modo, representa papel semelhante à medida de *fitness* em Algoritmos Genéticos em boa parte dos casos.

Uma vez que se definam a forma de representação dos dados e a medida de afinidade dos elementos, é necessário que se definam critérios para manutenção de indivíduos antigos e seleção de novos indivíduos, bem como os mecanismos para o processo de mutação de uma geração para outra. Nesse caso, ressalta-se que essas decisões precisam estar em grande sintonia com o algoritmo a ser utilizado, uma vez que esses parâmetros irão definir etapas importantes de seu funcionamento.

É importante observar aqui que essas escolhas podem influenciar, e muito, no sucesso ou fracasso do uso de uma dada abordagem. Também é importante ressaltar que isso não é exclusivo dos SIAs, uma vez que escolhas similares precisam ser feitas utilizando-se outras metodologias. Escolhas muito parecidas, quase idênticas, precisam ser feitas utilizando-se Algoritmos Genéticos, por exemplo.

Um agravante no processo de escolha é que não existem métodos a priori para a definição dos mesmos, e uma boa escolha pode envolver uma forte dose de empirismo. Mais ainda: essa escolha é específica para um problema

pontual, não podendo ser generalizada em uma forma ampla para outros problemas. Essa situação é bastante coerente com o fato que a solução para o problema sendo resolvido exige a representação de conhecimento sobre o problema em si. Como a escolha da representação depende do problema em si, este texto não irá apresentar mais detalhes a esse respeito, indicando a leitura de (DE CASTRO, 2001), (DE CASTRO; TIMMIS, 2002c) e (AICKELIN; DASGUPTA, 2006) para um maior aprofundamento.

5.8 Uso de SIAs em Segurança Computacional

A idéia de utilizar SIAs em segurança computacional surge de forma relativamente precoce, durante a evolução dos estudos em SIAs. Obviamente, as preocupações eram distintas dos trabalhos sendo desenvolvido atualmente, sendo especialmente focados em combate a vírus de computadores. Muitos desses trabalhos possuíam um direcionamento vindo da área de pesquisa em Vida Artificial, como por exemplo (KEPHART, 1994), que apresenta uma proposta de sistema imune computacional para combate a vírus.

Uma visão geral do uso de SIAs para detecção de intrusos pode ser encontrada em (AICKELIN; GREENSMITH; TWYCROSS, 2004). Cabe destacar dois fatos observados por esses pesquisadores à época da publicação: i) todos os trabalhos avaliados por eles eram baseados em modelos com distinção *self* $\times$ *nonself*; ii) os trabalhos eram em sua grande maioria baseados em seleção negativa. Nenhum trabalho encontrado por esses pesquisadores era baseado em rede idiotípica e apenas um utilizava seleção clonal, além de seleção negativa. Obviamente, novas abordagens surgiram após a publicação desse artigo, mas não é difícil concordar com os autores que "IDSs imunoinspirados ainda têm muito espaço para crescimento e muitas áreas a serem exploradas".

Duas equipes tem-se destacado atualmente por uso de SIAs em detecção de intrusos, o que pode ser inferido inclusive da leitura de (KIM *et al.*, 2007). A primeira equipe compreende Stephanie Forrest e seus orientandos, em especial Steven A. Hofmeyr, durante o tempo que esteve realizando seu doutoramento. Os trabalhos dessa equipe envolvem principalmente algoritmos inspirados no mecanismo de seleção negativa de células T. De outro lado, destacam-se os esforços mais recentes de Uwe Aickelin, Jamie Twycross e Julie Greensmith. Existem outros pesquisadores envolvidos nessa tarefa, como Dipankar Dasgupta, mas essas equipes são as que mais avançaram nessa área, com um número significativo de trabalhos publicados sobre o assunto.

Entre os trabalhos desenvolvidos por Forrest e sua equipe, destaca-se (FORREST; HOFMEYR; SOMAYAJI, 1997), em que os autores apontam as vantagens de se abordar segurança computacional sob o ponto de vista de uma imunologia computacional, como um mecanismo distribuído de proteção. Essa visão é expandida em (FORREST; HOFMEYR, 2001), onde se propõe abordar Imunologia como um sistema de processamento de informações, uma analogia extremamente interessante em ambientes computacionais. Por fim, em (HOFMEYR, 1999) e (HOFMEYR; FORREST, 2000), é apresentado o uso da seleção negativa para detecção de intrusos, a partir de análise de tráfego de rede.

Entre as novas abordagens, destaca-se principalmente o uso do Modelo do Perigo. Para (AICKELIN *et al.*, 2003), inclusive, essa teoria seria o elo de ligação entre SIAs e IDSs. Como comentado nesse texto, o Modelo do Perigo proporciona uma metáfora biológica mais adequada para o IDS que a visão tradicional baseada em *self* e *nonself*. Uma informação importante apontada nesse e em alguns outros trabalhos é o fato que o princípio de seleção negativa tem apresentado problemas de escalonamento quando aplicado a um tráfego real de rede.

Apesar do Modelo do Perigo não se focar em imunidade inata, a equipe de Aickelin tem dado maior atenção a esse aspecto da teoria e só mais recentemente surgiram trabalhos envolvendo a imunidade adaptativa. A maior parte dos trabalhos dos pesquisadores dessa equipe são focados em algoritmos baseados no funcionamento das células dendríticas, similares aos apresentados na Figura 5.4 (Seção 5.5). Essa tendência pode ser verificada, por exemplo, em (GREENSMITH; AICKELIN; CAYZER, 2005), (GREENSMITH; AICKELIN; TWYCROSS, 2006) e (GREENSMITH; TWYCROSS; AICKELIN, 2006), que apresentam o DCA (*Dendritic Cell Algorithm*). Em (GREENSMITH; AICKELIN, 2007) são apresentados resultados da aplicação desse trabalho na detecção de varreduras SYN[2], obtendo resultados relativamente promissores.

Apesar da importância e potencial do DCA, existem algumas falhas metodológicas com os trabalhos utilizando esse algoritmo na detecção de varreduras SYN. Como observado em (GREENSMITH; AICKELIN, 2007), o objetivo era "detectar uma varredura SYN lançada de uma máquina vítima, onde o DCA foi usado para monitorar o comportamento dessa vítima". Do ponto de vista da Segurança Computacional, este tipo de detecção não possui grande rele-

[2]Essa varredura tem por objetivo a descoberta de serviços ativos na máquina-alvo, ver detalhes sobre o estabelecimento de conexões TCP/IP, em que pacotes do tipo SYN são usados, na seção 2.4.1.

vância, porque ou a máquina teria sido comprometido por um invasor ou um usuário normal estaria iniciando o ataque. No primeiro caso, não seria possível confiar nos alertas, porque os intrusos iriam modificar o funcionamento normal da máquina, objetivando esconder seu ataque. Na segunda situação, o ataque pode ser evitado com o uso de processos e ferramentas tradicionais de segurança, como definição de permissões de usuários e regras de *firewall*.

Em outras palavras: a maior parte dos trabalhos aplicando DCA em detecção de varreduras SYN somente detecta se uma máquina está atacando outras máquinas, não se uma máquina está sendo atacada. Na maior parte das situações reais, o ataque inicial é feito geralmente a partir de máquinas pessoais, externas ou infiltradas, onde esta abordagem não pode ser aplicada. Assim, uma contribuição muito mais adequada para Segurança Computacional é detectar se uma máquina está sofrendo um ataque de varreduras, uma das principais motivações deste trabalho.

Existem poucos trabalhos baseados no Modelo do Perigo que utilizam mecanismos adaptativos. Em (TEDESCO; TWYCROSS; AICKELIN, 2006), é apresentado um trabalho integrando conceitos da imunidade adaptativa, ainda em fase experimental. O algoritmo implementado utiliza células dendríticas e células T. Entretanto o artigo não deixa claro vários detalhes de sua implementação e nem apresenta uma descrição formal do mesmo. Os resultados iniciais obtidos por essa equipe foram promissores, entretanto, apresentaram alto índice de falsos positivos.

Em (AICKELIN; GREENSMITH, 2007), por sua vez, é apresentado o TLR, um algoritmo inspirado no funcionamento dos receptores do tipo Toll (*Toll Like Receptors*). Esse algoritmo novamente é baseado nos princípios da imunidade inata, mas já faz uso de células T, de uma forma relativamente tímida. Mesmo assim, como comentado pelos autores, o algoritmo é inerentemente complexo, tornando difícil sua análise teórica e experimental. Esse foi um problema percebido durante a realização deste trabalho, ao se utilizar abordagens mais completas do sistema imune em si: quanto mais completo e elaborado o modelo, mais complexa a avaliação de seus resultados. Uma abordagem mais promissora é apresentada em (FANELLI, 2008), baseado na interação entre células dendríticas e células T, obtendo melhores resultados que o Snort[3] ao avaliar os dados constantes no IDEVAL99 (LIPPMANN *et al.*, 2000).

A detecção de SPAM utilizando SIAs é uma tarefa ainda pouco explorada, podendo ser citados, além da pesquisa efetuada por nossa equipe, os traba-

[3]Snort: `http://www.snort.org/`.

lhos realizados por Terri Oda e Tony White. O primeiro modelo conhecido para classificação de mensagens de e-mail utilizando SIA foi proposto em (ODA; WHITE, 2003a) e estendido em (ODA; WHITE, 2003b). O algoritmo atribui um peso para cada detector (linfócito), que é incrementado quando ocorre o reconhecimento de um SPAM e decrementado em caso de mensagens legítimas. Aplicado a um conjunto de 1200 mensagens, conseguiu identificar 90% do SPAM e 99% das mensagens legítimas, após treinamento com 1600 mensagens de SPAM e 1200 mensagens legítimas.

Mais recentemente, tem-se a utilização de uma rede imune competitiva com essa finalidade, o que é apresentado em (BEZERRA *et al.*, 2006), com a participação de Leandro Nunes de Castro e Fernando Von Zuben. O modelo utilizado nesse trabalho permite a expansão clonal dos anticorpos mais estimulados e a morte daqueles menos estimulados. O tamanho da rede é ajustado dinamicamente, dependendo dos dados de treinamento, e a rede utiliza vetores reais para representar o peso das conexões entre anticorpos. Quando aplicado a um conjunto de 481 SPAM e 618 mensagens legítimas, produziu taxas de classificação entre 97% e 98%, tendo sido utilizado 90% das mensagens para treinamento e 10% para teste.

Em um trabalho paralelo a este, nossa equipe aplicou o IA-AIS, apresentado na Seção 6.3.1, no processo de deteção de SPAM (GUZELLA *et al.*, 2005; GUZELLA *et al.*, 2008). Foi possível obter altos índices de classificação, superiores a 99%, com diferentes balanceamentos entre falsos positivos e negativos, dependendo dos parâmetros utilizados. Testes comparativos com classificador bayesiano mostraram o uso de SIAs como uma excelente alternativa para aplicação nesse tipo de problema.

5.9 Comentários Finais

Este capítulo apresentou os Sistemas Imunes Artificiais (SIAs), uma abordagem imunoinspirada para resolução de problemas diversos. Como pode ser inferido a partir da leitura deste capítulo, esta é uma área com grande potencial de estudo e desenvolvimento. Contribuições recentes da Imunologia deverão ser continuamente integradas à área, o que lhe garantirá por um bom tempo sua renovação, através da proposta de novos modelos.

Além disso, buscou-se apresentar também neste capítulo uma visão geral do uso de SIAs aplicados à Segurança Computacional. Com isso, pretende-se mostrar ao leitor que, até o momento de escrita deste texto, essa é ainda uma

área com bastante potencial explorativo e para a qual ainda há muito a ser feito, o que é destacado em (KIM *et al.*, 2007). Recomenda-se, inclusive, a leitura desse artigo ao leitor interessado em uma visão relativamente detalhada da aplicação de SIAs em detecção de intrusos.

CAPÍTULO 6

Algoritmos e Modelos Propostos

O que é a verdade, portanto? Um batalhão móvel de metáforas, metonímias, antropormofismos, enfim uma soma de relações humanas, que foram enfatizadas poética e retoricamente (...).

F. Nietzsche, Sobre Verdade e Mentira no Sentido Extra-Moral

(...) um conceito possui um devir que concerne, desta vez, a sua relação com conceitos situados no mesmo plano. Aqui, os conceitos se acomodam uns aos outros, superpõem-se uns aos outros, coordenam seus contornos, compõem seus respectivos problemas, pertencem à mesma filosofia, mesmo se têm histórias diferentes

Deleuze & Guatari, O que é a Filosofia?

6.1 Comentários Iniciais

COMO já comentado ao longo deste texto, problemas de segurança computacional tem se tornado cada vez mais freqüentes e os ataques virtuais cada vez mais comuns. À parte disso, poucas ferramentas computacionais incorporam técnicas de aprendizado de máquina, que permitam o aprimoramento automático e uma resposta mais eficiente.

Tendo em vista o crescente interesse na área de Sistemas Imunes Artificiais (SIAs), bem como a existência de uma aparente ligação entre essa área e a

de Segurança Computacional, este trabalho propõe o desenvolvimento de metodologias para detecção de intrusos em redes de computadores. A relevância dessa pesquisa pode ser verificada em vários trabalhos sobre temas correlatos, discutidos no capítulo anterior. Especificadamente, são objetivos deste trabalho a formulação de modelos baseados em SIAs para desenvolvimento de ferramentas de detecção de intrusos em redes TCP/IP.

6.2 O Sistema Imune como Metáfora

Em (MAGRO; VAZ, 1991), Cristina Magro e Nelson Vaz fazem toda uma discussão sobre a visão do sistema imune como uma metáfora, tentando construir uma ponte entre Imunologia e Lingüística. Nesse texto, a visão de metáfora é principalmente a de um elemento que auxilia na compreensão, "profetizando" a si mesmo como conhecimento: "A essência da metáfora consiste, então, em compreender e experienciar um tipo de coisa em termos de outra, aproximando conceitos de espécies distintas". Mais à frente nesse mesmo trabalho, esses autores apontam:

> Mostramos que os anticorpos e moléculas Ti são metáforas moleculares: são profecias que persistem na medida em que confirmam a si mesmas; são invenções moleculares pelas quais o sistema imune atribui significado a diferentes materiais. Ao dar origem a linfócitos emergentes que reagem com um dado material estranho, o sistema imune atribui significado a este material, reconhece sua presença e torna o organismo apto a manipulá-lo pelos canais regulares de englobamento, desmontagem e reutilização de materiais macromoleculares.
>
> (MAGRO; VAZ, 1991, p.130)

Dessa maneira, dentro da área de Computação Imunoinspirada, é importante ter em mente que o sistema imune é utilizado como metáfora, e cada abordagem de uso de SIAs faz uso de uma metáfora específica. Assim, por exemplo, a maior parte dos trabalhos de aplicação de SIAs em IDS fazem uso de uma metáfora "guerreira" do sistema imune, defendendo o organismo (o computador) dos agentes invasores (os pacotes com tentativas de invasão). Essa abordagem, assim como no sistema imune natural, apresenta problemas, uma vez que não engloba em si uma visão mais sistêmica dos problemas de Segurança Computacional.

Um exemplo dos problemas advindos dessa visão é o fato que muitos pacotes por si só não caracterizam uma tentativa de invasão, mas podem ser utilizados para essa finalidade. O protocolo ICMP, como comentado no Capítulo 2, é utilizado para envio de mensagens de erro no TCP/IP, podendo ser utilizado para verificação de disponibilidade de pontos da rede, por exemplo. Entretanto, esse protocolo pode ser utilizado para varredura de redes (*scanning*), em busca da existência de vulnerabilidades, como apontado no Capítulo 3.

Esse mesmo problema ocorre com o estabelecimento de conexões TCP, através de pacotes do tipo SYN[1]. Pacotes desse tipo podem ser utilizados para varredura dos serviços disponíveis no computador, bem como para ataques de SYN *flooding*, em que vários pacotes desse tipo são encaminhados ao alvo, com o objetivo de sobrecarregá-lo, gerando negação de serviço (DOS). Se esse ataque vem de diferentes pontos da rede, inclusive, um DDOS, torna-se extremamente complexo, se não quase impossível, diferenciar pedidos de conexão legítimos daqueles originados pelo ataque[2]. Esse tipo de ataque também pode ser realizado com outros tipos de protocolos, por exemplo o ICMP, em que o envio de grande quantidade de pacotes, em geral de tamanho grande, pode induzir o travamento de máquinas com sistemas operacionais desatualizados.

Um outro problema da abordagem "guerreira" é que, em vários tipos de ataques, pacotes ou mensagens podem ser forjados para se passarem como conexões normais. Uma abordagem desse tipo tende a não tratar adequadamente essa situação, gerando um volume indesejado de falsos positivos ou falsos negativos. Dessa maneira, esse trabalho sustenta a hipótese que é necessário o uso de uma visão mais abrangente do sistema imune para a proposta de SIAs mais eficientes quando aplicados na tarefa de detecção de intrusão.

Assim, este trabalho pautou-se em buscar a proposta e uso de modelos que utilizassem metáforas alternativas ao sistema imune, especialmente aquelas com uma visão mais holística do processo. Obviamente, isso trouxe uma série de dificuldades, uma vez que a aplicação da metáfora não é trivial mesmo com uma metáfora considerada por nós mais "simples". Assim, houve uma grande intenção, desde sua etapa inicial, em utilizar abordagens mais completas, desenvolvendo algoritmos que apontassem para novas pesquisas em formação da memória imunológica ou incorporassem, por exemplo, elementos da imunidade inata e adaptativa.

[1] Ver detalhes na Seção 2.4.1.

[2] Esse foi o tipo de ataque descrito por nós na Seção 3.4, em que foi possível bloquear o acesso da rede do CPDEE, na UFMG, aos servidores da UFLA.

6.3 Algoritmos e Modelos Propostos

6.3.1 IA-AIS

O primeiro algoritmo utilizado neste projeto foi o IA-AIS (*Innate and Adaptive Artificial Immune System*), inspirado inicialmente no CLONALG, mas com a inclusão de macrófagos (representando a imunidade inata) e células T no sistema. Esse algoritmo apóia-se na distinção entre próprio e não-próprio, por questões de simplificação, e foi apresentado inicialmente em (GUZELLA *et al.*, 2005) e mais recentemente em (GUZELLA *et al.*, 2008)[3]. Em uma visão mais geral, as etapas desse algoritmo podem ser descritas como:

1. Cria-se uma população inicial de macrófagos, treinada para reconhecer padrões moleculares de patógenos, mas não do próprio organismo.

   Neste caso, os macrófagos computacionalmente correspondem a um conjunto de detectores, com um tempo de vida e *status* associado. Podem ser implementados preferencialmente como objetos ou estruturas (registros). No contexto de detecção de intrusos, os patógenos correspondem aos dados associados a tentativas de intrusão. O algoritmo recebe um conjunto de arquivos de treinamento, com elementos de pacotes ou conexões de redes, cada entrada associada ou a um uso normal do computador ou a uma tentativa de ataque. São selecionados então um número pré-estabelecido de macrófagos, capazes de reconhecer entradas classificadas anteriormente como ataque, mas com baixa afinidade com entradas classificadas como normais.

2. Cria-se uma população inicial de células B e células T auxiliares, selecionadas por um processo de seleção negativa (precisam reconhecer antígenos não-próprios, mas não antígenos do próprio organismo).

   Células B e T também são detectores, a única diferença para macrófagos é que elas podem ser clonados e sofrer processo de mutação em seus receptores. Um receptor nada mais é que um conjunto de padrões, tendo sido representado neste trabalho como uma sequência de caracteres. O processo de seleção é semelhante ao de macrófagos, com a possibilidade de uso de bases diferenciadas: enquanto macrófagos utilizam padrões moleculares, linfócitos utilizam antígenos não-próprios. No contexto de intrusão de redes de computadores, poderiam ser considerados padrões

[3] IA-AIS foi desenvolvido de forma cooperativa por este pesquisador e Thiago Guzella.

moleculares os dados que viessem de tipos de ataques bem conhecidos e para os quais não há possibilidade de dúvidas, por exemplo. Quaisquer outros dados de ataques seriam melhor tratados pelo sistema como antígenos não-próprios. Neste trabalho, entretanto, padrões moleculares de patógenos e antígenos não-próprios vieram do mesmo conjunto de dados, com objetivo de simplificação. De forma semelhante a macrófagos, só são selecionados os linfócitos incapazes de reconhecer as entradas classificadas anteriormente como normais.

Quando da geração dos receptures dos linfócitos e macrófagos, o algoritmo utiliza-se de "sementes", que são trechos selecionados aleatoriamente a partir de uma base de dados. Com isso, tem-se que o receptor é criado, propositalmente, para reconhecer um trecho da entrada. Como o receptor é menor que a entrada (em geral de 5% a 30%), há uma boa dose de aleatoriedade nessa escolha. Além disso, o processo de seleção negativa busca justamente garantir que esse trecho é característico das entradas anormais na base de dados. Uma abordagem alternativa seria a de gerar receptores aleatórios e calcular em seguida a afinidade. Essa abordagem entretanto é bastante ineficiente, do ponto de vista computacional, uma vez que o resultado obtido será muito próximo ao conseguido utilizando-se de "sementes" para os receptores, mas exigindo maior tempo de processamento.

3. Cria-se uma população inicial de células T regulatórias, capazes de reconhecer elementos do próprio organismo.

   Nesse caso, tem-se uma inversão: são criados detectores capazes de reconhecer antígenos próprios, que computacionalmente correspondem a dados advindos de pacotes componentes de uso comum do computador. Assim, nessa etapa, tem-se a criação de detectores capazes de reconhecer pacotes de redes semelhantes aos que foram antes classificados como normais. Esse passo é a última etapa do treinamento do sistema, sendo seguido pela classificação dos dados de teste.

4. Após a etapa inicial de treinamento, para cada microorganismo M a ser avaliado, é feita sua classificação, nos passos a seguir.

   Após o treinamento, o algoritmo começa o processo de classificação, utilizando-se para isso de uma base de dados para avaliação. Essa base também consiste em um conjunto de pacotes de redes diversos, e cada entrada é tratada pelo algoritmo como um "microorganismo", um conjunto de antígenos. Cada entrada da base de avaliação é então avaliada

pelos sistemas imune inato e adaptativo e classificada de acordo com a "ligação" dessa entrada às celulas do algoritmo.

5. Apresenta-se M, que pode ser ligado por macrófagos e por linfócitos, à população de macrófagos.

   O primeiro elemento classificador do sistema é composto pelos macrófagos, sendo calculada a afinidade da entrada com os receptores desses tipos celulares. O cálculo da função de afinidade é feita à semelhança de algoritmos genéticos ou outras abordagens de SIAs: cada tipo de problema exige uma função de afinidade própria, que pode depender, inclusive, do formato dos dados de entrada. Por "ser ligado por macrófago", entende-se que o valor função de ativação naquele macrófago foi superior ao seu limiar de ativação, um dos parâmetros do algoritmo.

6. Se algum macrófago for ativado, é iniciada uma resposta imune, com a eliminação do patógeno e estimulação ou indução de linfócitos B e T, indo para o passo 12. Do contrário, continua-se a apresentação, no passo 7.

   Após o cálculo da afinidade, verifica-se quais macrófagos tiveram valores dessa função superiores ao seu limiar de ativação. Caso algum macrófago esteja nessa situação, então a entrada é classificada como uma tentativa de intrusão clássica (um PAMP). Além disso, são geradas linfócitos B e T capazes de reconhecer essa entrada, utilizando a própria entrada como "semente" para a geração dos receptores desses linfócitos. Nesse caso, o processo de classificação da entrada é terminado, indo para o próximo microorganismo. Caso nenhum macrófago "ligue-se" ao microorganismo, então isso implica que é necessária uma avaliação mais detalhada do mesmo, com as células do sistema imune adaptativo.

7. Apresenta-se M, como um grupo de antígenos Ags, à população de células B. Se nenhuma célula B for estimulada, uma resposta imune não é iniciada, e a apresentação da entrada é finalizada. Senão, continua-se no próximo passo.

   De forma semelhante aos macrófagos, calcula-se a afinidade da entrada (o "microorganismo") com cada linfócito B do sistema. Caso nenhum linfócito "ligue-se" a essa entrada, isso implica que a mesma não faz parte de uma tentativa de ataque, sendo classificada pelo sistema como normal. Também nesse caso, o processo de classificação é terminado, retomando-se o passo 12 com o próximo microorganismo. Caso algum linfócito B seja ativado, entretanto, a entrada é analisada pelos linfócitos T, para confirmação ou não dessa "ligação".

8. Cada linfócito B estimulado processa o grupo de antígenos Ags e os apresenta à população de células T. Se algum linfócito T for capaz de se ligar à algum antígeno Ag pertencente ao conjunto Ags, a célula B apresentadora recebe um segundo sinal, estimulatório ou regulatório, dependendo do tipo de célula T ativada. Caso a célula B receba mais sinais estimulatórios que regulatórios, ela é finalmente ativada, caso contrário, ela será suprimida no passo 11. É formada uma sub-população B_a de células B ativadas, que serão reproduzidas, nos passos a seguir.

   Nessa etapa, tem-se a avaliação da entrada pelos linfócitos T, auxiliares e regulatórios. Do somatório das respostas desses linfócitos, tem-se a ativação ou supressão das células B estimuladas no passo anterior. É importante observar que apenas na presença do segundo sinal, haverá estimulação da célula B. Ou seja: caso nenhum linfócito T reconheça o antígeno, então a célula B é suprimida. Caso pretenda-se um sistema mais reativo, o algoritmo poderia ser modificado para não reagir apenas quando os sinais regulatórios forem mais intensos que os sinais estimulatórios. As células B ativadas são clonadas e sofrem de hipermutação, nos passos a seguir.

9. Para cada um dos N linfócitos da população B_a com maior afinidade aos antígenos apresentados, são gerados n clones, segundo a equação a seguir:

$$n = \mu exp(-(1 - afinidade)) \tag{6.1}$$

   sendo μ o número máximo de clones a serem gerados por célula B ativada e $exp()$ a função exponencial.

   As células B ativadas no passo anterior são clonadas, numa proporção direta à sua afinidade ao antígeno sendo analisado: quanto maior sua afinidade, mais clones são gerados. O sistema como um todo é calibrado por um parâmetro (μ) que define o número máximo de clones permitidos por célula.

10. Os clones gerados são submetidos a hipermutação somática de seus receptores, com probabilidade inversamente proporcional à afinidade pelo antígeno que estimulou o linfócito original, gerando uma população de clones C. A taxa de mutação da cadeia, representada pela probabilidade de mutação de cada caracter, é dada por:

$$T_{mutacao} = 1 - exp(-\alpha(1 - afinidade)) \tag{6.2}$$

onde α é um valor positivo. Os clones gerados são apresentados ao mesmo antígeno que levou à ativação do linfócito original. Os N_c clones com maiores afinidades são adicionados à população de células B do sistema.

Nesse passo, os clones gerados anteriormente sofrem hipermutação somática, calibrada pelo parâmetro α. Após isso, a entrada que ativou inicialmente as células B são avaliadas por esses clones, calculando-se a afinidade. Os clones são então ordenados de acordo com essa afinidade e selecionados, de acordo com o parâmetro N_c, que indica o número máximo de clones permitidos por iteração.

11. Se nenhuma célula B tiver recebido suficiente segundo sinal estimulatório, uma resposta imune não é iniciada e a apresentação é finalizada.

    Caso a célula B receba menos sinais estimulatórios que regulatórios, ela é suprimida, ou seja: seu TTL é zerado, fazendo com que a mesma seja eliminada do sistema ao término da iteração. Nesse caso, a entrada é classificada como pertencente a um uso normal do computador.

12. Decrementa-se o TTL de todas as células do sistema. Adiciona-se algumas novas células (macrófagos e linfócitos), como forma de manter a população com tamanho relativamente estável. Continua-se o processo de classificação, com o próximo microorganismo M, no passo 5. Caso, todos os microorganismos tenham sido apresentados, o algoritmo é encerrado.

    Terminada a apresentação da entrada, o sistema é preparado para uma nova iteração, decrementando-se o TTL de todas as células ainda vivas, sendo eliminadas aquelas com TTL zerado. Novas células são geradas, de forma semelhante aos passos 1, 2 e 3, mas em quantidade bem menor. O objetivo aqui é permitir uma manutenção no número de células no sistema, bem como permitir aprendizagem dinâmica. Como recurso adicional, os linfócitos B e T auxiliares são gerados utilizando "sementes" aleatórias para o não-próprio, ampliando o espectro de atuação do sistema, tornando-o capaz de se ligar a novos padrões. Após a finalização da iteração, caso ainda existam entradas a serem avaliadas, o processo de classificação é retomado, no passo 5.

Uma visão sintética do algoritmo pode ser verificada na Figura 6.1, que mostra uma visão geral do mesmo, na Figura 6.2, que apresenta a etapa inicial de treinamento, e na Figura 6.3, que apresenta o processo de classificação de um dado microorganismo. Uma observação importante é que, quando da preparação de novas iterações, novas células são adicionadas ao sistema. Esse

fato, somado ao processo de clonagem e hipermutação durante a classificação faz com que o IA-AIS possua aprendizagem dinâmica, ou seja: o sistema vai "aprendendo", à medida que classifica novos elementos.

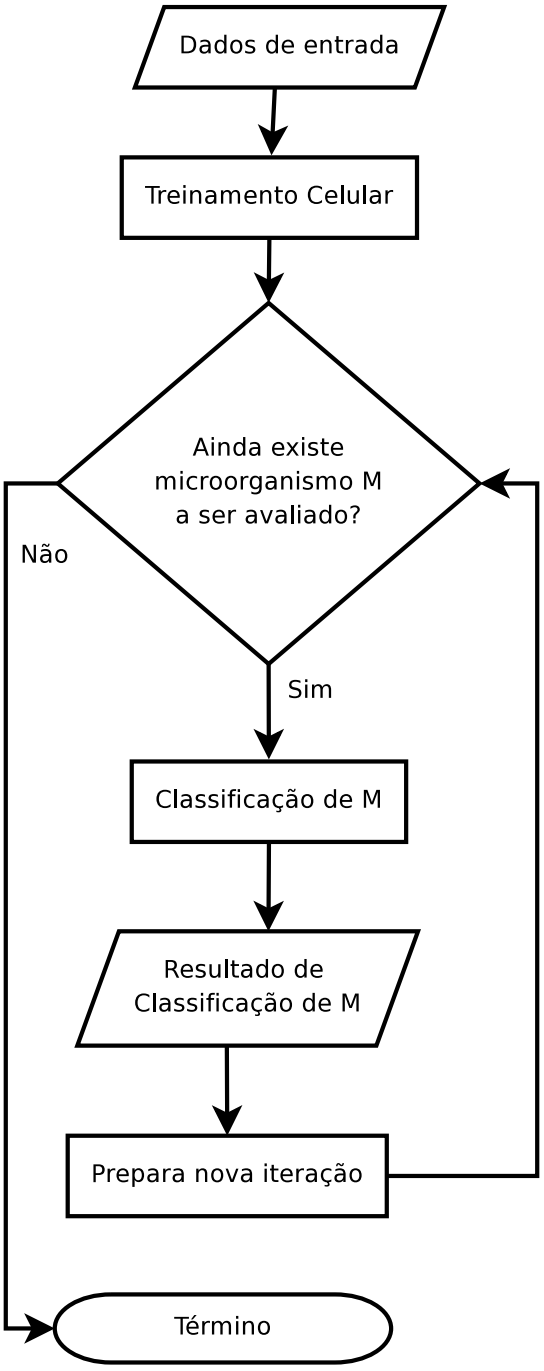


Figura 6.1: IA-AIS – Visão Geral

Apesar de inspirado inicialmente no CLONALG, existem várias diferenças entre esse algoritmo e o IA-AIS, além do uso de mecanismos da imunidade inata. O CLONALG faz uso de uma população específica de memória: todas as células presentes nessa população são, de fato, células de memória. No mo-

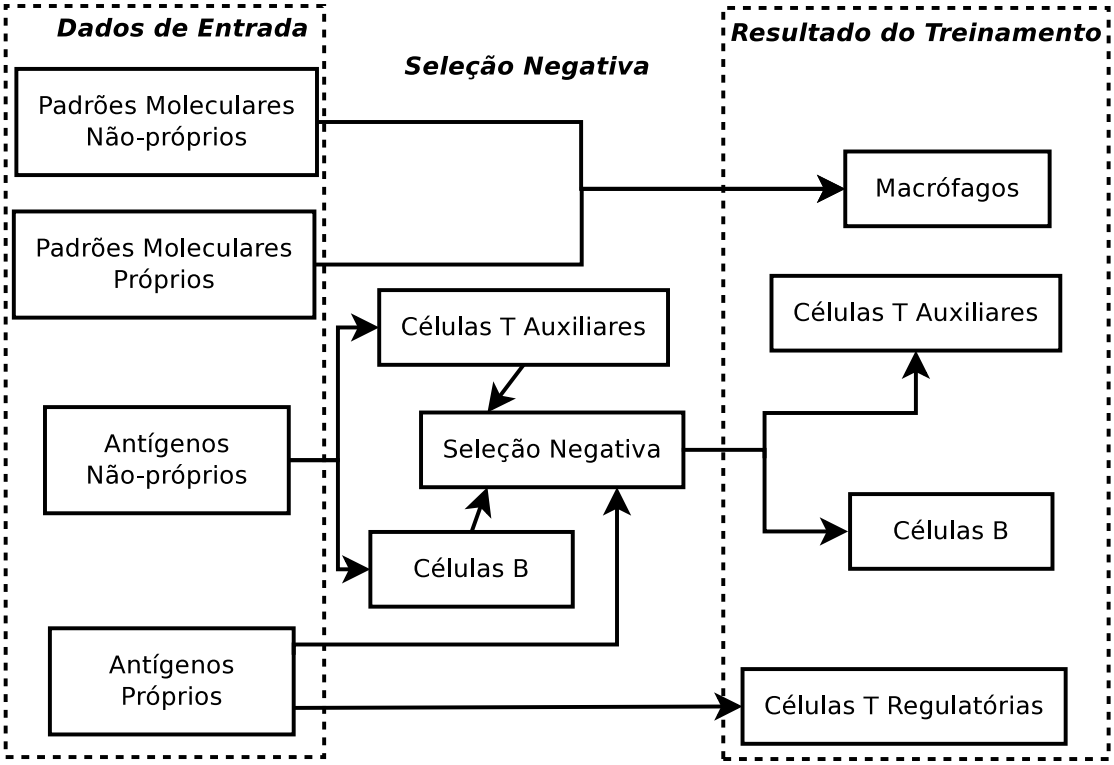


Figura 6.2: IA-AIS – Treinamento

delo proposto, não existe tal população, usando-se uma abordagem diferente: quando uma célula é criada, define-se um tempo de vida, um valor inteiro positivo representando a contagem regressiva para a morte da célula. Esse valor é decrementado após cada vez que um microorganismo é apresentado ao sistema, com a eliminação de células com um valor nulo de tempo de vida.

Para simular a competição pelo reconhecimento de patógenos, um *bônus* de tempo de vida é usado; quando ativada, uma célula terá o seu tempo de vida incrementado por esse valor, garantindo que células com um elevado número de ativações sejam mantidas, e que células pouco estimuladas sejam substituídas. Desse modo, a característica de "memória" de uma célula é intrínseca: uma célula pode ser considerada como de memória se possuir um tempo de vida elevado e/ou um grande número de ativações.

Como forma de facilitar a implementação do IA-AIS, a Figura 6.4 apresenta seu algoritmo em formato de pseudocódigo, detalhando principalmente a etapa de classificação. O algoritmo começa com a obtenção dos dados de treinamento e de teste, com a função GETTRAININGANDTESTDATA( ), detalhada na Figura 6.5. Essa função recebe dois arquivos de entrada, um de treinamento e um de avaliação e faz o tratamento inicial desses arquivos, especialmente o de treinamento.

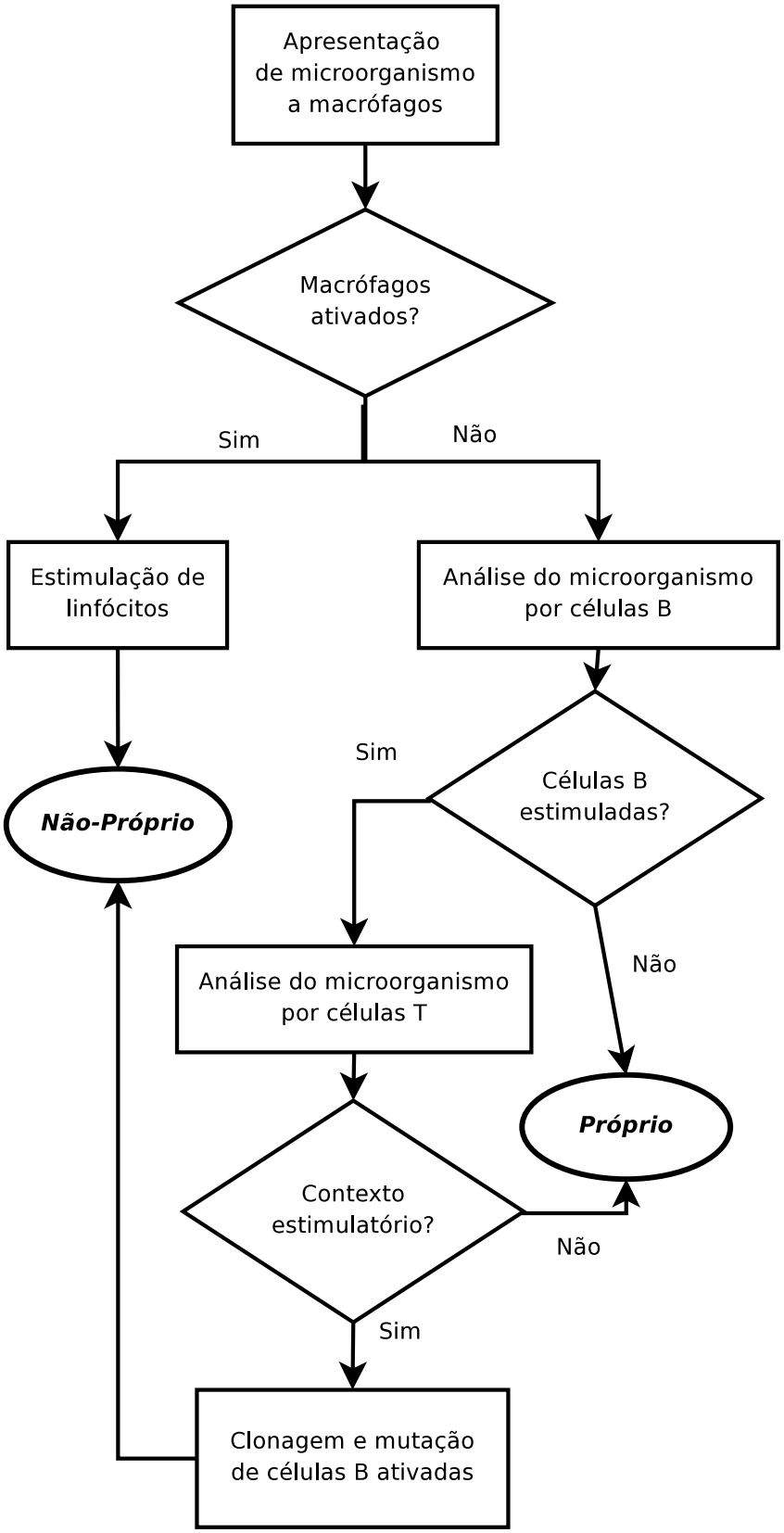


Figura 6.3: IA-AIS – Classificação

```
 1: getTrainingAndTestData(trainingFiles,testFiles)          ▷ [Preparation Phase]
 2:
 3: trainImmuneCells(pampSeeds, nonselfSeeds, selfSeeds)     ▷ [Training Phase]
 4:
 5: microorganisms ← extractAntigenPatterns(testData)        ▷ [Classification Phase]
 6: for all mic in microorganisms do
 7:     macsBinding ← presentToCellPopulation(mic, macrophages)
 8:     if size(macsBinding) > 0 then      ▷ microorganism binded by macrophages
 9:         classifyMicroorganism(mic, PAMP)
10:         for all binding in macsBinding do
11:             activateImmuneCell(binding.cell)
12:         end for
13:         estimulateLymphocytes(mic, selfSeeds)
14:     else                              ▷ microorganism not binded to macrophages
15:         bcsBinding ← presentToCellPopulation(mic, bcells)
16:         if size(bcsBinding) > 0 then          ▷ microorganism binded by B Cells
17:             tauxsBinding ← presentToCellPopulation(mic, tauxs)
18:             tregsBinding ← presentToCellPopulation(mic, tregs)
19:             tauxImpact ← evaluateBinding(tauxsBinding)
20:             tregImpact ← evaluateBinding(tregsBinding)
21:             if tauxImpact > tregImpact then          ▷ context is stimulating
22:                 classifyMicroorganism(mic, NONSELF)
23:                 allClones ← [ ]
24:                 for all binding in bcsBinding do          ▷ clone binded B Cells
25:                     activateImmuneCell(binding.cell)
26:                     clones ← cloneCell(binding.cell, binding.rate)
27:                     hypermutate(clones)
28:                     append(clones, allClones)
29:                 end for
30:                 clonesBinding ← presentToCellPopulation(mic, allClones)
31:                 clonesSelected ← selectBestClones(clonesBinding)
32:                 addToCellPopulation(clonesSelected, bcells)
33:             else          ▷ context is supressing, supress correspondig B Cell
34:                 classifyMicroorganism(mic, SELF)
35:                 for all binding in bcsBindind do
36:                     supressImmuneCell(binding.cell)
37:                 end for
38:             end if
39:         else                          ▷ not binded by immune system: self
40:             classifyMicroorganism(mic, SELF)
41:         end if
42:     end if
43:     updateAllCells( )                                  ▷ [Preparing new iteration]
44:     addNewImmuneCells(pampSeeds, nonselfSeeds selfSeeds)
45: end for
```

Figura 6.4: IA-AIS (Pseudocódigo)

O tratamento efetuado no arquivo de treinamento destina-se principalmente a obter em cada linha do arquivo o que será considerado como antígeno para o sistema, através das funções GETSELFSEEDS( ), GETNONSELFSEEDS( ) e GETPAMPSEEDS( ). Obviamente, cada base de dados exigirá um tratamento específico, mas relativamente simples, adequando as entradas ao algoritmo e à função de afinidade. O tratamento do arquivo de avaliação ou teste é feito de maneira similar pela função EXTRACTANTIGENPATTERNS( ), que cria uma lista de microorganismos (*microorganisms*), linha 5 do pseudocódigo apresentado na Figura 6.4. A variável *microorganisms*, portanto, é apenas um conjunto de antígenos a serem classificados pelo algoritmo.

```
1: function GETTRAININGANDTESTDATA(trainingFiles,testFiles)
2:     trainingData ← GETTRAININGDATA(trainingFiles)
3:     testData ← GETTESTDATA(testFiles)
4:     selfSeeds ← GETSELFSEEDS(trainingData)
5:     nonselfSeeds ← GETNONSELFSEEDS(trainingData)
6:     pampSeeds ← GETPAMPSEEDS(trainingData)
7:     return selfSeeds, nonselfSeeds, pampSeeds, testData
8: end function
```

Figura 6.5: Função GETTRAININGANDTESTDATA( )

A função TRAINIMMUNECELLS( ), por sua vez, utiliza parte dos dados processados pela GETTRAININGANDTESTDATA( ) e cria a população inicial de células do sistema. Essa função é detalhada na Figura 6.6, onde pode-se ver que ela faz chamada a duas outras funções: GENMACROPHAGES( ), que gera a população de macrófagos, e GENLYMPHOCYTES( ), que gera a populações de linfócitos. A função GENMACROPHAGES( ) recebe, além dos dados extraídos dos arquivos de treinamento (*pampSeeds* e *selfSeeds*), dois parâmetros gerais do sistema: o tamanho inicial da população de macrófagos (`startMac`), bem como o tempo inicial de vida dos mesmos (`macTTL`). Parâmetros similares existem para as células B (`startBC` e `bTTL`), células T auxiliares (`startTAux` e `taTTL`) e células T regulatórias (`startTR` e `trTTL`). É importante observar que, apesar do algoritmo permitir distinguir entre padrões moleculares de antígenos e antígenos não-próprios (*pampSeeds* e *nonselfSeeds*), neste trabalho assumiu-se que esses dados seriam os mesmos, como comentado anteriormente.

As funções GENMACROPHAGES( ) e GENLYMPHOCYTES( ) são na verdade são chamadas a uma função mais geral, chamada GENERATEIMMUNECELLS( ), como mostrado na Figura 6.7. Optou-se por essa abordagem, pois isso fa-

```
1: function trainImmuneCells(pampSeeds, nonselfSeeds, selfSeeds)
2:     macrophages ← genMacrophages(startMac, macTTL, pampSeeds, selfSeeds)
3:     tauxs ← genLymphocytes(TAUX, startTAux, taTTL, nonselfSeeds, selfSeeds)
4:     bcells ← genLymphocytes(BCELL, startBC, bTTL nonselfSeeds, selfSeeds)
5:     tregs ← genLymphocytes(TREG, startTR, trTTL, NONE, selfSeeds)
6:     return macrophages, bcells, tauxs, tregs
7: end function
```

Figura 6.6: Função trainImmuneCells( )

cilitará o desenvolvimento de testes futuros, em que a criação possa ser feita de forma diferenciada. A função generateImmuneCells( ), apresentada na Figura 6.8, cria uma lista de células imunes, de um tipo especificado (*cellType*) e com um tempo de vida inicial pré-definido *startTTL*, a partir de dados de treinamento (*nSeeds* e *sSeeds*). Em alguns casos, é possível criar tipos celulares com receptores aleatórios, bastando para isso que não sejam passados dados relativos ao "não-próprio" (*nSeeds*). Esse é o caso da geração das células T regulatórias na função trainImmuneCells( ).

```
1: function genMacrophages(numMac, macTTL, pSeeds, sSeeds)
2:     return generateImmuneCells(MAC, numMac, macTTL, pSeeds, sSeeds)
3: end function
4:
5: function genLymphocytes(type, numCells, startTTL, nsSeeds, sSeeds)
6:     return generateImmuneCells(type, numCells, startTTL, nsSeeds, sSeeds)
7: end function
```

Figura 6.7: Implementação de Funções de Geração de Células Imunes no IA-AIS

Em caso de geração com "sementes" do não-próprio, aleatoriamente é escolhida um trecho de uma entre as várias entradas pré-processadas. Esse trecho é transformado então no receptor da célula sendo criada pela função createImmuneCell( ). Essa função, por sua vez, tem o objetivo de criar uma estrutura ou objeto do tipo célula, que possui ao menos quatro campos, a saber: i) seu tipo celular, para uso em verificações; ii) seu receptor, principalmente para cálculo de afinidade; iii) o tempo de vida inicial; iv) o *status* da célula.

Após a criação de uma célula, a mesma passa por um processo de seleção negativa, recebendo as sementes do "próprio", sendo descartadas as células cuja afinidade for superior ao seu limiar de ativação. Esse limiar é

```
 1: function GENERATEIMMUNECELLS(cellType, numCells, startTTL, nSeeds, sSeeds)
 2:     immuneCells ← [ ]
 3:     num ← 0
 4:     while num < numCells do
 5:         if nSeeds ≠ NONE then                          ▷ segment selection/creation
 6:             nseed ← RANDOMCHOOSESEED(nSeeds)
 7:             segment ← RANDOMCHOOSESEGMENT(nseed)
 8:         else
 9:             segment ← GENERATERANDOMSEGMENT( )
10:         end ifCREATEIMMUNECELL(type, receptor, startTTL, NAIVE)
11:         imCell.receptor ← segment                      ▷ create immune cell
12:         imCell.type ← cellType
13:         imCell.TTL ← startTTL
14:         imCell.status ← NAIVE
15:         maxBinding ← 0                                 ▷ negative selection
16:         for all seed in sSeeds do
17:             binding ← AFFINITY(sSeeds, imCell)
18:             if binding > maxBinding then
19:                 maxBinding ← binding
20:             end if
21:         end for
22:         if maxBinding < activationThreshold[type] then
23:             APPEND(imCell, immuneCells)
24:             num ← num + 1
25:         end if
26:     end while
27:     return immuneCells
28: end function
```

Figura 6.8: Função GENERATEIMMUNECELLS( )

definido para cada tipo celular (ACTIVATIONTHRESHOLD[*type*]) como um parâmetro geral do algoritmo. Ao término de seu processamento, a função GENERATEIMMUNECELLS( ) irá devolver uma população de células do tipo solicitado.

Após o treinamento inicial do sistema, através do uso de receptores induzidos e seleção negativa, o IA-AIS inicia o processo de classificação dos dados de avaliação/teste, a partir da linha 5 do pseudocódigo apresentado na Figura 6.4. Depois do carregamentos dos dados a serem classificados, apresenta-se cada entrada de teste à população de macrófagos, através da função PRESENTTOCELLPOPULATION( ), detalhada na Figura 6.9. Essa função, que recebe um microorganismo e uma população celular, inicialmente verifica o tipo de população celular ao qual o microorganismo em questão será apresentado. Em seguida, calcula a afinidade desse microorganismo com todas as

células dessa população, verificando e retornando aquelas que tiveram esse valor acima do limiar pré-estabelecido em ACTIVATIONTHRESHOLD[*type*]).

```
 1: function PRESENTTOCELLPOPULATION(mic,cellPopulation)
 2:     bindings ← [ ]
 3:     type ← GETTYPE(cellPopulation)
 4:     for all imCell in cellPopulation do
 5:         taxAff ← AFFINITY(mic, imCell)
 6:         if taxAff ≥ activationThreshold[type] then
 7:             bindInfo.rate ← taxAff
 8:             bindInfo.cell ← imCell
 9:             APPEND(bindInfo, bindings)
10:         end if
11:     end for
12:     return bindings
13: end function
```

Figura 6.9: Função PRESENTTOCELLPOPULATION( )

Seguindo-se a linha 8 na Figura 6.4, caso algum macrófago ligue-se à entrada sendo avaliada, essa entrada é classificada como uma tentativa de intrusão, através da função CLASSIFYMICROORGANISM( ). Essa função tem o objetivo de sumarizar o processo de classificação, apontando, por exemplo, falsos positivos e negativos. Na sequência, as células ligadas são ativadas, através da função ACTIVATEIMMUNECELL( ), Figura 6.10, que consiste basicamente em incremento do tempo de vida e mudança do estado da célula. Em seguida, linha 13 da Figura 6.4, há a geração de células B e T capazes de reconhecer o microorganismo, através da função ESTIMULATELYMPHOCYTES( ), Figura 6.11. Essa função produz novas linfócitos, cujos receptores são produzidos a partir de trechos do segmento associado ao microorganismo.

```
 1: function ACTIVATEIMMUNECELL(cell)
 2:     cell.status ← ACTIVE
 3:     cell.TTL ← cell.TTL + ttlBonus[cell.type]
 4: end function
 5:
 6: function SUPRESSIMMUNECELL(cell)
 7:     cell.status ← SUPRESSED
 8:     cell.TTL ← 0
 9: end function
```

Figura 6.10: Funções ACTIVATEIMMUNECELL( ) e SUPRESSIMMUNECELL( )

```
1: function ESTIMULATELYMPHOCYTES( nsSeeds, sSeeds)
2:     nTAuxs ← GENLYMPHOCYTES(TAUX, estimTAux, taTTL, nsSeeds, sSeeds)
3:     nBCells ← GENLYMPHOCYTES(BCELL, estimBC, bTTL nsSeeds, sSeeds)
4:     APPEND(nTAux, tauxs)
5:     APPEND(nBCells, bcells)
6: end function
```

Figura 6.11: Função ESTIMULATELYMPHOCYTES( )

Caso o microorganismo não seja reconhecido por um macrófago, então o mesmo é apresentado aos linfócitos B do sistema, linha 15 da Figura 6.4, utilizando a função PRESENTTOCELLPOPULATION( ), já descrita anteriormente. Caso ocorra a ligação do microorganismo a algum linfócito B, então esse microorganismo é apresentado também às células T do sistema, linhas 17 e 18 da Figura 6.4 . Após essa segunda apresentação, é feita uma avaliação sobre qual tipo de linfócito T foi mais ativado nesse processo, utilizando, para isso, a função EVALUATEBINDING( ), Figura 6.12, que sumariza as taxas de afinidade dos linfócitos ativados.

```
1: function EVALUATEBINDING(mic,bindings)
2:     totalBinding ← 0
3:     for all bindInfo in bindings do
4:         totalBinding ← totalBinding + bindInfo.rate
5:     end for
6:     return totalBinding
7: end function
```

Figura 6.12: IA-AIS – Função EVALUATEBINDING( )

Se o contexto de ligação aos linfócitos T for estimulatório, havendo mais ligação com células auxiliares que regulatórias, então é dado início a um processo de clonagem das células B, através da função CLONECELL( ), Figura 6.13. O número de clones gerados é diretamente proporcional à taxa de ligação do microorganismo à célula B sendo clonada. Além disso, esse número é regulado pelo parâmetro μ do algoritmo, que indica o número máximo de clones gerados por cada célula. Nesse caso ainda, o microorganismo é classificado como um elemento invasor, uma tentativa de intrusão.

Em sequência ao processo de clonagem, os clones gerados sofrem um processo de hipermutação, através da função HYPERMUTATE( ), Figura 6.14. Nesse caso, a taxa é inversamente proporcional à taxa de ligação do microorganismo

à célula B sendo clonada e é regulada pelo parâmetro α do algoritmo, um número positivo que é utilizado para aumentar ou diminuir a probabilidade de mutação no sistema como um todo. Após o término do processo de clonagem e hipermutação em todas as células B do sistema, há uma seleção dos melhores clones, através da função SELECTBESTCLONES( ), Figura 6.15. Essa função ordena os clones de acordo com a taxa de afinididade desses com o microorganismo. Após essa ordenação, são selecionados os N_c clones com maior afinidade, sendo que N_c é um parâmetro geral do algoritmo, indicando o número máximo de clones permitido por iteração.

```
1: function CLONECELL(cell, rate)
2:     numClones ← μ× exp(- (1 - rate))
3:     cellClones ← [ ]
4:     for i=1 to numClones do
5:         clone ← COPYCELL(cell)
6:         APPEND(clone, cellClones)
7:     end for
8:     return cellClones
9: end function
```

Figura 6.13: Função CLONECELL( )

```
1: function HYPERMUTATE(clones,rate)
2:     mutationTax ← 1 - exp(- α×(1 - rate))
3:     for all cell in clones do
4:         for all position in cell.receptor do
5:             prob ← RANDOM( )
6:             if prob ≤ mutationTax then
7:                 RANDOMVALUECHANGE(cell.receptor, position)
8:             end if
9:         end for
10:    end for
11: end function
```

Figura 6.14: Função HYPERMUTATE( )

Caso mais células T regulatórias que auxiliares se liguem ao microorganismo, linha 33 da Figura 6.4, então o ambiente é supressor e as células B que se ligaram são suprimidas, por meio da função SUPRESSIMMUNECELL( ), figura 6.10. Além disso, o microorganismo é classificado como uma conexão normal, o que também ocorre caso o microorganismo não se ligue à nenhuma célula do sistema. Por fim, há uma atualização das células do sistema, através da eliminação das células suprimidas e decremento do TTL de cada célula

```
 1: function SELECTBESTCLONES(clonesBinding)
 2:     if sizeof(clonesBinding) > N_c then
 3:         ORDERBYRATE(clonesBinding)
 4:         selBinding ← clonesBinding[1 : N_c]
 5:     else
 6:         selBinding ← clonesBinding
 7:     end if
 8:     clones ← [ ]
 9:     for all binding in selBinding do
10:         APPEND(binding.cell, clones)
11:     end for
12:     return clones
13: end function
```

Figura 6.15: IA-AIS – Função SELECTBESTCLONES( )

restante. Em seguida, novas células são adicionadas, por meio da função ADDNEWIMMUNECELLS( ), Figura 6.16. Essa função possui funcionamento similar à TRAINIMMUNECELLS( ), Figura 6.6, diferindo apenas nos parâmetros gerais do algoritmo que são utilizados, no caso o número de células a serem adicionadas a cada iteração (`iterMac`, `iterTAux`, `iterBC` e `iterTR`).

```
 1: function ADDNEWIMMUNECELLS(pSeeds, nsSeeds, sSeeds)
 2:     nMacs ← GENMACROPHAGES(iterMac, macTTL, pSeeds, sSeeds)
 3:     nTAuxs ← GENLYMPHOCYTES(TAUX, iterTAux, taTTL, nsSeeds, sSeeds)
 4:     nBCells ← GENLYMPHOCYTES(BCELL, iterBC, bTTL nsSeeds, sSeeds)
 5:     nTRegs ← GENLYMPHOCYTES(TREG, iterTR, trTTL, NONE, sSeeds)
 6:     APPEND(nMacs, macs)
 7:     APPEND(nTAux, tauxs)
 8:     APPEND(nBCells, bcells)
 9:     APPEND(nTRegs, tregs)
10: end function
```

Figura 6.16: Função ADDNEWIMMUNECELLS( )

Uma das grandes potencialidades do IA-AIS é sua capacidade para incorporar, com relativa facilidade, outros elementos do sistema imune. Em (GUZELLA *et al.*, 2008), por exemplo, a implementação recebeu o acréscimo de células T regulatórias, como forma de manutenção do *self*, o que não existia originalmente. Esse potencial do IA-AIS, entretanto, vem com uma complicação adicional: o uso do algoritmo não é simples, uma vez que exige a definição de uma série de elementos (macrófagos, células T, células B, etc.). Além disso, ele exige a definição de grande número de parâmetros, alguns correlacionados,

por exemplo o tempo de vida médio das células B. Isso dificulta em muito uma análise de seus resultados, bem como a criação de heurísticas para determinação de valores adequados a esses parâmetros. Com o objetivo de melhor contextualizar o leitor quanto a essa situação, na Tabela 6.1 são apresentados os principais parâmetros utilizados pelo IA-AIS.

Tabela 6.1: Parâmetros do IA-AIS

Parâmetro	**Definição**
α	Coeficiente de mutação
μ	Número máximo de clones por célula
N_c	Número máximo de clones por iteração
`startMac`	Número inicial de macrófagos
`startBC`	Número inicial de células B
`startTAux`	Número inicial de células T Auxiliares
`startTR`	Número inicial de células T Regulatórias
`macTTL`	TTL inicial de cada macrófago
`bTTL`	TTL inicial de cada célula B
`taTTL`	TTL inicial de cada célula T Auxiliar
`trTTL`	TTL inicial de cada célula T Regulatória
`ttlBonus[MAC]`	TTL adicional para macrófagos ativados
`ttlBonus[BCELL]`	TTL adicional para células B ativadas
`estimBC`	Número de células B adicionadas por estimulação de macrófagos
`estimTAux`	Número de células T Auxiliares adicionadas por estimulação de macrófagos
`iterMac`	Número de macrófagos adicionados a cada iteração
`iterBC`	Número de células B adicionadas a cada iteração
`iterTAux`	Número de células T Auxiliares adicionadas a cada iteração
`iterTR`	Número de células T Regulatórias adicionadas a cada iteração
`activationThreshold[MAC]`	Limiar de ativação para macrófagos
`activationThreshold[BCELL]`	Limiar de ativação para células B
`activationThreshold[TA]`	Limiar de ativação para células T Auxiliares
`activationThreshold[TREG]`	Limiar de ativação para células T Regulatórias

6.3.2 DT-AIS

Como comentado na Seção 4.4, a *Danger Theory*, apesar de não propiciar uma visão holística do sistema imune, possui vários méritos. Além disso, enquanto metáfora inspiradora, apresenta-se como extremamente adequada para o desenvolvimento de SIAs para detecção de falhas e anomalias, o que é também defendido em (AICKELIN *et al.*, 2003). Mesmo, inclusive, com visões mais amplas do sistema imune, essa abordagem parece ser bastante adequada nesse contexto, e isso deve prevalecer por mais algum tempo ainda.

Nesse sentido, cabe destacar que só recentemente foi sugerido na literatura (TEDESCO; TWYCROSS; AICKELIN, 2006) o uso de mecanismos inatos e adaptativos do sistema imune, baseados nessa teoria. O conhecimento de um algoritmo mais completo do sistema imune (Figura 4.4 na página 102), utilizado por Matzinger para descrever o seu funcionamento motivou o desenvolvimento de um algoritmo baseado na *Danger Theory*. Esse algoritmo também é baseado no uso de mecanismos inatos e adaptativos do sistema imune, para uso em detecção de falhas. As etapas desse algoritmo, denominado DT-AIS (*Danger Theory Artificial Immune System*), podem ser descritas da seguinte forma:

1. Cria-se uma população inicial de APCs e células T virgens, a partir de um conjunto de antígenos escolhidos para treinamento.

   De forma semelhante ao IA-AIS, são criados detectores, com um tempo de vida e *status* associado, podem ser implementados preferencialmente como objetos ou estruturas (registros). Novamente, no contexto de detecção de intrusos, os antígenos correspondem a dados associados a conexões de redes. O algoritmo recebe um conjunto de arquivos de treinamento, com elementos de pacotes ou conexões de redes, cada entrada associada ou a um uso normal do computador ou a uma tentativa de ataque. São selecionados então um número pré-estabelecido de APCs e células T virgens, capazes de reconhecer entradas classificadas anteriormente como ataque, mas com baixa afinidade com entradas classificadas como normais. APCs representam no DT-AIS uma célula dos sistema imune inato, como uma célula dendrítica ou um macrófago, capaz de reconhecer padrões moleculares.

   É importante comentar aqui que apenas o treinamento inicial das APCs é necessário, uma vez que após isso o sistema pode ser usado como um

sistema de aprendizado dinâmico. Para melhorias de desempenho, entretanto, é interessante que a população inicial de linfócitos seja treinada por algum mecanismo de seleção negativa. Essa abordagem foi adotada durante a implementação deste trabalho, no qual as células T foram treinadas de forma semelhante às células B e células T auxiliares do IA-AIS.

2. O tecido é treinado (evolutivamente) para emissão de sinais co-estimulatórios, a partir da percepção de perigo. Esse treino é feito a partir de sinais existentes no sistema.

   Os dados de treinamento do DT-AIS são compostos de antígenos e sinais. Assim, para cada conexão de rede são associados os dados do pacote (antígenos) e dados do sistema (sinais). O treinamento dos sinais poderia ser efetuado de diversas maneiras, utilizando-se redes neurais, por exemplo. Neste trabalho, foi utilizado um treinamento de sinais a partir de uma avaliação estatística de cada sinal. Para isso, calculou-se as médias e desvios-padrão dos sinais, quando da sua ocorrência associada a patógenos ou ao comportamento normal. Para cada tipo de sinal, sua associação a perigo ou ausência de perigo, foi definida pela verificação das seguintes desigualdades:

$$av_d > (\gamma(av_n + std_n)) \tag{6.3}$$

$$av_n > (\gamma(av_d + std_d)) \tag{6.4}$$

   Onde av_d, std_d, av_n e std_n representam respectivamente a média e desvio-padrão do sinal associado a dados de perigo (av_d e std_d) e em momentos de comportamento normal (av_n, std_n). Nessas inequações, a constante γ é um parâmetro para permitir uma maior ou menor diferenciação dos sinais em seu uso na determinação do nível de perigo no ambiente.

   A definição dos detectores é dada da seguinte forma: i) caso a primeira desigualdade se verifique, então esse sinal pode ser utilizado para indicação de ambiente inflamatório; ii) caso a segunda desigualdade seja válida, então o sinal é tolerizante; iii) caso nenhuma das desigualdades seja válida, o sinal não é utilizado durante o processo de avaliação dos microorganismos.

3. Um novo micróbio M é apresentado ao sistema e é pré-processado por APCs, que coletam dados do ambiente e definem se apresentarão ou não sinais co-estimulatórios, a partir da ligação com o micróbio e dados ambientais.

   Após o treinamento, o algoritmo começa o processo de classificação, utilizando-se para isso de uma base de dados para avaliação. Essa base

também consiste em um conjunto de pacotes de redes diversos, e cada entrada é tratada pelo algoritmo como um "microorganismo", um conjunto de antígenos e sinais associados. Cada entrada da base de avaliação é então avaliada pelas APCs e, em caso de ligação, o mesmo é classificado como um PAMP, havendo a emissão de sinais co-estimulatórios para as células T. Além da avaliação antigênica, os sinais são também analisados: cada sinal associado ao antígeno é comparado de acordo com as desigualdades apresentadas nas inequações 6.3 e 6.4. De acordo com o resultado da comparação, há então uma emissão de sinais tolerizantes ou co-estimulatórios para as células T.

4. Para cada antígeno A_1 em M apresente A_1 para células T. Para cada célula T T_1 com reconhecimento antigênico, faça:

   (a) Se T_1 é tolerante, lança sinal tolerizante e incrementa tempo de vida.

   (b) Se T_1 é virgem e recebe mais sinais co-estimulatórios que regulatórios, torna-se efetora. Caso contrário, torna-se tolerante a esse antígeno.

   (c) Se T_1 é efetora e recebe mais sinais co-estimulatórios que regulatórios, aumenta reconhecimento antigênico, gera clones e incrementa tempo de vida.

   Calcula-se a afinidade do antígeno com cada célula T do sistema. Caso haja a ligação de alguma célula T, é verificado o contexto ativador, se tolerante ou co-estimulatório. De acordo com o tipo celular e os sinais do sistema, há um aumento ou decréscimo no reconhecimento de M como sendo ou não uma tentativa de intrusão. Nessa etapa há clonagem e hipermutação das células T efetoras que forem ativadas.

5. Determina-se a existência ou não de anomalia a partir do nível de reconhecimento antigênico do sistema.

   A definição final se o microorganismo apresentado é ou não uma tentativa de intrusão ou um pacote normal é dado pela somatória do reconhecimento antigênico de cada célula T que se ligar a esse microorganismo. Caso mais células efetoras sejam ativadas, então o microorganismo é tomado como um ataque, caso contrário é classificado como constituinte de uma conexão normal.

6. Diminui o tempo de vida de todas as células no sistema e gera novas células T virgens de forma aleatória, retornando ao passo 3 do algoritmo, até que não haja mais microorganismos a serem avaliados.

Semelhante ao IA-AIS, na etapa final, decrementa-se o TTL de todas as células do sistema e adiciona-se algumas novas células (APCs e células T virgens), como forma de manter a população com tamanho relativamente estável. Continua-se o processo de classificação, com o próximo microorganismo M, no passo 3. Caso, todos os microorganismos tenham sido apresentados, o algoritmo é encerrado.

É importante observar no DT-AIS que os detectores de sinais são treinadas apenas na etapa inicial, representando de certa forma a metáfora do desenvolvimento evolutivo das similares biológicas. Esse treinamento poderia ser feito de diversas formas, incluindo-se o uso do algoritmo de células dendríticas, apresentado na Figura 5.4 (página 127) ou variantes desse processo.

Uma forma simplificada para treinamento das APCs nesse algoritmo seria simplesmente selecionar um *pool* de células capazes de reconhecer sinais que caracterizem falhas ou anomalias no sistema, bem como possuam a habilidade de reconhecerem uma vasta quantidade de antígenos. Dessa maneira, essas células estariam mais próximas, enquanto metáforas, de suas similares biológicas na visão da *Danger Theory*. Uma metodologia parecida foi utilizada neste trabalho.

Existem claras semelhanças entre o IA-AIS e o DT-AIS, sendo que este último foi motivado pelos problemas comentados com o IA-AIS, sua complexidade e existência de diversos parâmetros. O DT-AIS foi proposto por este pesquisador justamente como uma forma intermediária e relativamente mais simples de se utilizar imunidade inata e adaptativa em SIAs. Dada a existência de menos elementos e parâmetros, ele é mais facilmente implementável, mas ainda assim consegue preservar elementos do IA-AIS significativos para este trabalho, entre eles o uso de uma visão mais ampla do sistema imune.

Uma visão sintética do algoritmo pode ser verificada na Figura 6.17, que mostra uma visão geral do mesmo, na Figura 6.18, que apresenta a etapa inicial de treinamento, e na Figura 6.19, que apresenta o processo de classificação de um dado microorganismo. De forma similar ao IA-AIS, quando da preparação de novas iterações, novas células são adicionadas ao sistema. Esse fato, somado ao processo de clonagem e hipermutação durante a classificação faz com que o DT-AIS também possua aprendizagem dinâmica, ou seja: o sistema vai “aprendendo”, à medida que classifica novos elementos.

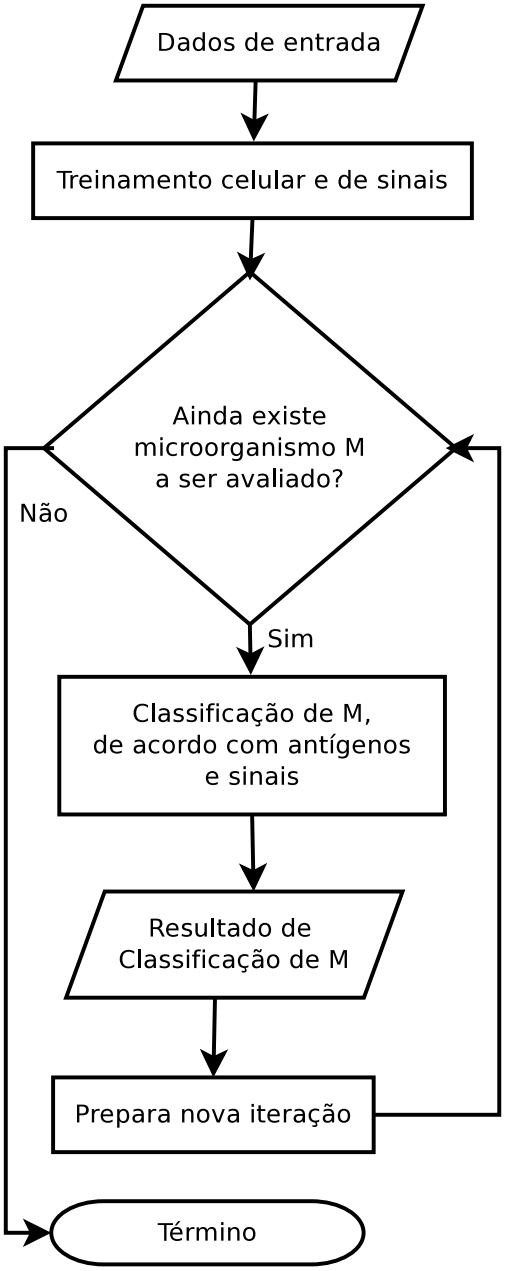


Figura 6.17: DT-AIS – Visão Geral

Pode ser percebido, a partir dos diagramas apresentados, Figura 6.17, Figura 6.18 e Figura 6.19, que as principais diferenças do DT-AIS com o IA-AIS são o menor uso de diferentes tipos celulares e o tratamento de sinais associados a antígenos. No DT-AIS, o processo de regulação da ativação celular é feito por uma análise do contexto, através da ativação de outras células e sinais associados ao antígeno, enquanto no IA-AIS essa tarefa é desempenhada por células T regulatórias.

Em vários outros aspectos, o DT-AIS possui semelhanças com o IA-AIS, como o uso de um tempo de vida associado a cada célula, por exemplo. As

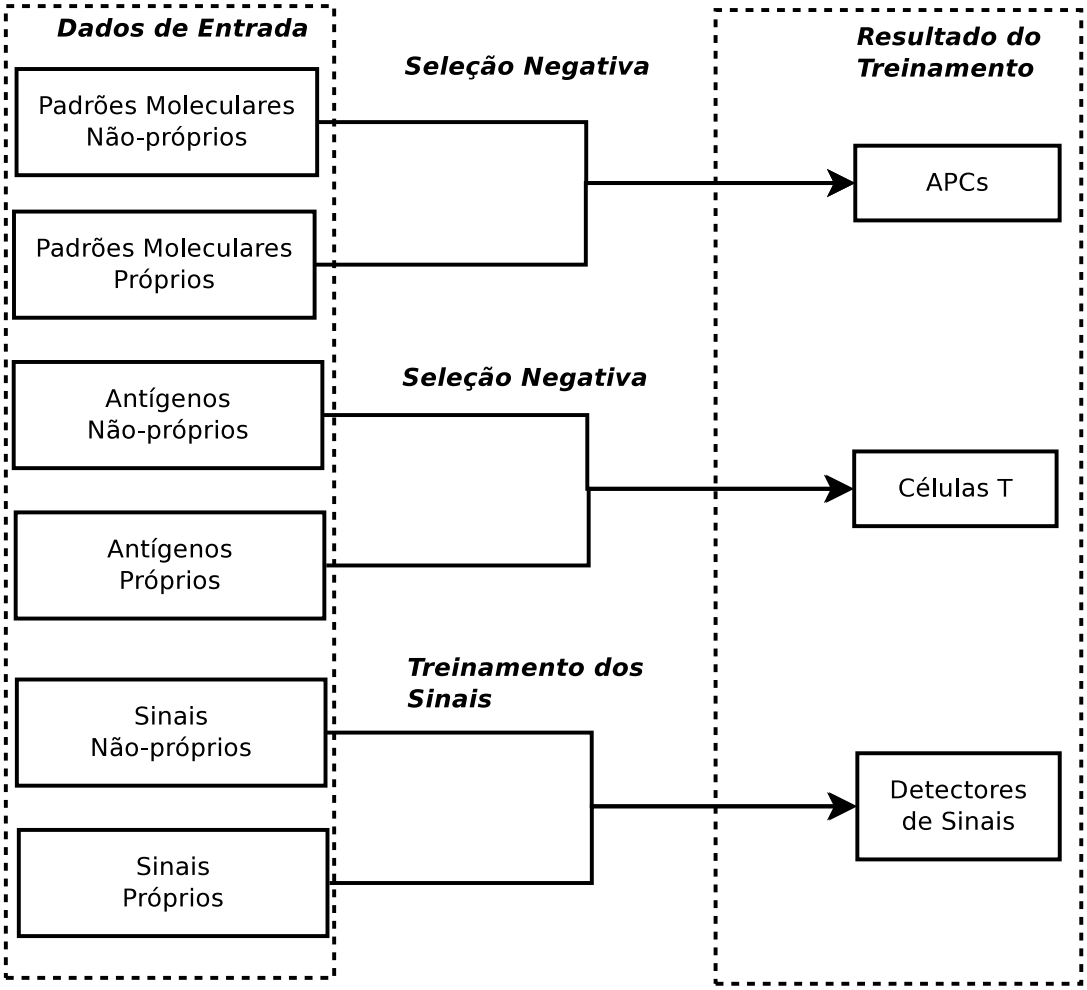


Figura 6.18: DT-AIS – Treinamento

semelhanças e diferenças ficam mais claras quando se analisa o pseudocódigo do DT-AIS, apresentado na Figura 6.20. Várias das funções e métodos utilizados nesse pseudocódigo já foram apresentados quando da descrição do IA-AIS, dessa forma nesta seção serão detalhados apenas os procedimentos utilizados exclusivamente pelo DT-AIS.

O algoritmo começa com a obtenção dos dados de treinamento e de teste, função GETTRAININGANDTESTDATA( ), e criação das células iniciais do sistema, funções GENERATEAPCS( ) e GENERATETCELLS( ). Essas funções, de forma similar às de criação no IA-AIS, são apenas chamadas a uma função mais geral, a GENERATEIMMUNECELLS( ), o que pode ser verificado na Figura 6.21. Destacam-se quatro parâmetros do DT-AIS utilizados nessa criação: tamanho da população inicial e tempo de vida inicial das APCs (`numAPCs` e `apcsTTL`) e células T (`numTCs` e `tcTTL`).

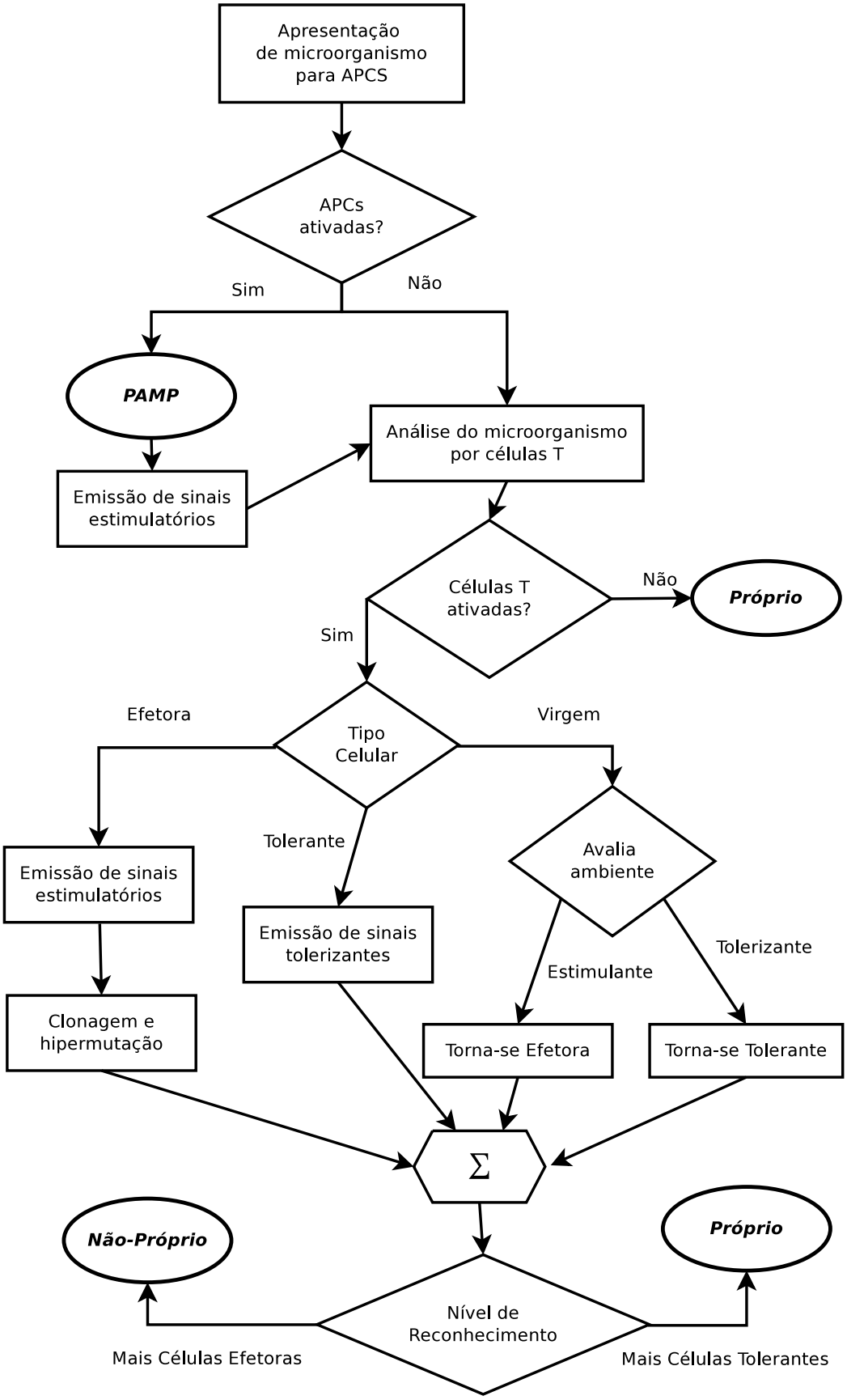


Figura 6.19: DT-AIS – Classificação

```
 1: getTrainingAndTestData(trainingFiles,testFiles)          ▷ [Preparation Phase]
 2:
 3: apcs ←generateAPCs(pampSeeds, selfSeeds)                 ▷ [Training Phase]
 4: tcells ← generateTCells(nonselfSeeds, selfSeeds)
 5: detectors ← trainSystemToSignals(pampSignals, selfSignals)
 6: microorganisms ← extractAntSigPatterns(testData)         ▷ [Classification Phase]
 7: for all mic in microorganisms do
 8:     context.enviroment ← 0
 9:     apcsBinding ← presentToCellPopulation(mic, apcs)
10:     if size(apcsBinding) > 0 then                        ▷ microorganism binded by APCs
11:         classifyMicroorganism(mic, PAMP)
12:         for all binding in apcsBinding do
13:             activateImmuneCell(binding.cell)
14:             context.enviroment ← context.enviroment + binding.rate
15:         end for
16:         nTCs ← generateImmuneCells(TCELL, estTCs, tcTTL, nsSeeds, sSeeds)
17:         append(nTCs, tcells)
18:     end if
19:     tcsBinding ← presentToCellPopulation(mic, tcells)
20:     if size(tcsBinding) > 0 then                         ▷ microorganism binded by T Cells
21:         context.enviroment = context.enviroment + evaluateSignals(mic,detectors)
22:         context.recType ← 0
23:         context.allClones ← [ ]
24:         for all binding in tcsBinding do                 ▷ verifies enviroment
25:             activateImmuneCell(binding.cell)
26:             context ← evalContext(binding.cell, context)
27:         end for
28:         if context.recType > 0 then                      ▷ Determinates anomaly by tcell detection
29:             classifyMicroorganism(mic, NONSELF)
30:         else
31:             classifyMicroorganism(mic, SELF)
32:         end if
33:         clonesBinding ← presentToCellPopulation(mic, context.allClones)
34:         clonesSelected ← selectBestClones(clonesBinding)
35:         addToCellPopulation(clonesSelected, tcells)
36:     else                                                 ▷ not binded by immune system: self
37:         classifyMicroorganism(mic, SELF)
38:     end if
39:     updateCellsTTL( )                                    ▷ [Preparing new iteration]
40:     addNewImmuneCellsDT(pampSeeds, nonselfSeeds selfSeeds)
41: end for
```

Figura 6.20: DT-AIS (Pseudocódigo)

Também de forma similar ao IA-AIS, ao término de cada iteração do algoritmo, linha 40 da Figura 6.20, novas células são adicionadas ao sistema, nesse caso usando a função addNewImmuneCellsDT( ), Figura 6.22. Essa função também faz chamada à generateImmuneCells( ). Dois novos parâ-

```
1: function GENERATEAPCS(pSeeds, sSeeds)
2:     return GENERATEIMMUNECELLS(APC, numAPCs, apcsTTL, pSeeds, sSeeds)
3: end function
4:
5: function GENERATETCELLS(nsSeeds, sSeeds)
6:     return GENERATEIMMUNECELLS(TCELL, numTCs, tcTTL, nsSeeds, sSeeds)
7: end function
```

Figura 6.21: Implementação de Funções de Geração de Células Imunes no DT-AIS

metros gerais do DT-AIS se destacam, `iterAPCs` e `iterTCs`, respectivamente o número de APCs e células T adicionadas em cada iteração.

```
1: function ADDNEWIMMUNECELLSDT(pSeeds, nsSeeds, sSeeds)
2:     nAPCs ← GENERATEIMMUNECELLS(APC, iterAPCs, apcsTTL, pSeeds, sSeeds)
3:     nTCs GENERATEIMMUNECELLS(TCELL, iterTCs, tcTTL, nsSeeds,sSeeds)
4:     APPEND(nAPCs, apcs)
5:     APPEND(nTCs, tcells)
6: end function
```

Figura 6.22: Função ADDNEWIMMUNECELLSDT( )

Dando sequência ao processo de treinamento do DT-AIS, linha 5 da Figura 6.20, há um treinamento dos detectores de sinais, através da função TRAINTISSUETOSIGNALS( ), apresentada na Figura 6.23. Como comentado anteriormente, neste trabalho, foi utilizado um treinamento de sinais a partir de uma avaliação estatística de cada sinal. Para isso, calculou-se as médias e desvios-padrão dos sinais, quando da sua ocorrência associada a patógenos ou ao comportamento normal, sendo avaliada as seguintes desigualdades:

$$av_d > (\gamma(av_n + std_n)) \tag{6.5}$$

$$av_n > (\gamma(av_d + std_d)) \tag{6.6}$$

Onde av_d, std_d, av_n e std_n representam respectivamente a média e desvio-padrão do sinal associado a dados de perigo (av_d e std_d) e em momentos de comportamento normal (av_n, std_n). Nessas inequações, a constante γ é um parâmetro geral do algoritmo para permitir uma maior ou menor diferenciação dos sinais em seu uso na determinação do nível de perigo no ambiente.

```
 1: function TRAINSYSTEMTOSIGNALS(pampSignals, selfSignals)
 2:     signalDetectors ← [ ]
 3:     for all type in signalTypes do
 4:         av_p ← AVERAGE(pampSignals[type])
 5:         std_p ← STANDARDDEVIATION(selfSignals[type])
 6:         av_s ← AVERAGE(pampSignals[type])
 7:         std_s ← STANDARDDEVIATION(selfSignals[type])
 8:         if av_p > (γ × (av_s + std_s)) then
 9:             signal.sigType ← 1                          ▷ Inflammatory signal
10:             signal.value ← (γ × (av_s + std_s))
11:             signal.average ← av_p
12:             signal.deviation ← std_p
13:         else if av_s > (γ × (av_p + std_p)) then
14:             signal.sigType ← -1                         ▷ Tolerant signal
15:             signal.value ← (γ × (av_p + std_p))
16:             signal.average ← av_s
17:             signal.deviation ← std_s
18:         else
19:             signal.sigType ← 0                          ▷ Unbiased signal
20:             signal.value ← 0
21:             signal.average ← 0
22:             signal.deviation ← 0
23:         end if
24:         signalDetectors[type] ← signal
25:     end for
26:     return signalDetectors
27: end function
```

Figura 6.23: Função TRAINTISSUETOSIGNALS( )

Pode ser percebido na implementação da função TRAINTISSUETOSIGNALS( ), Figura 6.23, que se a primeira desigualdade se verificar, então o sinal é inflamatório. Caso a segunda desigualdade seja válida, então o sinal é utilizado para indicação de ambiente tolerizante. Caso nenhuma das desigualdades seja válida, o sinal não é utilizado durante o processo de avaliação dos microorganismos, na fase de classificação.

Após o treinamento inicial do sistema, o DT-AIS inicia o processo de classificação dos dados de avaliação/teste, a partir da linha 6 do pseudocódigo apresentado na Figura 6.20. Depois do carregamentos dos dados a serem classificados, apresenta-se cada entrada de teste à população de APCs, através da função PRESENTTOCELLPOPULATION( ), já detalhada anteriormente. Caso alguma APC ligue-se ao microorganismo sendo avaliado, essa APC é ativada e há uma emissão de sinais estimulatórios ao ambiente, com o incremento na variável *context.enviroment*. Além disso, linha 16 da Figura 6.20, há a geração

de células T, cujos receptores são produzidos a partir de trechos do segmento associado ao microorganismo.

Após apresentação inicial do antígeno às APCs, há uma avaliação do mesmo pelas células T do sistema, linha 19 da Figura 6.20. Caso não ocorra nenhuma ligação, linha 37, então o antígeno é classificado como SELF, indicando um pacote componente de uma conexão normal. Entretanto, caso alguma célula T se ligue ao antígeno, é feita uma análise do contexto ativador, linha 21 da Figura 6.20. Inicialmente há uma avaliação dos sinais associados ao antígeno, utilizando os detectores treinados anteriormente. Essa avaliação é realizada através da função EVALMICSIGNALS( ), apresentada na Figura 6.24.

```
 1: function EVALUATESIGNALS(mic,signalDetectors)
 2:     micSignals ← EXTRACTSIGNAL(mic)
 3:     evalMicSignals ← 0
 4:     for all type in signalTypes do
 5:         detector ← signalDetectors[type]
 6:         if micSignals[type] ≥ detector.value then
 7:             if (detector.average + detector.deviation) ≠ 0 then
 8:                 eval ← micSignals[type]) / (detector.average + detector.deviation)
 9:                 eval ← detector.sigType × eval
10:                 evalMicSignals ← evalMicSignals + eval
11:             end if
12:         end if
13:     end for
14:     return evalMicSignals
15: end function
```

Figura 6.24: Função EVALMICSIGNALS( )

A função EVALMICSIGNALS( ) compara o valor de cada sinal com o do detector associado, calculando em seguida o quanto esse valor é maior que a soma do valor médior com o desvio-padrão do detector. Esse cálculo é utilizado para verificar o quanto o sinal desvia-se do ponto de ativação do detector. Em seguida, o resultado obtido é somado ou subtraído, se o sinal é estimulatório ou tolerizante respectivamente, de um valor utilizado para a análise de todos os sinais associados ao antígeno.

Após a análise dos sinais, para cada célula T que se ligar ao antígeno, é feita uma análise do contexto ativador, utilizando-se a função EVALCONTEXT( ), Figura 6.25. Indiferente ao contexto ativador, se estimulatório ou tolerizante, a célula é ativada. A função EVALCONTEXT( ) inicialmente verifica se a célula T já possui um tipo definido. Em caso afirmativo lançando sinal tolerante ou estimulatório, de acordo com o tipo celular, sendo que esse sinal é propor-

cional à taxa de ligação da célula ao antígeno. Caso a célula T seja virgem e receba mais sinais co-estimulatórios que regulatórios, torna-se efetora; em caso contrário, torna-se tolerante a esse antígeno. Caso a célula seja efetora, a mesma passa por processo de clonagem e hipermutação, de forma similar às células B no IA-AIS.

```
 1: function EVALCONTEXT(cell, context)
 2:     if cell.type = TOLERANT then
 3:         context.recType ← context.recType - 1
 4:         context.enviroment = context.enviroment - binding.rate
 5:     else if cell.type = EFFECTOR then
 6:         context.recType ← context.recType + 1
 7:         context.enviroment = context.enviroment + binding.rate
 8:         clones ← CLONECELL(binding.cell, binding.rate)
 9:         HYPERMUTATE(clones)
10:         APPEND(clones, context.allClones)
11:     else                                          ▷ Cell is Naïve
12:         if context.enviroment > 0 then            ▷ Stimulatory enviroment
13:             context.recType ← recType + 1
14:             binding.cell.type ← EFFECTOR
15:         else                                      ▷ Tolerant enviroment
16:             context.recType ← recType - 1
17:             binding.cell.type ← TOLERANT
18:         end if
19:     end if
20:     return context
21: end function
```

Figura 6.25: Função EVALCONTEXT( )

Uma vez analizado o contexto ativador, a classificação do antígeno é determinada pelo tipo de célula que se liga ao antígeno. Caso a ligação ocorra mais com células T efetoras que tolerantes, então o antígeno é classificado como uma tentativa de ataque, NONSELF. Em caso contrário, ou na ausência de ligação com células T, o antígeno é classificado como uma conexão normal da rede, SELF. Por fim, há uma atualização das células do sistema, através da eliminação das células suprimidas e decremento do TTL de cada célula restante. Além disso, como comentado anteriormente, novas células são adicionadas, por meio da função ADDNEWIMMUNECELLSDT( ).

De forma semelhante ao IA-AIS, o DT-AIS possui capacidade para incorporar, com relativa facilidade, outros elementos do sistema imune, o que será objeto de estudo de trabalhos futuros por esse pesquisador. A versão atual do DT-AIS é relativamente simples, comparado ao IA-AIS, exigindo uma definição

muito menor de parâmetros, o que pode ser verificado na Tabela 6.2, em são apresentados os principais parâmetros utilizados pelo algoritmo.

Tabela 6.2: Parâmetros do DT-AIS

Parâmetro	**Definição**
α	Coeficiente de mutação
μ	Número máximo de clones por célula
N_c	Número máximo de clones por iteração
γ	Coeficiente de diferenciação dos sinais
`startAPCs`	Número inicial de macrófagos
`startTC`	Número inicial de células T
`apcTTL`	TTL inicial de cada APC
`tcTTL`	TTL inicial de cada célula T
`ttlBonus[APC]`	TTL adicional para macrófagos ativados
`ttlBonus[TCELL]`	TTL adicional para células T ativadas
`estTC`	Número de células T adicionadas por estimulação de APCs
`iterAPCs`	Número de APCs adicionadas a cada iteração
`iterTCs`	Número de células T adicionadas a cada iteração
`activationThreshold[APC]`	Limiar de ativação para APCs
`activationThreshold[TCELL]`	Limiar de ativação para células T

6.4 Metodologia de Implementação

6.4.1 AISF - *Artificial Immune System Framework*

Para implementação dos algoritmos IA-AIS e DT-AIS, foi desenvolvido o AISF (*Artificial Immune System Framework*), uma biblioteca de classes e funções em Python, com suporte a um grande número de processos e elementos do sistema imune. Atualmente, essa biblioteca está sendo expandida para permitir uma representação mais adequada de citocinas e seu processamento por células do sistema imune. Com isso, pretende-se não apenas facilitar a implementação de algoritmos imunoinspirados, mas também criar um ferramental para estudos em modelagem do sistema imune. Uma visão geral do AISF, à época de finalização desta tese, pode ser obtida a partir da Figura 6.26, Figura 6.27 e Figura 6.28, que apresentam as principais classes implementadas.

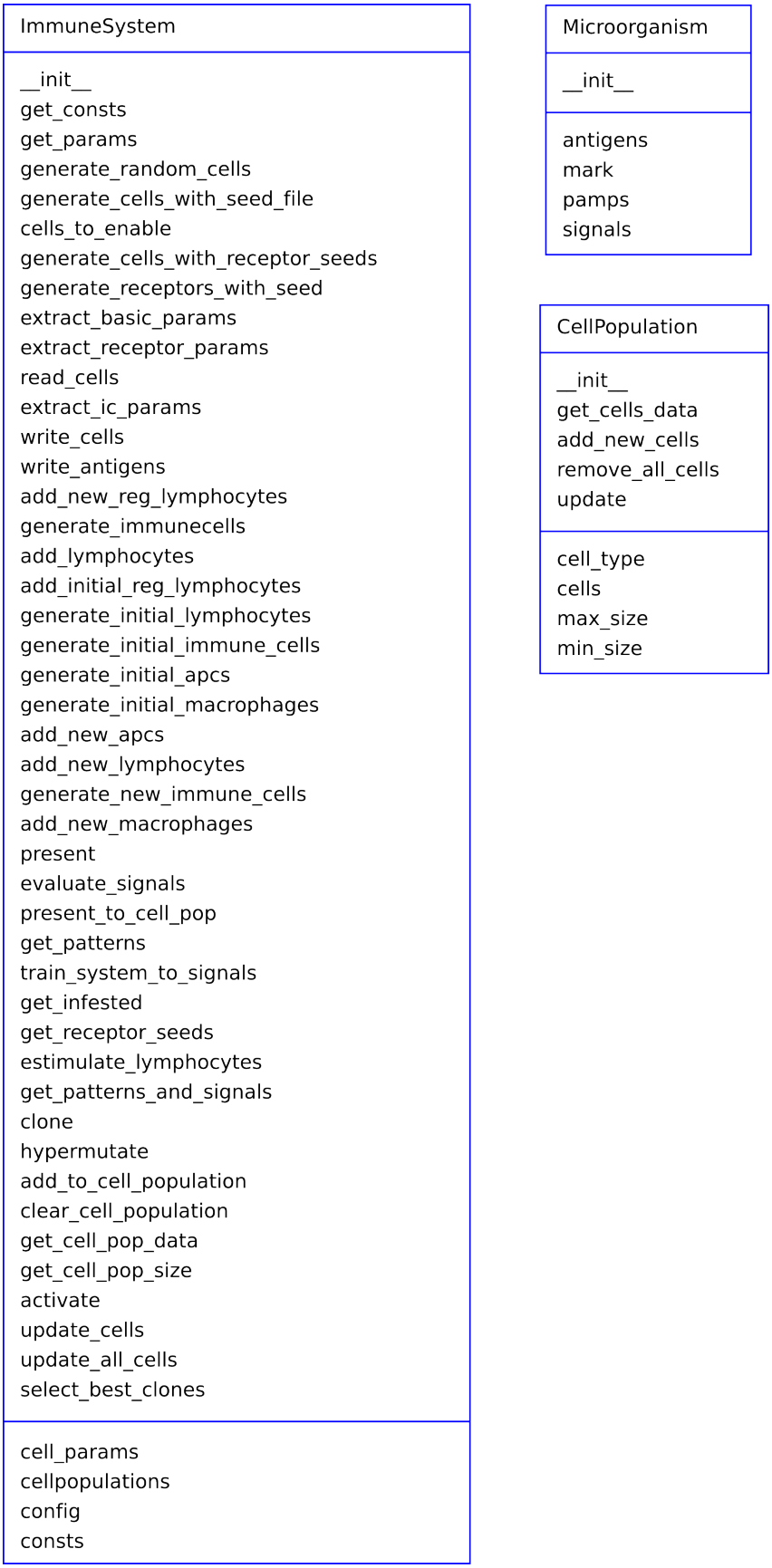


Figura 6.26: AISF - Classes ImmuneSystem, Microorganism e CellPopulation

Na Figura 6.26, tem-se em destaque a classe *ImmuneSystem*, que é a classe principal de todo o AISF. Os algoritmos utilizados neste trabalho, por exemplo, são implementados a partir de instâncias dessa classe. Destaca-se, entre os elementos dessa classe, o atributo *cellpopulations*, o conjunto de populações de células do sistema. Entre as funções implementadas por essa classe, encontram-se aquelas ligadas à geração das várias populações celulares, bem como as responsáveis pelo processo de apresentação de antígenos. Também foram implementados métodos para salvar e ler populações celulares inteiras a partir de arquivos.

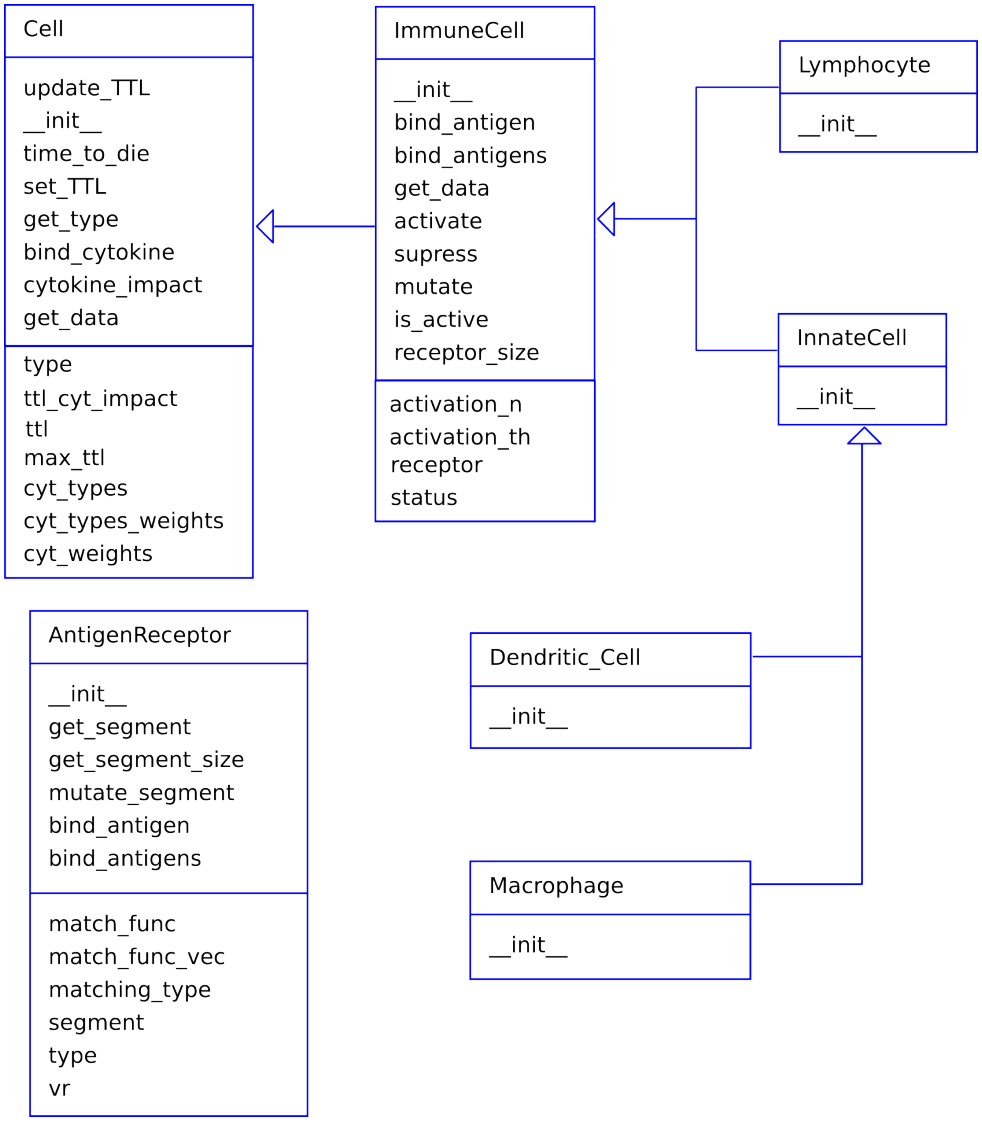


Figura 6.27: AISF - Classes Básicas de Células e Classes do Sistema Imune Inato

Os tipos celulares atualmente implementados podem ser verificados na Figura 6.27, que apresenta os tipos básicos e células do sistema imune inato, e na Figura 6.28, que apresenta os tipos de linfócitos disponíveis atualmente no sistema. A maior parte do processamento das células imunes é feito pela

classe abstrata *ImmuneCell*, que implementa os mecanismos de mutação, ativação e supressão das células do sistema imune. Entre seus elementos, merece destaque o parâmetro *receptor*, que é uma instância da classe *AntigenReceptor*, responsável pelo processo de ligação do antígeno à célula. Outro parâmetro importante, herdado da classe básica *Cell*, é o *ttl*, que indica o tempo de vida da célula em questão.

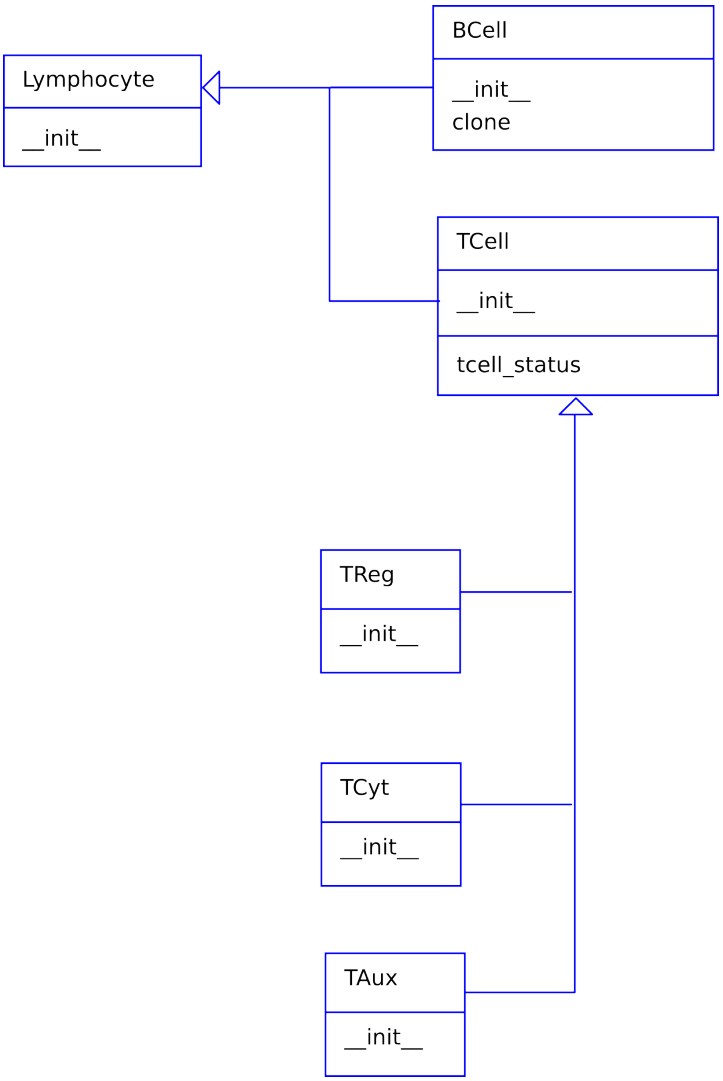


Figura 6.28: AISF - Classes de Linfócitos

A maior parte do processamento de diferentes tipos de células imunes, portanto, é bastante similar no AISF, diferenciando-se em alguns detalhes significativos. Entre esses detalhes, por exemplo, encontra-se o atributo *vr* da classe *AntigenReceptor*, que define se aquele receptor antigênico pode ou não sofrer mutação em células filhas. O único parâmetro das classes celulares imunes que não é totalmente compartilhado é *tcell_status*, utilizado apenas nas classes associadas aos linfócitos T e que indica se uma célula é tolerante, efetora ou simples (*naive*).

6.4.2 Representação dos Dados

O AISF permite a representação e uso de diversos tipos de informações e em diferentes formatos, facilitando o uso do *framework* em inúmeros tipos de aplicações. No caso típico de detecção de intrusos, optou-se pelo uso da representação em formato de seqüência de caracteres dos dados contidos ou associados aos pacotes de rede, evitando-se o uso de representações que não fossem significativas ao problema em si, em valores binárias por exemplo. A título de exemplo, na Figura 6.29 é apresentado um dos formatos que foram utilizados neste trabalho. No caso, `[data]` representa trechos, em hexadecimal, retirados do início e término da área de dados do pacote.

```
udp,5,76,40977,2,0,64,58399,38,56,[data]
udp,5,79,40978,2,0,64,52643,67,59,[data]
icmp,5,116,18295,0,0,64,3,3,0,0,[data]
```

Figura 6.29: Exemplos de representações dos dados

Para checar a possibilidade de ligação entre um receptor e um determinante antigênico ou padrão molecular, calcula-se a afinidade entre eles. Neste trabalho, adotou-se como função de afinidade a distância de Hamming, dada por:

$$afinidade = \frac{1}{b}(\sum_{i=1}^{b} eq(X_{i+offset}, Y_i)) \tag{6.7}$$

que calcula a similaridade entre duas cadeias X e Y, em que Y é mais curta ou de mesmo tamanho que X. Nessa fórmula ainda, b é o número de *bits* em Y e *offset* é a posição inicial de ligação em X, e o operador eq devolve 1 se dois caracteres forem iguais e 0 em caso contrário. Para o caso de duas cadeias idênticas, o somatório seria igual a B, resultando numa afinidade igual a 1. Cabe observar que essa medida difere da usada pelo sistema imune humano, em que o reconhecimento é feito por complementaridade, e não por similaridade.

Durante o desenvolvimento deste trabalho, outras funções de afinidade foram avaliadas, como o uso de *xor* ou distância entre caracteres. Essas funções, entretanto, não se mostraram adequadas ao problema em questão, com resultados bastante inferiores ao da solução adotada. Também foi avaliado o uso da distância baseada na abordagem Gestalt, utilizada para comparação de similaridade de sequências de caracteres (RATCLIFF; METZENER, 1988). Essa medida verifica o quão similares são duas sequências de caracteres a partir do maior "casamento" possível entre seus segmentos. Essa função mostrou-se

bastante adequada como medida, entretanto computacionalmente ineficiente, dado o grande número de comparações efetuadas. Entretanto, essa função pode-se mostrar adequada para uso em aplicações de modelagem computacional, que exijam mecanismos de ligação mais complexos.

6.4.3 Definição dos Parâmetros

É importante observar que os algoritmos aqui apresentados possuem diversos parâmetros, que podem, em muito, influenciar o desempenho dos mesmos. Entre esses parâmetros, encontram-se, por exemplo:

- tamanho inicial das populações utilizadas (APCs, linfócitos, anticorpos, etc.);
- limiar de ativação das APCs e linfócitos;
- tempo médio de vida das células e percentual de incremento, em caso de ativação;
- número de clones em caso de ativação.

Esses parâmetros são definidos em sua grande maioria através do arquivo de configuração que é utilizado para criar uma instância da classe *ImmuneSystem*. Na Figura 6.30, tem-se um extrato de um arquivo que foi utilizado durante este trabalho, na implementação do DT-AIS. Esse arquivo de configuração utiliza-se de uma formatação bastante conhecida, utilizando-se de seções, definidas entre colchetes, e parâmetros diversos com atribuições de valores pelo uso do caracter de igualdade ou dois pontos[4]. Na seção *general*, são definidos os elementos gerais do sistema, como o nível de *debug*, para impressão de mensagens nos diversos processos, os tipos celulares utilizados e aqueles parâmetros gerais que independem do tipo de célula em questão, como a taxa geral de clonagem ou mutação.

Para a definição dos tipos celulares, utilizou-se na configuração do recurso de herança, através do parâmetro *like*. Dessa maneira, um tipo celular pode "herdar" sua configuração básica de outro tipo, redefinindo apenas alguns dos parâmetros. No exemplo apresentado na Figura 6.30, apenas o tipo celular *basics* não herda suas configurações de outro tipo celular. Na realidade, como pode ser verificado na seção *general*, o tipo *basics* nem é implementado nessa

[4]O AISF aceita qualquer um desses dois símbolos na atribuição de valores.

```
[general]
debug= 1
dcells: yes
macrophages: no
bcells: no
tcells: no
tcyts: yes
max_clones = 2
sel_clones = 4
mutation_factor = 0.4
signal_factor = 1

# a basic cell, to be inherited by others
[basics]
creation_method: none
max_pop_size: 2000
min_pop_size: 0
ttl_start = 300
max_ttl = 10000
ttl_bonus = 200
cells_by_generation = 0
cells_by_turn = 0

[immunecells]
like: basics
segment_size = 6
segment_type = 0
activation_th: 0.7
status = 0
matching_type = 0
cells_by_generation = 800
cells_by_turn = 10
cells_by_stimulation = 0

[dcells]
like: immunecells
max_pop_size: 400
ttl_start = 5000
ttl_bonus = 2000
activation_th: 0.8
cells_by_generation = 200
cells_by_turn = 0.1

[tcyts]
like: immunecells
ttl_start = 400
ttl_bonus = 300
cells_by_stimulation = 1
```

Figura 6.30: Exemplo de Arquivo de Configuração de AISF

instanciação do AISF, tendo sido utilizada apenas para definição de parâmetros comuns aos vários tipos celulares.

Outros dois parâmetros importantes dizem respeito ao tipo de representação dos dados e a função de afinidade utilizada. Em *segment_type* é informado o formato adotado para representação dos dados, sendo que o tipo informado no arquivo de exemplo representa caracteres binários. O uso de caracteres normais, com letras, números e símbolos, seria representado pelo tipo 1. Atualmente, apenas esses dois tipos encontram-se representados no AISF, mas o sistema permite facilmente a definição de novos tipos. No parâmetro *matching_type* é informado qual é a função de afinidade a ser utilizada, sendo que 0 representa o uso da distância de Hamming, adotada neste trabalho.

6.5 Bases de Dados

6.5.1 KDD99

A primeira base de dados utilizada neste trabalho foi o KDD99, também conhecida como DARPA/KDD-99 ou KDD99 Dataset e disponível em (HETTICH; BAY, 1999), uma base de dados bastante utilizada para testes de algoritmos de aprendizado de máquina e ferramentas de detecção de intrusos. Ela inclui vários exemplos de ataques computacionais, encobertos em meio a tráfego normal de rede, incluindo uma grande variedade de intrusões simuladas em um ambiente de rede militar. Cada conexão é detalhada em termos de 41 características, incluindo-se aquelas relacionadas ao protocolo TCP, ao conteúdo do pacote e tráfego de rede. Essa base foi desenvolvida pelo Programa e Avaliação de Detecção de Intrusos da DARPA, em 1998, e foi preparada pelos Laboratórios Lincoln do MIT, possuindo mais de cinco milhões de registros, com diferentes distribuições dos tipos de ataques (YU; TSAI; WEIGERT, 2007). Em geral, a maior parte dos trabalhos adotando essa base de dados utiliza um subconjunto da mesma, com 311.029 registros anotados.

Bons resultados nessa base de dados são obtidos com taxa de detecção entre 98% a 99,5%, como pode ser visto em (KAYACIK; ZINCIR-HEYWOOD; HEYWOOD, 2007). Este trabalho utiliza *Self-Organizing Feature Maps* (SOMs), um tipo de rede neural artificial não supervisionada, para avaliar essa base de dados, comparando seus resultados com abordagens semelhantes também utilizando SOMs. Em (YU; TSAI; WEIGERT, 2007), também foi utilizada uma SOM, mas com otimização automática do sistema, utilizando o *feedback* dos

dados, quando falsas predições eram feitas, obtendo ganho significativo com esta metodologia.

Cabe ressaltar que os algoritmos apresentados neste capítulo permitem com relativa facilidade o uso de otimização automática do sistema, um mecanismo de aprendizado contínuo, possibilitando que correções sejam efetuadas em tempo de execução para corrigir eventuais desvios. Na maior parte dos casos, bastaria a remoção de células que levaram a um reconhecimento errado e geração de novas células com reconhecimento correto. Entretanto, é importante destacar que esse tipo de otimização deve ser utilizado com certa cautela, uma vez que ela depende do *feedback* do usuário ou do sistema para cada classificação efetuada pelo algoritmo em questão.

No problema de detecção de intrusos, em especial, é importante que o algoritmo tenha a possibilidade de uso desse tipo de estratégia, mas que não dependa disso para conseguir a grande maioria de seus resultados. Isso é justificado pelo fato que, na maior parte dos casos, o administrador de um sistema computacional só terá intenção de intervir em um sistema de detecção de intrusos quando falhas do mesmo estiverem atrapalhando o funcionamento normal da rede ou ele não estiver classificando corretamente um ataque para o qual foi configurado inicialmente para detectar, por exemplo. Por causa dessa situação, neste trabalho não foi utilizado qualquer tipo de correção do sistema durante a etapa de classificação, com a finalidade de evitar resultados que não poderiam ser repetidos com facilidade em um ambiente real.

Puderam ser verificados dois trabalhos bastante recentes envolvendo SIAs nessa base de dados, mas existem trabalhos anteriores avaliando, por exemplo, os problemas de escalabilidade dos algoritmos baseados em seleção negativa nessa base de dados, destacando-se (STIBOR; TIMMIS; ECKERT, 2005). Um dos trabalhos atuais utilizando essa base de dados é (GU; GREENSMITH; AICKELIN, 2008), baseado no algoritmo de células dendríticas. Os autores desse trabalho conseguiram cerca de 75% de acerto na detecção de anomalias e nenhum falso positivo com os dados normais. Melhores resultados, mais que 99% de correta classificação, são apresentados em (POWERS; HE, 2008), que utilizou uma abordagem híbrida baseada em SIA e SOM. Os resultados obtidos nesses dois trabalhos confirmam, de certa maneira, a adequação do uso de técnicas e algoritmos imunoinspirados no problema de detecção de intrusos.

6.5.2 *Packet Datasets* - Bases com Pacotes Capturados

A KDD99 é uma boa base de dados para comparação de algoritmos, mas, como comentado em (WU; YEN, 2009), ela já não reflete a situação da rede em dias atuais. Isto torna necessário a construção de novas bases de dados para verificar o desempenho de algoritmos nessa área. Na ausência de bases públicas recentes, durante a realização deste trabalho foram construídas duas bases de dados, capturando-se pacotes de rede em uma máquina alvo, utilizando-se tcpdump[5] e Scapy[6]. O tempo total de captura de cada base foi de aproximadamente duas horas, uma hora em cada situação de uso normal ou sob ataque. Em nossa opinião, esse tempo é suficiente para verificação do potencial de algoritmos e metodologias de detecção de intrusos, uma vez que durante esse tempo foi possível explorar vários usos de aplicativos de rede, durante o uso normal, bem como vários ataques tiveram que ser repetidos para obtenção do volume desejado de dados.

Para simulação do uso normal do computador, foram iniciadas diversas aplicações de rede usuais, como navegadores *web*, clientes de *e-mail* e mensagem instantânea, aplicações VoIP, entre outras. Durante a captura dos pacotes "anormais", foram iniciados vários ataques a partir de outra máquina, utilizando-se Metasploit[7] e Nmap[8], ferramentas bem conhecidas por especialistas de testes de intrusão. Com Metasploit, foram iniciados alguns ataques à máquina em questão, utilizando-se alguns dos *exploits* que poderiam ser usados contra ela. Com Nmap, foram explorados principalmente os ataques de varredura TCP, em especial os pacotes de varredura SYN. Na primeira base de dados foram coletados cerca de 100.000 pacotes ao todo, com 50.000 pacotes em cada classe (normal ou ataque).

Foram coletados não apenas pacotes TCP, mas também pacotes UDP e ICMP, assim as entradas coletadas possuem campos variáveis, dependendo do tipo de pacote capturado. Isso foi feito inclusive, para verificar a performance do algoritmo com dados variáveis, simulando mais completamente um ambiente real. Foram coletadas as informações básicas de cada pacote de rede, como seu tamanho, tamanho do cabeçalho, portas TCP e UDP utilizadas, *flags* existentes, e outras informações significativas. Caso o pacote tivesse área de dados, então eram coletados os primeiros e últimos cinco *bytes* de dados brutos, em formato hexadecimal. Assim, as entradas, com tamanho aproximado

[5]tcpdump: `http://www.tcpdump.org/`.
[6]Scapy: `http://www.secdev.org/projects/scapy/`
[7]Metasploit: `http://www.metasploit.com/`.
[8]Nmap: `http://nmap.org/`

de 50 caracteres, tinham uma parte inicial com informação do pacote e uma parte final com amostra dos dados, como pode ser verificado na Figura 6.31, onde `[data]` representa dado binário.

```
udp,5,28,15974,0,0,39,53104,51554,8,[data]
icmp,5,68,18593,0,0,64,3,3,0,0,[data]
tcp,5,44,64504,0,0,57,53104,3,2,2048,0,[data]
```

Figura 6.31: Amostra de Entradas Capturadas

Um exemplo de *script* em Python para captura desses dados pode ser verificado no Apêndice A.1, que acrescenta ao início da entrada o momento de captura. Esse tempo foi descartado quando da geração dessa primeira base de pacotes, uma vez que não é um elemento significativo para a detecção de intrusos. É importante observar que, diferente de (FANELLI, 2008), não foram utilizados os endereços de origem e destino de cada pacote, uma vez que esses valores podem ser facilmente modificados por intrusos em vários tipos de ataques. Mais ainda, máquinas com conexão discada ou ADSL costumam trocar seus endereços IPs em cada conexão com a Internet. Assim, esse tipo de informação precisa ser utilizado com extrema cautela quando usados em técnicas de detecção de intrusos.

Uma segunda base de dados foi coletada de forma relativamente similar, mas desta vez adicionando-se sinais do sistema, com a finalidade de uso em algoritmos baseados na Teoria do Perigo, no caso o DT-AIS. Para isso, foram associados a cada pacote de rede informações coletadas do sistema a partir de dados obtidos do arquivo `/proc/net/snmp`. Esse arquivo fornece, em ambientes UNIX, informações absolutas e gerais sobre o tráfego de rede em um dado computador, como por exemplo o número de pacotes TCP enviados e recebidos, Figura 6.32. Os dados obtidos foram normalizados, tendo sido calculada a taxa de variação a cada segundo, o que mostra o estado atual do sistema computacional com relação à rede, adequando-se de forma extremamente propícia à metáfora de sinais celulares. O *script* utilizado para captura dos sinais encontra-se listado no Apêndice A.2.

Vários dos sinais disponíveis em `/proc/net/snmp` não possuem potencial para uso em detecção de intrusos, sendo assim foram capturados apenas 23 sinais, associados principalmente ao tráfego ou à percepção de erros. Foram capturados, entre outros, o volume de pacotes recebidos por segundo em cada um dos protocolos básicos (IP, TCP, ICMP e UDP). Também foram capturados o número de pacotes de cada tipo que foram recebidos com erros e o número de segmentos retransmitidos. Por fim, também foi salvo o número de conexões

```
Ip: Forwarding DefaultTTL InReceives InHdrErrors InAddrErrors ForwDatagrams
InUnknownProtos InDiscards InDelivers OutRequests OutDiscards OutNoRoutes
ReasmTimeout ReasmReqds ReasmOKs ReasmFails FragOKs FragFails FragCreates
Ip: 2 64 169108 0 0 0 0 0 168662 164799 0 0 0 0 0 0 0 0 0
Icmp: InMsgs InErrors InDestUnreachs InTimeExcds InParmProbs InSrcQuenchs
InRedirects InEchos InEchoReps InTimestamps InTimestampReps InAddrMasks
InAddrMaskReps OutMsgs OutErrors OutDestUnreachs OutTimeExcds OutParmProbs
OutSrcQuenchs OutRedirects OutEchos OutEchoReps OutTimestamps
OutTimestampReps OutAddrMasks OutAddrMaskReps
Icmp: 213 31 213 0 0 0 0 0 0 0 0 0 0 3029 0 3029 0 0 0 0 0 0 0 0 0 0
IcmpMsg: InType3 OutType3
IcmpMsg: 213 3029
Tcp: RtoAlgorithm RtoMin RtoMax MaxConn ActiveOpens PassiveOpens
AttemptFails EstabResets CurrEstab InSegs OutSegs RetransSegs InErrs OutRsts
Tcp: 1 200 120000 -1 6220 0 44 138 3 145703 135501 2325 0 2301
Udp: InDatagrams NoPorts InErrors OutDatagrams RcvbufErrors SndbufErrors
Udp: 19326 3317 0 23944 0 0
UdpLite: InDatagrams NoPorts InErrors OutDatagrams RcvbufErrors SndbufErrors
UdpLite: 0 0 0 0 0 0
```

Figura 6.32: Arquivo `/proc/net/snmp`

TCP estabelecidas no computador no instante de captura de cada pacote. Na Figura 6.33, são apresentados exemplos de entradas dessa nova base de dados, onde `[packet]` representa as informações do pacote, como apresentadas na Figura 6.31. Nesse caso foi utilizado o *script* de captura de pacotes apresentado no Apêndice A.1, e o tempo capturado por esse *script* e o de captura de sinais apresentado no Apêndice A.2 foi utilizado para associar sinais a cada pacote capturado.

```
5926,0,0,0,0,0,0,0,0,0,6,0,0,0,5537,5537,0,0,5537,0,389,0,0:[packet]
10005,0,0,0,0,0,0,0,0,0,0,0,0,0,10005,10005,0,0,10005,0,0,0,0:[packet]
848,0,0,0,0,0,0,0,0,0,0,0,0,0,848,848,0,0,848,0,0,0,0:[packet]
```

Figura 6.33: Amostra de Sinais Capturados

É importante comentar que, quando da captura dessa segunda base de dados de pacotes, testes já haviam sido feitos com a primeira base de dados, utilizando-se o IA-AIS, com resultados bastante promissores (UCHÔA; MOTA-SANTOS; CAMINHAS, 2009). Dessa maneira, houve um estímulo para ampliar o escopo dos testes, envolvendo novos tipos de ataques e uso normal do sistema. Assim, nessa nova base de dados foram utilizados aplicativos adicionais de rede e uma quantidade maior de *sites* foi acessada, para captura dos pacotes normais de rede. Durante os ataques, além do Metasploit e do Nmap, também foram utilizados Nessus[9], OpenVAS[10] e SARA[11], aplicativos clássicos

[9]Nessus: `http://www.nessus.org/`.
[10]OpenVAS: `http://www.openvas.org/`.
[11]SARA: `http://www-arc.com/sara/`.

de verificação de vulnerabilidades. O objetivo foi o de verificar a possibilidade de detecção de uma amplitude maior de tipos de ataques, e a estabilidade dos algoritmos frente a dados mais variáveis. Foram capturados ao todo aproximadamente 90.000 pacotes de uso normal e 50.000 pacotes de ataque.

Como já comentado no Capítulo 3, várias ferramentas de segurança computacional, como as de captura de pacotes, são utilizadas em ataques computacionais. Essas ferramentas geralmente são utilizadas pelo invasor em uma etapa inicial do ataque, objetivando detectar falhas e vulnerabilidades no sistema alvo. Assim, é extremamente importante o desenvolvimento de ferramentas que detectem esse tipo de varredura, com a finalidade de evitar vários ataques ainda em sua fase embrionária.

6.6 Comentários Finais

Neste capítulo foram apresentados os algoritmos propostos neste trabalho, bem como o AISF, um *framework* em Python para a implementação de algoritmos imunoinspirados. Também foram apresentadas as bases de dados utilizadas durante os testes dos algoritmos. Tanto as bases de dados como a implementação deverão ser liberadas à comunidade acadêmica utilizando-se da licença CC-GPL[12]. No próximo capítulo, serão apresentados os principais testes efetuados, bem como os resultados obtidos com a implementação deste trabalho.

[12]Licença CC-GPL: `http://creativecommons.org/licenses/GPL/2.0/legalcode.pt`

CAPÍTULO
7

Resultados e Discussão

Frutos, ó frutos!
Eu vos aguardo e espero em mim, ó frutos!
Minha fome não se deterá a meio caminho;
Só satisfeita emudecerá;
Preceitos não poderão dar conta dela
E com privações nunca pude senão alimentar a alma.

André Gide, Os Frutos da Terra

Eia tu: desperta, e voa
por sobre o abismo e sobre os sonos
daqueles que um dia foram como ti:
e,
se disserem que não podes voar, não fales: simplesmente voa.

O Resto de Tudo, 1997

7.1 Comentários Iniciais

O trabalho aqui proposto pode ser caracterizado em quatro etapas de desenvolvimento:

Busca de Conhecimento: Esta etapa consistiu-se de uma extensa revisão bibliográfica em busca de inspiração para a proposta de novos modelos de aplicações de SIAs em Segurança Computacional. Aqui, pretendeu-se

não apenas vislumbrar metodologias diferenciadas de aplicação dos SIAs, mas também a proposta de novos algoritmos nessa área.

Proposição de Novas Metodologias: Utilizando-se a metáfora de um sistema imune computacional, um sistema que proteja elementos computacionais de ataques virtuais, pretendeu-se nesta etapa a proposição de modelos e algoritmos para uso de SIAs em Segurança Computacional, apresentados no Capítulo 6.

Implementação de Protótipos: Esta etapa foi caracterizada pelo desenvolvimento do AISF, permitindo a implementação dos algoritmos e modelos propostos na etapa anterior.

Testes dos protótipos: Esta etapa encontra-se apresentada neste capítulo e ainda deve ocupar parte dos trabalhos futuros, uma vez que os resultados obtidos abriram um grande leque de possibilidades e novas indagações.

O trabalho contou com o apoio, além do orientador e co-orientador do projeto, do aluno de iniciação científica Thiago Guzella, que focou seus esforços, sob nossa orientação, no uso do IA-AIS na detecção de SPAM (GUZELLA *et al.*, 2005; GUZELLA *et al.*, 2008). Os resultados apresentados neste texto dizem respeito apenas ao trabalho desenvolvido por este pesquisador, no processo de detecção de intrusos em redes de computadores.

7.2 Ambiente de Testes

Os testes foram realizados, utilizando-se as bases descritas na Seção 6.5 do capítulo anterior, em dois laboratórios computacionais. O primeiro laboratório era constituído de 10 estações com processador de 1800 MHz e 512 MB de RAM. O segundo laboratório era composto ao todo cinco computadores, dois desses *dual core*, e um *notebook*. Todos os computadores utilizavam o Gentoo Linux, com *kernel* Vanilla da série 2.6 configurado e compilado por este pesquisador[1].

Na Tabela 7.1 são apresentadas as configurações básicas dos computadores dos dois laboratórios, bem como o tempo de processamento, em segundos, de um teste efetuado com 55584 entradas utilizadas para treinamento e 6176

[1]O termo Vanilla refere-se ao kernel Linux oficial, sem qualquer alteração por desenvolvedores das diversas distribuições.

entradas para classificação/teste. É importante destacar que esse tempo variou de acordo com o tamanho do receptor e, principalmente, do tamanho da população de células em cada um dos algoritmos, sendo aqui apresentado apenas para dar ao leitor uma noção do desempenho da implementação.

Tabela 7.1: Descrição dos Computadores Utilizados

Máquina	**Descrição**	**Tempo de Execução**	
		IA-AIS	**DT-AIS**
Lab. 1 / Estações	Processador AMD Athlon 64 de 1800 Mhz, 512MB de Memória RAM DDR2, Disco SATA de 80GB, *chipset* nVidia		
Lab. 2 / Notebook	Processador AMD Athlon 64 de 2000 Mhz, 2GB de Memória RAM DRAM, Disco IDE de 100GB, *chipset* ATI	8323s	32094s
Lab. 2 / Maq. 1	Processador AMD Athlon 64 X2 Dual Core de 2600 Mhz, 3GB de Memória RAM DDR2, Disco SATA de 160GB, *chipset* nVidia	5921s	21342s
Lab. 2 / Maq. 2	Processador AMD Athlon 64 X2 Dual Core de 2200 Mhz, 4GB de Memória RAM DDR, Disco SATA de 160GB, *chipset* VIA	6638s	24043s
Lab. 2 / Maq. 3	Processador AMD Athlon 64 de 2000 Mhz, 2GB de Memória RAM SDRAM, Disco SATA de 160GB, *chipset* VIA	7390s	27979s
Lab. 2 / Maq. 4	Processador AMD Athlon de 1460 Mhz, 2GB de Memória RAM DIMM, Disco IDE de 80GB, *chipset* VIA	13747s	53208s
Lab. 2 / Maq. 3	Processador AMD Athlon de 1460 Mhz, 1GB de Memória RAM DIMM, Disco IDE de 40GB, *chipset* VIA	15125s	53752s

É importante ressaltar também que não houve preocupação, por parte deste pesquisador, em fazer uma implementação eficiente do AISF. A clareza da implementação foi tomada como prioridade, uma vez que há a pretensão de sua utilização para uso apenas em protótipos, não em sistemas comerciais. A prioridade da clareza também se justificou pela intensão de uso do AISF também em modelagem e no desenvolvimento e implementação de outros

algoritmos imunoinspirados. Dessa maneira, sempre que havia uma dúvida sobre como implementar uma determinada função ou procedimento, sempre se optou pela implementação mais clara, nunca a mais eficiente.

Duas metodologias de testes foram utilizadas para avaliação dos algoritmos, uma com seleção aleatória dos dados de treinamento e classificação/-teste e a outra baseada em particionamento dos dados coletados. Na primeira metodologia, cada teste em si consistiu de uma bateria de seis diferentes testes, realizados com diferentes conjuntos de treinamento e classificação, extraídos a partir do conjunto original de dados. Cada configuração de um dado algoritmo foi, portanto, executada seis vezes, com dados diferenciados. Nenhuma heurística foi utilizada para gerar as partições dos dados, portanto não havia qualquer garantia de um elemento da base de teste possuir um representante na base de treinamento. Isso foi feito, inclusive, para verificar o desempenho e estabilidade do algoritmo em situações similares às reais, quando ocorrem novas situações de ataque.

A segunda metodologia adotada, denominada na literatura de validação cruzada com k-pastas (*k-fold cross validation*, em inglês), consiste em particionar os dados em k partições (ou pastas). Durante cada teste, uma pasta é selecionada para teste/classificação e as demais pastas, ou parte delas, são utilizadas para treinamento do algoritmo. Com isso, tem-se que um dado utilizado para treinamento não é usado para teste/classificação, garantindo que os resultados do algoritmos não foram influenciados por uma classificação "viciada". Neste trabalho, a maioria dos testes utilizou-se de particionamento em dez pastas, tendo sido realizada uma bateria de dez testes, cada um utilizando a i-ésima pasta para teste e as demais para treinamento.

7.3 Testes Efetuados

7.3.1 IA-AIS no KDD99

Vários testes foram efetuados com o IA-AIS no KDD99, com o objetivo de verificar seu comportamento e desempenho, à medida que alguns de seus parâmetros eram modificados. Dois testes iniciais foram realizados, utilizando a seleção aleatória de dados de treinamento e classificação, sendo o primeiro com 5% do conjunto etiquetado do KDD99, dividido em três arquivos: dois (*self* e *nonself*) de arquivos de treinamento, com 2.500 entradas

cada, e um arquivo de teste, com 10.000 entradas. Como comentado anteriormente, essa partição foi repetida seis vezes, gerando três diferentes conjuntos de testes de treinamento e classificação. Assim, nesse primeiro teste, foram utilizados ao todo 90.000 entradas em cada avaliação de uma dada configuração.

Os principais parâmetros do IA-AIS utilizados nesse primeiro teste são listados na Tabela 7.2 e referem-se, em sua grande maioria àqueles apresentados durante a descrição do IA-AIS, Seção 6.3.1 e sumarizados na Tabela 6.1, na página 160. Os parâmetros auxiliares, listados na Tabela 7.2, são: i) o tamanho máximo da população de cada tipo celular, para evitar um crescimento populacional demasiado; ii) o limiar de ativação para a função de afinidade e iii) o tamanho do receptor das células do sistema.

Tabela 7.2: Configuração Inicial do IA-AIS

Parâmetro	**Célula B**	**Célula T Aux.**	**Célula T Reg**	**Macrófago**
TTL	400	300	300	5000
TTL bônus	300	200	200	2000
Limiar de Ativação	0.98	0.98	0.98	0.99
População Máxima	10000	10000	10000	2000
População Inicial	400	300	200	200
Células por iteração	12	10	5	1
Células por estimulação	2	2	–	–
Fator de Mutação	0.4	–	–	–
Máximo de clones por célula	4	–	–	–
Máximo de clones por etapa	10	–	–	–
Tamanho do receptor	30	30	30	20

O teste inicial foi repetido em um arquivo maior de teste, com 120.000 entradas em cada partição, para verificar o impacto da ação do tempo no sistema como um todo e sua adaptabilidade a novas entradas. Uma visão geral dos resultados desses testes iniciais é apresentada na Tabela 7.3, que mostra a média e desvio-padrão obtidos com a realização dos testes nas seis partições. Os resultados obtidos confirmaram uma expectativa inicial de baixo desvio nos resultados e ganho de desempenho com o uso progressivo do algoritmo.

Os dados apresentados na Tabela 7.3 referem-se aos elementos considerados por este pesquisador como mais importantes na análise do algoritmo. A detecção pelo sistema inato, por exemplo, caracteriza o quanto o algoritmo baseou sua avaliação utilizando apenas o reconhecimento por macrófagos, sem passar por uma segunda avaliação. Nesse caso, na tabela encontra-se listado o percentual de entradas que foram classificadas diretamente pelo sistema

Tabela 7.3: Testes com Configuração Inicial – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**10.000 entradas**	**120.000 entradas**
Detecção pelo Sistema Inato (%)	79.53 ± 0.77	79.74 ± 0.29
Estágios de Clonagem	0.50 ± 1.22	0.83 ± 2.04
Ocorrência de Supressão	0.67 ± 0.52	5.50 ± 3.78
Falsos Positivos (%)	0.05 ± 0.05	0.05 ± 0.05
Falsos Negativos (%)	1.23 ± 0.32	0.60 ± 0.19
Classificação Correta (%)	98.73 ± 0.27	99.33 ± 0.20

imune inato. Valores altos desse item são interessantes quando os dados possuem pouca variação entre si, com classes relativamente bem definidas. Nesse caso, o sistema inato possui menor custo computacional e o algoritmo será mais eficiente para o problema em questão.

Com dados "menos comportados", mais variáveis e com classes cujos limites não são claros, é interessante uma atuação maior do sistema imune inato, o que pode ser analisado olhando-se os estágios de clonagem e a ocorrência de supressão. Os estágios de clonagem referem-se ao número de iterações em que células B passaram por processo de clonagem e hipermutação, tendo sido ativadas, portanto. A ocorrência de supressões refere-se ao número de situações em que células B foram ativadas pelos dado de entrada, mas suprimidas por células T regulatórias. É importante destacar aqui que a soma desses dois itens (estágios de clonagem e ocorrência de supressão) informa o total de entradas que foram detectadas pelo sistema imune adaptativo.

As três últimas linhas da Tabela 7.3 referem-se ao grau de acerto do algoritmo, indicando o seu desempenho. Por falsos positivos, aponta-se o percentual de entradas que foram classificadas incorretamente como sendo uma tentativa de intrusão. Os falsos negativos, por sua vez, indicam o percentual de entradas associadas a tentativas de ataque e que foram classificadas incorretamente como conexões normais. Por fim, o índice de entradas classificadas corretamentes é apontado na última linha da tabela.

A expectativa inicial de um comportamento relativamente convergente em torno de uma região dos parâmetros devia-se a uma certa similaridade do algoritmo com alguns sistemas complexos, o que permitiu pensar em regiões de estabilidade e funções de potência do sistema como um todo. Indiferente a essa expectativa, os resultados apontaram como adequada a metodologia de avaliar o impacto de modificação dos parâmetros utilizando arquivos menores de teste. Essa foi a abordagem seguida, portanto, nos demais testes, para

Tabela 7.4: Novas Configurações do IA-AIS

Parâmetros Modificados	**Conf. 2**	**Conf. 3**	**Conf. 4**	**Conf. 5**	**Conf. 6**	**Conf. 7**
Pop. Inicial Macrófagos	2000	100	100	100	100	100
Pop. Inicial Células T Aux.	300	300	300	800	800	800
Pop. Inicial Células T Reg.	200	200	200	400	400	400
Pop. Máxima Macrófagos	2000	200	200	100	100	100
Pop. Máxima Linfócitos	5000	5000	5000	10000	10000	10000
Macrófagos por iteração	1	0.1	0.1	0.1	0.1	0.1
Células B por iteração	12	12	12	10	10	10
Tam. do receptor de linfócitos	30	30	20	20	20	20
Limiar Ativação Macrófagos	0.99	0.99	0.99	0.99	0.90	0.95
Limiar Ativação Linfócitos	0.98	0.98	0.98	0.98	0.80	0.90

propiciar a realização de uma grande quantidade de testes, explorando alguns dos limites do algoritmo.

Um resultado indesejável verificado nesses testes foi a baixa existência de estágios de clonagem, representando uma baixa ação do sistema imune adaptativo no processo de classificação. Alguns elementos da configuração inicial (Tabela 7.2) tem um grande impacto neste fato, principalmente o tamanho do receptor dos macrófagos e o tamanho inicial de sua população. Caso pretenda-se um sistema mais reativo com dados variáveis, é interessante fomentar uma melhor ação do sistema imune adaptativo. Como comentado anteriormente, em uma situação de dados mais "bem comportados", é interessante ter uma maior atuação do sistema imune inato, porque ele possui um custo computacional menor.

Com o objetivo de avaliar o impacto da configuração dos macrófagos no sistema como um todo, foram avaliadas outras configurações, modificando alguns dos parâmetros da configuração inicial, Tabela 7.4. Os resultados obtidos, mostrados na Tabela 7.5, permitiram estabelecer a importância de um balanceamento entre os tamanhos das populações de macrófagos e linfócitos: quanto maior a diferença entre esses parâmetros, maior o impacto da célula mais populosa durante a execução do algoritmo. Em um problema onde os dados sejam muito variáveis, é necessário uma maior atuação dos linfócitos, por exemplo.

Como pode ser verificado, comparando-se os resultados das configurações 5, 6 e 7, o limiar de ativação é um parâmetro extremamente importante. Entre outros ítens, ele depende da variabilidade dos dados e da expectativa de ligação a novos e diferentes dados. Em nossa opinião, KDD99 é um conjunto de dados relativamente "bem comportado", e os melhores resultados foram

Tabela 7.5: Testes com Novas Configurações – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 2**	**Conf. 3**	**Conf. 4**
Detecção pelo Sistema Inato (%)	79.23 ± 0.65	78.02 ± 0.79	77.73 ± 1.10
Estágios de Clonagem	0.00 ± 0.00	2.33 ± 3.50	16,17 ± 16,12
Ocorrência de Supressão	0.17 ± 0.41	6.83 ± 6.05	15.83 ± 14.55
Falsos Positivos (%)	0.03 ± 0.05	0.03 ± 0.05	0.03 ± 0.05
Falsos Negativos (%)	0.90 ± 0.29	2.37 ± 0.91	2.35 ± 0.89
Classificações Corretas (%)	99.05 ± 0.26	97.60 ± 0.91	97.62 ± 0.92

Ítem Observado	**Conf. 5**	**Conf. 6**	**Conf. 7**
Detecção pelo Sistema Inato (%)	77.85 ± 1.04	77.15 ± 2.02	76.46 ± 2.49
Estágios de Clonagem	21.83 ± 15.77	0.00 ± 0.00	0.17 ± 0.41
Ocorrência de Supressão	14.67 ± 14.19	21.50 ± 5.09	23.50 ± 14.87
Falsos Positivos (%)	0.00 ± 0.00	0.02 ± 0.04	0.03 ± 0.05
Falsos Negativos (%)	2.13 ± 0.45	3.27 ± 2.01	3.97 ± 2.29
Classificações Corretas (%)	97.87 ± 0.45	96.72 ± 2.00	96.00 ± 2.32

obtidos com altos valores de limiar de ativação. Essa afirmação pode ser confirmada pela alta taxa de detecção pelo sistema inato, cerca de 78% nas configurações testadas, estimulada pela existência de várias entradas repetidas no conjunto de dados.

A definição do tamanho do receptor é também um grande desafio. Valores altos induzem um decremento no processo de ligação, especialmente com altos valores de limiar de ativação. Um tamanho pequeno, por outro lado, tem o impacto de incrementar a probabilidade de ligação até um limiar indesejável, quando as células não possuem mais elementos suficientes para diferenciar entre o próprio e o não-próprio.

Quatro testes adicionais foram feitos, para verificar o impacto do tamanho do receptor, alterando-se apenas esse parâmetro, em todos os tipos celulares, na configuração 5, com resultados apresentados na Tabela 7.6. Essa configuração foi escolhida dado seus bom resultados na base de dados e boa atuação do sistema linfocitário. Os melhores resultados foram obtidos com receptores com tamanho cerca de 15% do tamanho de cada entrada de dados. Esse não é um resultado absoluto, pois ele depende do tamanho mínimo necessário à identificação do antígeno pela célula. Assim, em diferentes contextos outros valores precisam ser avaliados. Também foram avaliados os valores iniciais e de bônus do TTL, mas nenhum resultado significativo foi obtido. Isso permite, de certa maneira, afirmar que esses ítens não são tão sensíveis quanto tamanho do receptor e tamanho da população.

Tabela 7.6: Avaliando Tamanho do Receptor – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**10**	**15**	**25**	**30**
Det. Sistema Inato (%)	76.91 ± 1.03	77.67 ± 1.17	77.10 ± 1.23	76.30 ± 2.12
Estágios de Clonagem	18.50 ± 27.05	4.50 ± 7.45	21.50 ± 13.20	32.50 ± 56.67
Ocor. de Supressão	27.67 ± 18.08	19.00 ± 19.89	18.17 ± 15.87	32.50 ± 41.45
Falsos Positivos (%)	0.05 ± 0.04	0.03 ± 0.05	0.03 ± 0.05	0.02 ± 0.04
Falsos Negativos (%)	3.13 ± 1.01	2.65 ± 1.00	2.85 ± 1.12	3.48 ± 1.07
Class. Corretas (%)	96.82 ± 0.99	97.32 ± 0.98	97.12 ± 1.11	96.52 ± 1.07

Como forma de validar os melhores resultados, testes foram novamente executados nas configurações 1, 2 e 5 utilizando uma base de teste com 25.000 entradas em cada partição, com resultados apresentados na Tabela 7.7. Os valores obtidos confirmam nossa afirmação inicial sobre ganhos de performance com uso progressivo do algoritmo, mostrando, indiretamente, sua capacidade de convergência. Além disso, os resultados mostram uma grande estabilidade no algoritmo como um todo, com resultados bastante similares mesmo com alterações consideráveis em parâmetros significativos.

Tabela 7.7: Validando Resultados em uma Base de Teste Maior – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 5**
Detecção pelo Sistema Inato (%)	79.66 ± 0.23	79.54 ± 0.29	78.64 ± 0.26
Estágios de Clonagem	0.00 ± 0.00	0.00 ± 0.00	10.17 ± 8.70
Ocorrência de Supressão	1.67 ± 3.14	2.17 ± 2.56	18.83 ± 8.52
Falsos Positivos (%)	0.03 ± 0.05	0.03 ± 0.05	0.00 ± 0.00
Falsos Negativos (%)	0.70 ± 0.11	0.82 ± 0.10	1.65 ± 0.10
Classificação Correta (%)	99.27 ± 0.10	99.15 ± 0.05	98.35 ± 0.10

Como forma de validar e confirmar os resultados, os testes efetuados usando a metodologia de seleção aleatória de dos dados de treinamento e classificação foram repetidos utilizando-se uma variação do k-pastas. Os dados foram particionados em oitenta partições, tendo sido utilizado em cada teste nove partições para treinamento e uma partição para teste, como ilustrado na Tabela 7.8. Desse modo, cada execução do algoritmo utilizou 1/8 dos dados, sendo que isso foi adotado para permitir a execução de mais testes com diferentes configurações.

Os testes efetuados utilizando particionamento dos dados confirmaram os resultados, o que pode ser verificado na Tabela 7.9, relativa aos testes com as configurações de 1 a 7, e na Tabela 7.10, relativa aos testes variando o tamanho do receptor na configuração 5. Novamente, foi possível obter bons

Tabela 7.8: Pastas Usadas para Treinamento e Teste com o KDD99

Treinamento	Teste
9, 17, 25, 33, 41, 49, 57, 65, 73	1
10, 18, 26, 34, 42, 50, 58, 66, 74	2
11, 19, 27, 35, 43, 51, 60, 67, 75	3
12, 20, 28, 36, 44, 52, 61, 68, 76	4
14, 21, 29, 37, 45, 53, 62, 69, 77	5
15, 22, 30, 38, 46, 54, 63, 70, 78	6
16, 23, 31, 39, 47, 55, 64, 71, 79	7
17, 24, 32, 40, 48, 56, 65, 72, 80	8

resultados em várias configurações, conseguindo-se mais de 98% de acerto com a configuração 2. Ressalta aos olhos que, nas configurações com maior atuação do sistema imune inato, os resultados foram bastante parecidos nas duas abordagens metodológicas.

Tabela 7.9: Testes Adicionais – Metodologia de Particionamento dos Dados

Ítem Observado	Conf.1	Conf. 2	Conf. 3	Conf. 4
Det. Sist. Inato (%)	79.14 ± 0.28	79.18 ± 0.28	77.58 ± 1.28	77.99 ± 1.58
Est. de Clonagem	3.75 ± 4.27	3.38 ± 1.69	33.50 ± 77.14	14.50 ± 9.23
Ocor. de Supressão	16.88 ± 4.16	15.88 ± 4.58	15.38 ± 2.88	47.00 ± 36.90
F. Positivos (%)	0.03 ± 0.05	0.03 ± 0.05	0.03 ± 0.05	0.03 ± 0.05
F. Negativos (%)	1.11 ± 0.27	0.91 ± 0.42	2.19 ± 1.48	2.09 ± 1.47
Class. Corretas (%)	98.85 ± 0.25	98.88 ± 0.28	97.79 ± 1.46	97.86 ± 1.49

Ítem Observado	Conf. 5	Conf. 6	Conf. 7
Det. Sist. Inato (%)	72.11 ± 5.38	73.54 ± 3.67	73.59 ± 3.23
Est. de Clonagem	120.38 ± 165.80	0.50 ± 0.76	0.25 ± 0.46
Ocor. de Supressão	36.38 ± 4.96	207.38 ± 54.64	145.12 ± 16.16
F. Positivos (%)	0.01 ± 0.04	0.00 ± 0.00	0.00 ± 0.00
F. Negativos (%)	6.25 ± 4.27	6.72 ± 3.68	6.69 ± 3.25
Class. Corretas (%)	93.69 ± 4.23	93.28 ± 3.68	93.31 ± 3.25

Tabela 7.10: Avaliando Tamanho do Receptor – Metodologia de Particionamento dos Dados

Ítem Observado	10	15	25	30
Det. Sistema Inato (%)	75.24 ± 4.02	72.45 ± 5.66	73.98 ± 4.29	72.82 ± 4.14
Estágios de Clonagem	25.00 ± 13.59	49.38 ± 77.80	50.25 ± 10.11	65.38 ± 35.76
Ocor. de Supressão	148.00 ± 37.04	110.50 ± 53.37	22.38 ± 7.07	18.88 ± 4.19
Falsos Positivos (%)	0.00 ± 0.00	0.03 ± 0.05	0.03 ± 0.05	0.01 ± 0.04
Falsos Negativos (%)	4.64 ± 4.08	7.00 ± 4.67	5.49 ± 4.27	6.44 ± 4.22
Class. Corretas (%)	95.36 ± 4.08	92.99 ± 4.65	94.46 ± 4.24	93.53 ± 4.22

Nas configurações com maior atuação de linfócitos, entretanto, os resultados foram relativamente inferiores na metodologia de particionamento de

dados, principalmente no que diz respeito aos falsos negativos. Isso aponta para a importância de um ajuste fino nos parâmetros do algoritmo, de acordo com o problema em questão, especialmente no que diz respeito à atuação do sistema imune adaptativo. O número de células no sistema e principalmente a proporção entre a população do sistema inato e a população do sistema adaptativo possuem grande impacto no sistema como um todo, influenciando não apenas a eficiência, mas também o desempenho do algoritmo.

7.3.2 IA-AIS com Bases de Pacotes

INSPIRADO nos resultados e conclusões anteriores com o KDD99, foi verificado o desempenho do IA-AIS com a primeira base de pacotes. Novamente, seis testes foram feitos com cada configuração, com a metodologia de seleção aleatória dos dados de treinamento e classificação, mas desta vez usando 2000 entradas nos arquivos de treinamento (normal e anormal) e 10000 entradas em cada arquivo de teste, totalizando 84000 pacotes em cada bateria de teste. A partir da configuração 5 anterior, alterou-se o número máximo de macrófagos para 200, para obter uma melhor ação do sistema adaptativo. Na sequência, verificou-se cinco combinações de limiar de ativação e tamanho de receptor, apresentados na Tabela 7.11.

Tabela 7.11: Configurações nos Testes com Pacotes

Parâmetro	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**	**Conf. 5**
Tamanho do Receptor	10	8	6	5	6
Limiar de Ativação de Macrófagos	0.9	0.9	0.8	0.7	0.9
Limiar de Ativação de Linfócitos	0.8	0.8	0.7	0.6	0.8

Os resultados obtidos, Tabela 7.12, apontam o IA-AIS como uma ferramenta extremamente promissora para esse tipo de problema, com 96,5% de detecção correta em sua melhor configuração. Eles confirmam a importância em usar uma combinação dos sistemas inatos e adaptativos em problemas de classificação com grande variabilidade na composição dos dados. Além disso, os resultados corroboram a importância de uma boa sintonia dos parâmetros utilizados, especialmente tamanho das populações e dos receptores.

Os testes foram repetidos utilizando-se k-pastas: os dados foram divididos em dez pastas, nove pastas usadas para treinamento e uma pasta usada para teste/classificação. Os resultados em geral, apresentados na Tabela 7.13, mantiveram-se bastante próximos dos anteriores, Tabela 7.12, com exceção da configuração 4, a com menor limiar de ativação para as células do sistema.

Tabela 7.12: Avaliando IA-AIS com Primeira Base de Pacotes – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Detecção pelo Sistema Inato (%)	45.47 ± 2.33	46.08 ± 1.35	49.68 ± 0.61
Estágios de Clonagem	43.67 ± 16.24	35.50 ± 18.13	6.33 ± 6.50
Ocorrência de Supressão	228.17 ± 52.53	199.33 ± 36.20	126.33 ± 14.71
Falsos Positivos (%)	0.25± 0.05	0.10 ± 0.00	0.53 ± 0.08
Falsos Negativos (%)	6.55 ±2.08	5.85 ± 1.04	2.98 ± 0.67
Classificação Correta (%)	93.22 ± 2.06	94.05 ± 1.04	96.50 ± 0.68

Ítem Observado	**Conf. 4**	**Conf. 5**
Detecção pelo Sistema Inato (%)	45.33 ± 1.36	46.21 ± 0.99
Estágios de Clonagem	0.00 ± 0.00	15.00 ± 27.03
Ocorrência de Supressão	60.67 ± 10.07	210.00 ± 18.15
Falsos Positivos (%)	1.30 ± 0.32	0.25 ± 0.05
Falsos Negativos (%)	8.15 ± 1.11	6.17 ± 0.89
Classificação Correta (%)	90.55 ± 1.00	93.58 ± 0.89

Isso com certeza tornou o algoritmo mais sensível aos dados de treinamento e classificação, uma vez que os receptores tem maior facilidade de ligação com qualquer tipo antigênico nessa configuração em especial.

Como esperado, houve bem menos atuação da parte inata do algoritmo nesse novo conjunto de dados. Apesar disso, os resultados são ainda muito bons, mostrando a importância em utilizar mais que um mecanismo de detecção nesse tipo de problema, com grande variabilidade dos dados. Isso é corroborado justamente pelos resultados obtidos com a configuração 4, o pior resultado nesse conjunto de dados e com nenhum reconhecimento ativo por parte do sistema adaptativo nas duas metodologias usadas.

Outro fato importante a ser observado nesses resultados é a estabilidade geral do algoritmo. Houve grande variabilidade dos processos de clonagem e supressão, mostrando a adaptabilidade do mesmo aos dados, entretanto, os resultados finais foram bastante estáveis, com baixo desvio-padrão nas taxas de detecção. Isto mostra o alto potencial do algoritmo em detecção de anomalias. Testes adicionais foram efetuados, alterando a população inicial e tamanho máximo da população para valores maiores, obtendo resultados bastante similares, mostrando que a estabilidade é resultado da interação do sistema, não do número de detectores, evitando assim problemas de escalabilidade.

Na seqüência, testes foram efetuados com a segunda base de pacotes, a que continha sinais do ambiente de rede, além das informações dos pacotes em si. Como o IA-AIS não possui mecanismos para tratamento dos sinais,

Tabela 7.13: Avaliando IA-AIS com Primeira Base de Pacotes – Metodologia de Particionamento dos Dados

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Detecção pelo Sistema Inato (%)	33.72 ± 1.17	37.97 ± 1.52	43.06 ± 2.50
Estágios de Clonagem	805.50 ± 96.21	644.70 ± 106.13	541.90 ± 267.30
Ocorrência de Supressão	1087.50 ± 83.24	838.20 ± 112.37	729.20 ± 76.48
Falsos Positivos (%)	0.06± 0.05	0.10 ± 0.05	0.16 ± 0.12
Falsos Negativos (%)	11.06 ± 1.13	8.38 ± 1.40	4.78 ± 0.64
Classificação Correta (%)	93.22 ± 2.06	94.05 ± 1.04	95.06 ± 0.70

Ítem Observado	**Conf. 4**	**Conf. 5**
Detecção pelo Sistema Inato (%)	22.78 ± 0.85	43.75 ± 1.42
Estágios de Clonagem	0.00 ± 0.00	450.20 ± 91.46
Ocorrência de Supressão	1151.60 ± 57.15	1143.10 ± 122.80
Falsos Positivos (%)	0.00 ± 0.00	0.14 ± 0.08
Falsos Negativos (%)	29.60 ± 0.86	4.48 ± 0.67
Classificação Correta (%)	70.40 ± 0.86	95.39 ± 0.68

surgiu a indagação se melhores resultados seriam obtidos utilizando-se os sinais como parte do antígeno, ou simplesmente descartando-os. Novamente foram utilizadas as configurações apresentadas na Tabela 7.11 e as duas metodologias de avaliação do algoritmo. Utilizando a metodologia de de seleção aleatória dos dados de treinamento e classificação, os resultados obtidos com a utilização e descarte dos sinais como parte do antígeno são apresentados na Tabela 7.14 e na Tabela 7.15, respectivamente. Os resultados adotando a metodologia de k-pastas, por suas vez, são apresentados na Tabela 7.16, com utilização dos sinais, e na Tabela 7.17, com descarte dos sinais.

Os resultados obtidos são extremamente interessantes e bastante iluminativos sobre o desempenho e funcionamento do IA-AIS. Uma menor eficiência do sistema já era esperada, obviamente, uma vez que é extremamente difícil a detecção de varreduras de vulnerabilidades. Ainda assim, foi possível obter resultados superiores a 80% nesse processo, o que é extremamente satisfatório, do ponto de vista da Segurança Computacional, dado o baixo desempenho das ferramentas atualmente existentes nesse tipo de tarefa. O fato mais importante desses resultados, entretanto, são os indícios de pontos onde o sistema pode ser melhorado, com vistas a desempenhos superiores.

Um elemento que chama fortemente a atenção nos testes efetuados considerando o sinal como parte do antígeno é a inexistência de uma resposta ativa do sistema linfocitário do algoritmo. Essa atuação torna-se presente e significativo na maior parte das configurações quando esses sinais são descartados,

Tabela 7.14: Avaliando IA-AIS com Segunda Base de Pacotes – Sinais como Parte do Antígeno – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Detecção pelo Sistema Inato (%)	19.74 ± 1.81	24.95 ± 1.39	19.44 ± 2.02
Estágios de Clonagem	647.67 ± 190.12	209.67 ± 57.62	117.33 ± 32.09
Ocorrência de Supressão	2386.83 ± 178.60	1402.83 ± 265.57	2004.17 ± 80.45
Falsos Positivos (%)	1.15 ± 0.24	1.12 ± 0.23	1.15 ± 0.24
Falsos Negativos (%)	16.95 ± 2.29	15.10 ± 1.64	21.35 ± 1.68
Classificação Correta (%)	79.15 ± 0.98	83.80 ± 1.50	77.53 ± 1.57

Ítem Observado	**Conf. 4**	**Conf. 5**
Detecção pelo Sistema Inato (%)	7.24 ± 2.96	19.86 ± 0.63
Estágios de Clonagem	0.00 ± 0.00	131.33 ± 35.66
Ocorrência de Supressão	1220.33 ± 444.04	1976.50 ± 36.59
Falsos Positivos (%)	0.63 ± 0.16	0.60 ± 0.13
Falsos Negativos (%)	33.93 ± 2.66	20.23 ± 1.05
Classificação Correta (%)	65.43 ± 2.68	79.15 ± 0.98

Tabela 7.15: Avaliando IA-AIS com Segunda Base de Pacotes – Sinais Descartados – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Detecção pelo Sistema Inato (%)	26.59 ± 1.83	27.72 ± 0.88	20.90 ± 2.51
Estágios de Clonagem	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Ocorrência de Supressão	66.67 ± 9.73	47.33 ± 11.76	32.17 ± 2.71
Falsos Positivos (%)	0.13± 0.05	0.10 ± 0.00	0.37 ± 0.08
Falsos Negativos (%)	14.08 ±2.07	12.88 ± 1.19	20.03 ± 2.57
Classificação Correta (%)	85.77 ± 2.05	87.03 ± 1.18	79.50 ± 2.53

Ítem Observado	**Conf. 4**	**Conf. 5**
Detecção pelo Sistema Inato (%)	17.15 ± 2.44	26.53 ± 1.53
Estágios de Clonagem	0.00 ± 0.00	15.00 ± 27.03
Ocorrência de Supressão	33.33 ± 7.53	47.00 ± 17.15
Falsos Positivos (%)	0.43 ± 0.14	0.13 ± 0.05
Falsos Negativos (%)	23.85 ± 2.66	14.13 ± 1.65
Classificação Correta (%)	75.73 ± 2.75	85.73 ± 1.64

avaliando-se apenas os dados contidos nos pacotes. Isso aponta, obviamente para um dos seguintes caminhos: i) melhoria da função de afinidade, tratando sinais e pacotes de formas diferenciadas; ii) incorporação de sinais pelo IA-AIS, como um elemento inflamatório do ambiente. Obviamente, o segundo caminho parece muito mais promissor e mesmo adequado, do ponto de vista de metáfora biológica. Entretanto, isso precisa ser analisado com bastante

Tabela 7.16: Avaliando IA-AIS com Segunda Base de Pacotes – Sinais como Parte do Antígeno – Metodologia de Particionamento dos Dados

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Detecção pelo Sistema Inato (%)	14.22 ± 4.36	23.59 ± 4.10	9.77 ± 4.67
Estágios de Clonagem	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Ocorrência de Supressão	233.30 ± 64.50	228.30 ± 29.83	183.60 ± 37.63
Falsos Positivos (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Negativos (%)	28.55 ± 3.63	20.79 ± 3.40	32.26 ± 3.88
Classificação Correta (%)	71.45 ± 3.63	79.21 ± 3.40	67.72 ± 3.86

Ítem Observado	**Conf. 4**	**Conf. 5**
Detecção pelo Sistema Inato (%)	13.52 ± 0.97	21.70 ± 5.98
Estágios de Clonagem	0.00 ± 0.00	0.00 ± 0.00
Ocorrência de Supressão	277.30 ± 17.38	241.00 ± 41.59
Falsos Positivos (%)	0.00 ± 0.00	0.00 ± 0.00
Falsos Negativos (%)	29.13 ± 0.81	22.37 ± 4.95
Classificação Correta (%)	70.87 ± 0.81	77.62 ± 4.95

Tabela 7.17: Avaliando IA-AIS com Segunda Base de Pacotes – Sinais Descartados – Metodologia de Particionamento dos Dados

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**
Det. pelo Sistema Inato (%)	13.95 ± 2.29	24.28 ± 3.57	6.71 ± 3.63
Estágios de Clonagem	1437.90 ± 1224.60	912.20 ± 418.48	279.20 ± 133.11
Ocorrência de Supressão	1192.80 ± 502.32	882.20 ± 139.15	421.70 ± 245.91
Falsos Positivos (%)	0.01± 0.03	0.00 ± 0.00	0.00 ± 0.00
Falsos Negativos (%)	20.80 ± 4.36	13.02 ± 3.49	32.61 ± 3.24
Classificação Correta (%)	79.17 ± 4.36	86.96 ± 3.49	67.38 ± 3.25

Ítem Observado	**Conf. 4**	**Conf. 5**
Det. pelo Sistema Inato (%)	2.58 ± 0.39	17.84 ± 8.08
Estágios de Clonagem	0.00 ± 0.00	385.60 ± 88.76
Ocorrência de Supressão	196.40 ± 25.86	729.00 ± 341.01
Falsos Positivos (%)	0.00 ± 0.00	0.00 ± 0.00
Falsos Negativos (%)	38.21 ± 0.31	22.53 ± 6.90
Classificação Correta (%)	61.79 ± 0.31	77.44 ± 6.88

cuidado, buscando avaliar como o processamento de sinais poderia ser incorporado pelo sistema.

Outro resultado importante obtido no conjunto total de testes foi a verificação da segunda configuração como a mais estável e promissora entre todas. Apesar de não necessariamente obter os melhores resultados sempre, ela foi a que obteve os resultados mais constantes, com baixo desvio-padrão nos resultados. Testes e análises adicionais precisam ser efetuados para verificar o

motivo desse resultado, bem como se isso se manterá com outros conjuntos de dados ou problemas similares. De qualquer forma, essa configuração deverá ser utilizada como base em futuros testes do IA-AIS.

7.3.3 DT-AIS com Bases de Pacotes

Os testes efetuados com o DT-AIS obviamente foram influenciados pelos resultados com o IA-AIS na definição de seus parâmetros iniciais. Naturalmente, existem bem menos elementos a serem configurados no DT-AIS, simplificando em muito as possibilidades e testes, uma vez que esse utiliza-se apenas de células T e APCs. Para implementação no AISF, utilizou-se células dendríticas, como APCs, e células T citotóxicas, adotando-se uma configuração bastante similar à configuração inicial do IA-AIS, Tabela 7.18.

Tabela 7.18: Configuração Inicial do DT-AIS

Parâmetro	**Célula T**	**APC**
TTL	400	5000
TTL bônus	300	2000
Limiar de Ativação	0.7	0.8
População Máxima	400	2000
População Inicial	200	800
Células por iteração	1	0.1
Células por estimulação	1	–
Fator de Mutação	0.4	–
Máximo de clones por célula	2	–
Máximo de clones por etapa	4	–
Tamanho do receptor	6	6

Houve uma avaliação inicial do parâmetro de diferenciação dos sinais, mas os testes apontaram pouco impacto desse parâmetro, a menos que esse parâmetro sofresse variações relativamente grandes, multiplicados ou divididos por múltiplos de 10. Mesmo nesse caso, os resultados obtidos nesses testes preliminares mostram que esse parâmetro é muito pouco sensível, aparentemente podendo ser descartado do algoritmo. Decidiu-se por sua manutenção apenas para que essa situação pudesse ser verificada e possivelmente confirmada em outros tipos de problemas, antes de um descarte definitivo. Neste trabalho, optou-se por não utilizar esse parâmetro nos testes apresentados. Impulsionado pelos resultados com o IA-AIS, variou-se apenas os tamanhos dos receptores e o limiar de ativação, como apresentado na Tabela 7.19.

Tabela 7.19: Configurações do DT-AIS nos Testes com Pacotes

Parâmetro	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**
Tamanho do Receptor (APCs e Linfócitos)	6	8	6	8
Limiar de Ativação de APCs	0.8	0.8	0.9	0.9
Limiar de Ativação de Linfócitos	0.7	0.7	0.8	0.8

Parâmetro	**Conf. 5**	**Conf. 6**	**Conf. 7**
Tamanho do Receptor (APCs e Linfócitos)	10	8	10
Limiar de Ativação de APCs	0.9	0.95	0.95
Limiar de Ativação de Linfócitos	0.8	0.9	0.9

As configurações não foram testadas na primeira base de pacotes, uma vez que este algoritmo precisa de sinais para sua execução. Os resultados obtidos com a segunda base de pacotes usando a metodologia de seleção aleatória dos dados de treinamento e classificação são apresentados na Tabela 7.20, na qual são destacados, entre outros elementos, o número de ocorrência de processos de estimulação de linfócitos por APCs, o que aponta para o reconhecimento inato do antígeno. Além disso, são informados também os números de clones gerado em todo o processo e a proporção entre células tolerantes e efetoras ao término do processo. Os resultados obtidos usando a metodologia de particionamento dos dados (k-pastas), por sua vez, são apresentados na Tabela 7.21.

Tabela 7.20: Avaliando DT-AIS com Segunda Base de Pacotes – Metodologia de Seleção Aleatória dos Dados de Treinamento e Classificação

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**
Estimulações	2653 ± 223	3011 ± 144	2695 ± 161	3384 ± 126
Clones Gerados	21155 ± 484	21312 ± 401	20864 ± 363	20023 ± 357
Tol./Efetor (%)	0.93 ± 0.04	0.87 ± 0.04	0.95 ± 0.03	0.57 ± 0.05
Falsos Positivos (%)	4.83 ± 0.22	4.45 ± 0.16	4.37 ± 0.15	2.57 ± 0.45
Falsos Negativos (%)	4.57 ± 0.31	3.90 ± 0.26	4.63 ± 0.26	4.08 ± 0.38
Class. Correta (%)	90.63 ± 0.31	91.67 ± 0.36	90.90 ± 0.27	93.35 ± 0.73

Ítem Observado	**Conf. 5**	**Conf. 6**	**Conf. 7**
Estimulações	2537 ± 109	3398 ± 149	1733 ± 113
Clones Gerados	19277 ± 478	17822 ± 278	16636 ± 639
Tol./Efetor (%)	0.62 ± 0.08	0.62 ± 0.03	0.77 ± 0.08
Falsos Positivos (%)	2.58 ± 0.58	0.52 ± 0.15	0.40 ± 0.09
Falsos Negativos (%)	5.30 ± 0.61	6.32 ± 0.66	8.25 ± 0.86
Classificação Correta (%)	92.18 ± 0.98	93.17 ± 0.78	91.35 ± 0.79

Tabela 7.21: Avaliando DT-AIS com Segunda Base de Pacotes – Metodologia de Particionamento dos Dados

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**
Estimulações	640 ± 156	1701 ± 1203	2016 ± 765	2690 ± 476
Clones Gerados	14941 ± 1472	17392 ± 397	16748 ± 772	16921 ± 636
Tol./Efetor (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Positivos (%)	0.00 ± 0.00	0.00 ± 0.00	0.06 ± 0.13	0.16 ± 0.47
Falsos Negativos (%)	10.79 ± 3.02	5.76 ± 0.71	7.09 ± 1.52	6.76 ± 1.35
Class. Correta (%)	89.28 ± 2.80	94.21 ± 0.70	92.85 ± 1.48	93.12 ± 1.59

Ítem Observado	**Conf. 5**	**Conf. 6**	**Conf. 7**
Estimulações	1777 ± 82	2727 ± 487	1111 ± 170
Clones Gerados	16398 ± 653	14467 ± 1550	13530 ± 1353
Tol./Efetor (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Positivos (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Negativos (%)	7.39 ± 1.21	11.39 ± 3.07	13.04 ± 2.56
Classificação Correta (%)	92.59 ± 1.23	88.60 ± 3.08	86.96 ± 2.56

Os resultados obtidos, acima de 90% na taxa de detecção, apontam para o alto potencial do algoritmo como uma solução efetiva para detecção de varreduras de vulnerabilidades, podendo ser utilizado como uma barreira inicial contra tentativa de intrusões. Esse potencial é reforçado pelo alto grau de estabilidade, maior que o IA-AIS, nos resultados finais, mesmo com variações mais significativas nos parâmetros. De uma maneira mais simplista, pode-se dizer que os melhores resultados ocorreram em geral nas configurações com valores mais intermediários de tamanho de receptor e limiar de ativação.

É importante observar que, com a metodologia de seleção aleatória de dados de treinamento e classificação, a taxa de falsos positivos vai diminuindo à medida que se aumenta o limiar de ativação, com conseqüente aumento da taxa de falsos negativos, o que poderia ser utilizado como opção de configuração de um sistema baseado nesse algoritmo para obter um nível de reação maior ou menor do sistema, a critério do usuário. Esse comportamento não se manifestou com a metodologia de particionamento de dados, possivelmente pelo maior uso de dados para treinamento, o que praticamente fez com que o sistema não apresentasse falsos positivos em todas as configurações.

Uma informação interessante na Tabela 7.20 é o baixo nível de células tolerantes geradas, praticamente 1 célula desse tipo para cada 100 células efetoras. Usando a metodologia de k-pastas, esse valor foi ainda inferior, sendo praticamente nulo em termos percentuais, como pode ser visto na Tabela 7.21. Isso mostra um sistema altamente reativo, ao qual pode ser inte-

ressante acrescentar novos mecanismos regulatórios ou tolerizantes, como o uso de segunda ativação. A reatividade do sistema também pode ser verificada pelo alto número de clones gerados ao todo. É importante observar que, apesar do alto número de células geradas no sistema durante sua execução, a população final ficou em torno de 1500 linfócitos, o que permite apontar que não é necessário um grande número de células para a obtenção de níveis razoáveis de detecção.

7.3.4 DT-AIS no KDD-99

Os testes efetuados com o DT-AIS na segunda base de pacotes apresentaram resultados muito bons, superiores ao do IA-AIS. Isso por si só já permitiria apontá-lo como um algoritmo extremamente promissor para detecção de intrusos em redes de computadores, tornando desnecessário a aplicação do mesmo em outras bases de dados. Apesar disso, aplicou-se o DT-AIS ao KDD-99 apenas para verificar se o mesmo iria também apresentar um comportamento adequado com dados representados de forma diferenciada. Para implementação no AISF, adotou-se a configuração inicial já apresentada na Tabela 7.18, página 202. Também nessa base de dados, variou-se apenas os tamanhos dos receptores e o limiar de ativação, como apresentado na Tabela 7.22.

Tabela 7.22: Configurações do DT-AIS nos Testes com KDD-99

Parâmetro	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**
Tamanho do Receptor de APCs	15	20	15	20
Tamanho do Receptor de Linfócitos	20	30	15	20
Limiar de Ativação de APC	0.99	0.99	0.90	0.90
Limiar de Ativação de Linfócitos	0.98	0.98	0.80	0.80

Parâmetro	**Conf. 5**	**Conf. 6**	**Conf. 7**	**Conf. 8**
Tamanho do Receptor de APCs	25	20	20	25
Tamanho do Receptor de Linfócitos	25	20	20	25
Limiar de Ativação de APCs	0.90	0.95	0.80	0.95
Limiar de Ativação de Linfócitos	0.80	0.90	0.70	0.90

Como o objetivo era apenas validar o potencial do DT-AIS, os testes foram efetuados usando apenas uma variação do k-pastas, usando a mesma abordagem adotada no IA-AIS na metodologia de particionamento dos dados. Os dados foram particionados em oitenta partições, tendo sido utilizado em cada teste nove partições para treinamento e uma partição para teste, como já apre-

sentado na Tabela 7.8, página 196. Cada entrada do KDD-99 foi considerada como antígeno e seus valores numéricos foram também tratados como sinais. Os resultados obtidos com essa metodologia encontram-se apresentados na Tabela 7.23.

Tabela 7.23: Avaliando DT-AIS com KDD-99 – Metodologia de Particionamento dos Dados

Ítem Observado	**Conf. 1**	**Conf. 2**	**Conf. 3**	**Conf. 4**
Estimulações	4811 ± 13	4314± 1548	4713 ± 305	4683 ± 295
Clones Gerados	20503 ± 177	20434 ± 188	21212 ± 63	21211 ± 61
Tol./Efetor (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Positivos (%)	4.71 ± 0.67	4.80 ± 0.79	6.30 ± 0.14	6.29 ± 0.12
Falsos Negativos (%)	1.68 ± 0.36	2.06 ± 0.26	0.60 ± 0.14	0.61 ± 0.12
Class. Correta (%)	93.53 ± 0.82	93.15 ± 0.91	93.12 ± 0.13	93.11 ± 0.14

Ítem Observado	**Conf. 5**	**Conf. 6**	**Conf. 7**	**Conf. 8**
Estimulações	4752 ± 202	4810 ± 61	3714 ± 1318	4820 ± 30
Clones Gerados	21166 ± 111	21129 ± 112	20691 ± 418	20992 ± 153
Tol./Efetor (%)	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Falsos Positivos (%)	6.05 ± 0.34	6.11 ± 0.41	5.19 ± 0.68	5.85 ± 0.54
Falsos Negativos (%)	0.53 ± 0.22	0.81 ± 0.06	1.61 ± 1.09	0.94 ± 0.11
Classificação Correta (%)	93.41 ± 0.34	93.06 ± 0.37	93.16 ± 0.80	93.21 ± 0.49

Os testes apresentados na na Tabela 7.23 confirmam o grande potencial e a estabilidade do DT-AIS aplicado ao problema de detecção de intrusos. Ressalta-se o fato que novamente os testes apresentaram um baixo nível de células tolerantes geradas, inferiores a uma parte por mil, quando comparadas às células efetoras, confirmando a possibilidade de vantagens no desempenho com o acréscimo de mecanismos regulatórios ou tolerizantes. Além disso, apesar de grande variação em alguns dos parâmetros, o algoritmo manteve-se relativamente estável no processo de classificação, havendo apenas pequenas diferenças quanto ao tipo de erro efetuado, se falso positivo ou negativo. Mesmo essas diferenças foram relativamente pequenas, inferiores a dois pontos percentuais, tendo o algoritmo conseguido mais de 93% de acerto em todas as configurações.

Apesar do resultado obtido ser inferior àquela conseguido com o IA-AIS, o índice de acerto é muito bom, comparado aos encontrados na literatura. Além disso, como já ressaltado anteriormente, o KDD-99 é uma base que atualmente já não reflete a realidade, e os melhores resultados do DT-AIS com a base de pacotes apontam-no quando mais adequado quando existem sinais disponíveis no problema em questão. Inclusive, o fato de que as entradas do KDD-99 serviram ao mesmo tempo como antígeno (a entrada inteira) e sinais

(os valores numéricos) podem ter influenciado num menor desempenho do DT-AIS, quando comparado ao IA-AIS nessa base de dados. Devido à desatualização do KDD-99, não houve o interesse em melhor definir o que seriam sinais e antígenos nessa base de dados, uma vez que os resultados obtidos já mostram o DT-AIS como extremamente promissor para uso efetivo em detecção de intrusos em redes de computadores.

7.4 Discussão

Os resultados obtidos com as realizações dos testes com o IA-AIS e DT-AIS permitem verificar a extrema adequação do uso de metáforas mais abrangentes sobre o sistema imune no processo de detecção de intrusos. Essas metáforas permitiram obter uma alta taxa de detecção, mesmo em um ambiente relativamente "hostil", com dados extremamente variáveis, mas com poucas diferenças entre um pacote normal e um de ataque. Dessa maneira, os dois algoritmos mostraram bom desempenho em um ambiente bastante próximo a situações reais, indicando a possibilidade de implementação dos mesmos como parte de sistemas comerciais de detecção de intruso.

Sobre o IA-AIS, é importante observar que várias simplificações e modificações podem ser efetuadas, como o não uso de células T regulatórias, por exemplo. Outra possibilidade seria o uso de APCs, ao invés de macrófagos, com tratamento de sinais, permitindo o uso desse algoritmo quando o problema em questão possuir sinais associados a antígenos. Essas modificações precisam ser avaliadas com cuidado, verificando-se o impacto desses elementos no sistema como um todo. Após essas verificações, é nossa opinião ser necessário um estudo avaliando o desempenho do sistema com um baixo número de células, com vistas a obter uma melhor relação custo/benefício em termos de desempenho e eficiência.

Sobre o DT-AIS, existem fortes indícios que o sistema em sua forma original não seria capaz de detectar corretamente cerca de 8% das entradas da segunda base de pacotes, sem modificações no algoritmo em si. Uma modificação possível seria a inclusão de mecanismos de segunda ativação e regulação, numa abordagem similar ao IA-AIS. Isso, entretanto, necessita ser avaliado com extrema cautela, a fim de evitar um grande esforço computacional com poucos resultados práticos.

De qualquer forma, os resultados obtidos com o DT-AIS apontam-no como extremamente adequado para implementação, seja diretamente no sistema

operacional, seja como parte de algum IDS comercial. Sua relativa simplicidade mostra-o como eficiente o suficiente para uso real em sistemas atuais. Em geral, uma entrada é apresentada para cada célula do sistema uma única vez, o que aponta para uma complexidade $nlog(n)$, onde n é o número de células no sistema. Dado que o tamanho do receptor é bastante inferior ao número de células, o tamanho da população é o elemento mais crítico na eficiência do algoritmo e precisa ser avaliado severamente antes de uma adoção em ambientes de produção.

Antes desses testes populacionais, entretanto, é interessante avaliar os dois algoritmos em novos problemas e bases de dados, com o objetivo de obter uma visão geral mais completa dos mesmos. Isso deve ser feito com a intenção de verificar, entre outros aspectos, quais elementos devem ser valorizados com vistas a melhores desempenho e estabilidade, e quais configurações são mais adequadas a partir de informações do problema e do formato de representação dos dados.

Como já comentado anteriormente, além da distância de Hamming, foram utilizadas outras funções de afinidade, com resultados sofríveis. Isso mostra a extrema necessidade de adequação dos algoritmos imunoinspirados ao problema em questão, com a utilização de representações e medidas de afinidade adequadas, ao invés da adoção de estratégias por demais artificiais. Essa posição não é apenas nossa, tendo sido advogada recentemente por John Timmis, como pode ser verificado, por exemplo, em (FREITAS; TIMMIS, 2007), (TIMMIS *et al.*, 2008a) e (TIMMIS *et al.*, 2008b). Esses textos também apontam para a necessidade de proposições de algoritmos que utilizem uma visão mais completa do sistema imune, com maior embasamento biológico, um certo “retorno às origens” nas palavras desse pesquisador.

Apesar de não influenciado diretamente pelas opiniões de Timmis, somos obrigados a concordar com várias dessas afirmações, e de certa forma o trabalho aqui apresentado atende a várias de suas sugestões futuras de pesquisas. Em (FREITAS; TIMMIS, 2007), por exemplo, são propostos que os SIAs utilizem:

- uma representação híbrida dos dados, não ignorando os dados não-numéricos;
- funções de afinidade adequadas aos dados sendo avaliados, não apenas uma função padrão;
- mecanismos duplos para ativação celular e clonagem;

- bases de dados mais desafiadoras e mais próximas do mundo real.

Em (TIMMIS *et al.*, 2008a) e (HART; TIMMIS, 2008), por sua vez, é apontado com destaque que SIAs futuros irão apresentar homeostase, beneficiando-se da interação entre modelos inatos e adaptativos do sistema imune, com vários componentes sistema como um todo. Além disso, é afirmado que os componentes desses SIAs serão naturalmente distribuídos e que o sistema como um todo irá apresentar aprendizado de longa duração.

Em nossa opinião, o trabalho apresentado neste texto vai ao encontro dessas tendências, apontando em muito para as possibilidades de expansão de pesquisas nessa área. Acreditamos que os resultados concretos apresentados neste capítulo devem fomentar o estímulo de novos modelos de Computação Imunoinspirada, com o desenvolvimento de novos algoritmos utilizando uma visão mais completa do sistema imune. Além disso, por si só o trabalho já aponta para a possibilidade de implementação de mecanismos práticos de detecção de intrusos em sistemas computacionais reais.

Optou-se neste trabalho por não efetuar uma análise de sensibilidade dos parâmetros, uma vez que acreditamos que isso deverá ser feito somente após a aplicação dos algoritmos em outros problemas e situações, permitindo uma análise mais completa e detalhada dos mesmos. Isso também foi motivado pelo fato que os testes permitiram perceber que boa parte dos parâmetros é correlacionada, tornando uma tarefa relativamente complexa uma análise desse tipo sem a execução de uma grande bateria de testes. Esses testes são necessários justamente para verificar não apenas a sensibilidade, mas o quanto um parâmetro impacta o desempenho do outro.

Entretanto, os testes efetuados permitem apontar ao leitor qual a importância e o impacto parcial da maior parte dos parâmetros apresentados na Tabela 6.1, página 160, e na Tabela 6.2, página 173. Nos dois algoritmos, a população dos diferentes tipos celulares impacta o desempenho do algoritmo como um todo para um determinado perfil, em termos de reatibilidade e adaptabilidade do sistema às entradas sendo avaliadas.

Ampliando-se a população de macrófagos no IA-AIS, por exemplo, direciona-se o algoritmo para um reconhecimento de padrões em situações em que os mesmos estejam mais claramente separado em classes. Em contextos em que as classes associadas aos padrões não estejam bem definidas, é interessante uma maior atuação do sistema imune adaptativo, através de células B e T no algoritmo. Assim a proporção entre as populações de macrófagos e linfócitos irá direcionar a linearização da convergência geral do algoritmo: quanto

mais macrófagos, maior a tendência do algoritmo direcionar-se para classes linearmente separáveis.

Por sua vez, o caráter reativo do algoritmo é direcionado pela proporção entre linfócitos T Auxiliares e linfócitos T regulatórios. Quanto maior a população de linfócitos regulatórios, maior a chance de supressão de linfócitos B, tornando o algoritmo menos reativo às entradas sendo avaliadas. Quanto maior o número de linfócitos T disponíveis, por sua vez, maiores as chances de uma segunda ativação ocorrer, resultando em reação do sistema.

É importante destacar que o ajuste populacional exige uma calibragem bastante precisa, adequada ao problema em questão, uma vez que esse é o elemento que mais impacta na eficiência, mas também no velocidade de execução do algoritmo. Quanto mais células, maior a probabilidade de uma classificação correta, mas um maior número de comparações será efetuado, tornando o processo mais lento. Assim, é desejável que o tamanho populacional seja devidamente avaliado, para conseguir o máximo de desempenho possível com um mínimo de comparações.

A população também impacta o caráter adaptativo do algoritmo: a quantidade de células adicionadas por estimulação ou a cada iteração irá definir se o algoritmo tenderá para um comportamento mais repetitivo ou se irá se adequando paulatinamente ou abruptamente à novas entradas. A proporção desses valores com relação ao tamanho inicial das populações, bem como o tamanho máximo populacional, permitido irá impactar justamente na adaptabilidade como um todo do algoritmo. Isso por sua vez, é contrabalançado pelo TTL inicial de cada célula, bem como o TTL que é adicionado a células que foram ativadas: quanto maior esses valores, maior a tendência das células mais antigas se manterem no sistema.

Os parâmetros relacionados à clonagem, por outro lado, devem ser sintonizados de acordo com a variabilidade das entradas: quanto maiores esses valores, maior a tendência do algoritmo a classificar entradas relativamente similares como pertencendo a uma mesma classe. O número máximo de clones por célula ou por iteração também afeta a população de linfócitos B no sistema, impactando diretamente na reatibilidade do algoritmo. É importante observar que uma maior reatibilidade pode também implicar em um maior número de falsos positivos. Por sua vez, um sistema pouco reativo tende a produzir mais falsos negativos, o que fomenta a necessidade de um ajuste preciso desses valores, de acordo com o problema em questão.

Os últimos parâmetros a serem observados no IA-AIS, o limiar de ativação dos diferentes tipos celulares, devem ser calibrados de acordo com a função de ativação adotada e a variabilidade das entradas. Esses parâmetros são extremamente sensíveis ao problema em questão, como pode ser verificado nos testes com as diferentes bases de dados, devendo ser ajustados caso a caso. Cabe ressaltar que em geral, optou-se por um valor maior desse parâmetro para as células inatas, uma vez que metaforicamente esses elementos estão associados a padrões mais conservados, ou seja: padrões melhor definidos em classes no problema em questão. A idéia aqui é que o sistema adaptativo possua maior reatibilidade a elementos novos, possuindo, portanto, valores menores no limiar de ativiação.

As observações feitas para o IA-AIS valem em sua grande maioria, com os devidos ajustes, para o DT-AIS, como por exemplo a proporção entre populações de células do sistema inato e adaptativo impactar a reatibilidade do algoritmo, implicando na necessidade de um ajuste desses valores adequados ao problema em questão. Aqui também, a existência de um número menor de parâmetros torna menos complexa, mas não necessariamente fácil, a calibragem desses valores. Na verdade, foi possível observar no DT-AIS que o coeficiente de diferenciação dos sinais, γ, é muito pouco sensível no problema de detecção de intrusos, tendo sido descartada a sua variação nos testes efetuados. Como já alertamos, entretanto, é importante a realização de testes em outros problemas antes de um descarte final desse parâmetro.

7.5 Comentários Finais

Neste capítulo foram apresentados um conjunto de testes efetuados com os algoritmos IA-AIS e DT-AIS, com resultados bastante promissores. Esses resultados apontam para a contribuição do trabalho não só na área de Segurança Computacional, como também em Computação Imunoinspirada. Mais ainda, apontaram para a importância do uso de uma visão mais completa do sistema imune, sem a necessidade de metáforas "guerreiras" na proposição de novos modelos.

Além disso, o trabalho apresentado é de certa forma pioneiro, não só pelo nível de resultado conseguido, quando comparado a outros SIAs, mas por ir ao encontro de tendências futuras de pesquisa na área. O trabalho utiliza-se de metáforas inspiradas em uma visão holística do funcionamento do sistema imune, o que se mostrou bastante adequado ao problema de detecção de intru-

sos. Assim, mesmo que os algoritmos em si não sejam utilizados, acreditamos ser importante que projetistas de IDS busquem inspiração na Imunologia ao propor novos modelos e estratégias para a detecção de intrusos.

CAPÍTULO

8

Conclusão e Trabalhos Futuros

Depois, sobram a vida e o Ele:
dai-me, Senhor a calma perene
e a ternura plena de uma manhã de domingo.

Canto ao Ele 10, 1993

Eis o porco: ele está morto e temperado,
assado foi em vários espetos,
e agora espera o seu apetite.

Mas o porco não termina em si, existem outros porcos
que nasceram de sua semente suína
e esperam o dia de suas próprias fogueiras.

A Propósito de Eco, 2009

NESTE trabalho foram apresentados dois algoritmos imunoinspirados, IA-AIS e DT-AIS, aplicados ao problema de detecção de intrusos. Os modelos incluem elementos do sistema imune inato e adaptativo, bem como um relacionamento entre esses elementos. Os modelos propostos obtiveram altas taxas de classificação correta, possuindo grande potencial para uso efetivo em ambientes computacionais. Os algoritmos em si são relativamente simples, apesar de embasados em uma visão mais completa sobre o funcionamento do sistema imune, comparado aos modelos em uso atualmente pela comunidade acadêmica. Essa visão propicia vislumbrar um rico potencial para os mesmos,

bem como inúmeros trabalhos futuros envolvendo desde o algoritmo em si até sua aplicação prática.

Investigações futuras precisam ser feitas, por exemplo, para verificar modificações nos algoritmos, avaliando o impacto dos mecanismos tolerizantes ou regulatórios. Outra possibilidade em potencial é o desenvolvimento de novos modelos utilizando características dos dois algoritmos propostos neste trabalho. Além disso, atualmente encontramos em estudos sobre a possibilidade da incorporação da hipótese da resposta imunológica policlonal. Essa hipótese, proposta em (BERNASCONI; TRAGGIAI; LANZAVECCHIA, 2002) e discutida na Seção 4.3, estabelece que mesmo na ausência de antígenos específicos, células B de memória se diferenciam e proliferam em resposta a estímulos policlonais derivados de micróbios ou células T ativas. Acreditamos que seja interessante verificar o impacto dessa hipótese em variantes dos dois algoritmos, ou mesmo em variantes de algoritmos clássicos, como o CLONALG.

É possível prever que simplificações possam ser feitas nos algoritmos, especialmente no IA-AIS, antes de uma implementação efetiva em sistemas de produção. Após uma bateria mais severa de testes, explorando diferentes possibilidades, pode ser necessário um certo distanciamento da metáfora original, utilizando-se apenas os elementos estritamente necessários dos algoritmos para o processo de detecção de intrusos. Isso porque a metáfora imunológica, apesar de extremamente adequada a esse problema, pode ser aplicada de diferentes formas. Pode-se vislumbrar, por exemplo, o uso dessa metáfora tanto em um nível mais profundo, no processo de reconhecimento dos antígenos a um nível mais arquitetural, em que módulos de um IDS sejam construídos com base nos elementos do sistema imune. Ainda é cedo para afirmar qual o melhor nível de aplicação da metáfora, e talvez a melhor resposta a esse tipo de questionamento seja o uso de uma metáfora mais completa, indo desde o reconhecimento até a estrutura geral do sistema.

É nossa intenção implementar os algoritmos, ou uma variante, como módulo do *kernel* do Linux, após a realização de novos e exaustivos testes. Entre os testes necessários, encontram-se aqueles com conjuntos de dados incluindo novos tipos de ataque. Para isso, é interessante o desenvolvimento de *honeypots*, máquinas que serviriam apenas como "iscas" para potenciais invasores, permitindo a captura de suas tentativas de invasão, para posterior avaliação. Além disso, é importante verificar, após esses testes, qual o melhor valor para os tamanhos das populações celulares, utilizando uma ótica custo/benefício. Assim, apesar dos resultados extremamente positivos obtidos nos testes dos algoritmos, pensamos nos mesmos como bastante promisso-

res, mas precisando de uma maior exploração desse potencial antes de uma adoção em ambiente real.

A exploração desse potencial passa inclusive pelo uso dos algoritmos aqui propostos em outros problemas de detecção de falhas, com o objetivo de melhor explorar o desempenho do algoritmo nessa tarefa. Isso permitirá, inclusive uma melhor análise dos parâmetros desses algoritmos em sua eficiência e eficácia na resolução de problemas. Isso deve ser feito com a intenção de verificar, entre outros aspectos, quais elementos devem ser valorizados com vistas a melhores desempenho e estabilidade, e quais configurações são mais adequadas a partir de informações do problema e do formato de representação dos dados.

Entre os méritos deste trabalho, destaca-se o uso de uma visão mais p do sistema imune, uma necessidade atual na área de Algoritmos Imunoinspirados. Em nossa opinião, o sistema imune natural tem ainda muito a contribuir para o desenvolvimento de novos modelos de sistemas imunes artificiais. Essa visão motivou o desenvolvimento do AISF, um *framework* criado para permitir a implementação de modelos alternativos, mas usando uma abordagem similar, de uma forma relativamente prática. O *framework*, assim como a implementação dos algoritmos será disponibilizado à comunidade acadêmica em futuro próximo, após melhor documentação do mesmo, utilizando a licença CC-GPL.

A maior contribuição desse trabalho é obviamente a proposição e análise inicial dos dois algoritmos. Os resultados obtidos com a aplicação dos mesmos aponta para a viabilidade concreta do uso de metáforas inspiradas em uma visão mais completa do sistema imune, quando aplicadas ao problema de detecção de intrusos. Dessa maneira, o trabalho trás contribuições não apenas para a área de Computação Bioinspirada, mas também para a área de Segurança Computacional, por mostrar o rico potencial dos estudos trandisciplinares nessas áreas. Além disso, espera-se que o trabalho estimule uma série de novos estudos em modelos mais completos de algoritmos imunoinspirados por parte dos pesquisadores da área.

Além dos algoritmos e do *framework*, uma contribuição adicional foi a criação de conjuntos de dados envolvendo situações atuais de ataque. As bases coletadas durante a realização deste trabalho serão disponibilizadas ao público acadêmico, para possibilitar estudos comparativos mais atualizados na área de detecção de intrusos. A principal base utilizada pela grande maioria dos trabalhos ainda é o KDD99, que já encontra-se bastante defasado e

não oferece grandes desafios aos algoritmos aplicados na área. Cabe destacar que a coleta dessas bases foi motivada pelo desconhecimento da existência de bases públicas atualizadas e que pudessem ser utilizadas na pesquisa.

É importante observar que este trabalho foi desenvolvido sob a forte influência de uma experiência prática anterior deste pesquisador com administração de servidores e laboratórios em Linux. De certa forma, boa parte da metodologia e abordagens utilizadas, como não utilizar endereços IPs como parte de antígenos, foram inspirados por essa vivência próxima ao problema real. Isso fomentou a necessidade de potencial prático, tornando necessário a busca de modelos com grandes possibilidades de adoção a médio prazo, o que foi conseguido nesse trabalho.

Obviamente, do ponto de vista de pesquisador em Segurança Computacional, não é possível se iludir com soluções automatizadas. O uso de ferramentas computacionais é extremamente importante nessa tarefa, mas a segurança de um ambiente não depende apenas desse tipo de recurso. Em nossa atuação nessa área, foi possível verificar "*in loco*" a ocorrência de ataques e invasões, bem como o alto nível de vulnerabilidades na maioria dos computadores, incluindo-se servidores. Em boa parte dos casos, as vulnerabilidades existiam não pela inexistência de mecanismos computacionais de proteção, mas sim do desconhecimento dos usuários da existência desses mecanismos, ou mal-uso dos mesmos.

Infelizmente, é possível perceber que a segurança da informação ainda não é uma necessidade crítica por parte dos usuários, que em geral menosprezam os riscos existentes, em pró de uma maior comodidade. Esse fato ocorre não apenas com usuários normais de computadores, mas mesmo entre profissionais da área de tecnologia. Um elemento que consideramos mais crítico é o fato que a maioria dos cursos que formam os futuros profissionais da área de Computação darem pouca ou nenhuma atenção para a necessidade de formação nessa área. Por exemplo, programadores são formados, mesmo em boas escolas superiores, tendo tido apenas uma ou duas aulas sobre o desenvolvimento seguro de aplicações. Isso serve para estimular, indiretamente, toda uma fábrica de vulnerabilidades, uma vez que a segurança da informação não recebe o devido nível de exigência por parte dos usuários.

Dessa maneira, é importante ressaltar que, à parte das necessidades do desenvolvimento de novas ferramentas para segurança das informações, deve-se investir também em mecanismos e processos que forcem uma mudança, mesmo que pequena, por parte dos usuários. Essa é, obviamente, uma tarefa

complexa e complicada, mas que precisa ser iniciada para evitar problemas críticos no futuro. Caso a Internet não se torne um ambiente menos hostil, do ponto de vista da Segurança Computacional, o seu crescimento será em breve sufocado pelos problemas advindos das falhas computacionais.

Assim, este pesquisador considera que o trabalho apresentado é apenas uma entre várias velas que precisam ser acesas para diminuir o nível de "escuridão" de insegurança da rede. Como já comentado, mais e mais os ataques focam-se nos usuários finais, alguns desses ataques sendo relativamente "grosseiros" e podendo ser evitados com uma simples análise criteriosa. Alguns tipos de ataques, entretanto, ultrapassam esse nível, exigindo realmente o desenvolvimento de novas ferramentas e metodologias de proteção. Esperamos, portanto que este trabalho possibilite o desenvolvimento de novas e mais completas ferramentas computacionais de segurança, permitindo que a Internet continue sendo um rico espaço para criação e debates de idéias, estendendo as fronteiras da comunicação humana.

Referências Bibliográficas

ABBAS, A. K.; LICHTMAN, A. H.; POBER, J. S. *Imunologia Celular e Molecular*. 4. ed. Rio de Janeiro: Revinter, 2003.

ABRAHAMSOHN, I. d. A. Células e Órgãos do Sistema Imune. In: CALICH, V.; VAZ, C. (Ed.). *Imunologia*. Rio de Janeiro: Revinter, 2001. cap. 1, p. 31–54.

AICKELIN, U.; BENTLEY, P. J.; CAYZER, S.; KIM, J.; MCLEOD, J. Danger Theory: The Link Between AIS and IDS? In: TIMMIS, J.; BENTLEY, P. J.; HART, E. (Ed.). *Artificial Immune Systems, Second International Conference, ICARIS 2003, Edinburgh, UK, september 1-3, 2003, Proceedings*. Edinburgh: Springer, 2003. (Lecture Notes in Computer Science, v. 2787), p. 147–155.

AICKELIN, U.; DASGUPTA, D. Artificial immune systems. In: BURKE, E. K.; KENDALL, G. (Ed.). *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. London: Springer, 2006. cap. 13, p. 375–99.

AICKELIN, U.; GREENSMITH, J. Sensing Danger: Innate Immunology for Intrusion Detection. *Information Security Technical Reports*, Elsevier, v. 12, n. 4, p. 218–227, 2007. Disponível em: `http://ima.ac.uk/publications`.

AICKELIN, U.; GREENSMITH, J.; TWYCROSS, J. Immune system approaches to intrusion detection - a review. In: NICOSIA, G.; CUTELLO, V.; BENTLEY, P. J.; TIMMIS, J. (Ed.). *Artificial Immune Systems, Third International Conference, ICARIS 2004, Catania, Sicily, Italy, september 13-16, 2004*. Catania: Springer, 2004. (Lecture Notes in Computer Science, v. 3239), p. 316–329.

ANDREWS, P. S.; TIMMIS, J. Inspiration for the Next Generation of Artificial Immune Systems. In: JACOB, C.; PILAT, M. L.; BENTLEY, P. J.; TIMMIS, J.

(Ed.). *Artificial Immune Systems: 4th International Conference, ICARIS 2005, Banff, Alberta, Canada, August 14-17, 2005.* Berlin: Springer-Verlag, 2005. (Lecture Notes in Computer Science, v. 3627), p. 126–38.

ANONYMOUS. *Maximum Linux Security: A Hacker's Guide to Protecting Your Linux Server and Workstation*. Indianapolis: Sams, 2000.

ARANO, T. *IPv4 Address Report*. 27 out. 2008. WWW. Disponível em: `http://www.potaroo.net/tools/ipv4/index.html`. Acesso em: 27/10/2008.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. *ABNT NBR ISO/IEC 27002*: Tecnologia da informação – técnicas de segurança – código de prática para a gestão da segurança da informação. Rio de Janeiro, 31 ago. 2005. 120 p.

BARAN, P. On Distributed Communications: I Introduction to Distributed Communications Networks. *IEEE Trans. Comm. Systems*, ago. 1964.

BARAN, P. *Interview 378*. 24 out. 1999. IEEE History Center The Institute of Electrical and Electronics Engineers, Inc. (WWW). Disponível em: `http://www.ieee.org/portal/cms_docs/iportals/aboutus/history_center/oral_history/pdfs/Baran378.pdf`.

BARBOSA, B. H. *Seminários de Sistemas Nebulosos: Autômatos Celulares*. Belo Horizonte, 2004.

BARTON, G. M.; MEDZHITOV, R. Control of Adaptive Immune Responses by Toll-like Receptors. *Current Opinion in Immunology*, v. 14, p. 380–3, 2002.

BATISTA, L. d. S. *Investigação de Novas Abordagens em Sistemas Imunes Artificiais para Otimização*. Dissertação (Dissertação de Mestrado) — Escola de Engenharia da Universidade Federal de Minas Gerais, Belo Horizonte, 6 abr. 2009.

BAUTTS, T.; DAWSON, T.; PURDY, G. N. *Linux Network Administrator's Guide*. 3. ed. Sebastopol: O'Reilly, 2005.

BEER, R. D. Autopoiesis and Cognition in the Game of Life. *Artificial Life*, n. 10, p. 309–26, 2004.

BERNARDES, A. T.; SANTOS, R. M. Z. dos. Immune Network at the Edge of Chaos. v. 186, p. 173–87, 1997.

BERNASCHI, M.; CASTIGLIONE, F. Design and Implementation of an Immune System Simulator. *Computers in Biology and Medicine*, Pergamon, v. 31, n. 5, p. 303–31, 2001.

BERNASCHI, M.; CASTIGLIONE, F.; SUCCI., S. An High Performance Simulator of the Immune Response. *Future Generation Computer Systems*, Elsevier Science, v. 15, n. 3, p. 333–42, 1999.

BERNASCHI, M.; SUCCI, S. Large-scale Cellular Automata Simulations of the Immune System Response. *Physical Review E*, The American Physical Society, v. 61, n. 2, p. 1851–4, fev. 2000.

BERNASCONI, N.; ONAI, N.; LANZAVECCHIA, A. A Role for Toll-like Receptors in Acquired Immunity: Up-regulation of TLR9 by BCR Triggering in Naive B Cells and Constitutive Expression in Memory B Cells. *Blood*, v. 101, n. 11, p. 4500–4, jun. 2003.

BERNASCONI, N.; TRAGGIAI, E.; LANZAVECCHIA, A. Maintenance of Serological Memory by Polyclonal Activation of Human Memory B Cells. *Science*, v. 298, n. 5601, p. 2199–202, dez. 2002.

BERNERS-LEE, T. WWW: Past, Present and Future. *IEEE Computer*, v. 29, n. 10, p. 69–77, out. 1996.

BEZERRA, G. B.; BARRA, T. V.; FERREIRA, H. M.; KNIDEL, H.; DE CASTRO, L. N.; VON ZUBEN, F. J. An Immunological Filter for Spam. In: BERSINI, H.; CARNEIRO, J. (Ed.). *Artificial Immune Systems: 5th International Conference, ICARIS 2006, Oeiras, Portugal, September 4-6, 2006, Proceedings*. Oeiras: Springer, 2006. (LNCS, v. 4163), p. 446–458.

BIERMANN, E.; CLOETE, E.; VENTER, L. M. A Comparison of Intrusion Detection Systems. *Computers & Security*, v. 20, n. 8, p. 676–683, 2001.

BOCK, G.; GOODE, J. (Ed.). *Immunoinformatics The New Kid in Town*. New Jersey: John Wiley & Sons, 2003. (Novartis Foundation Symposium, v. 254).

BONISSONE, P. P. Soft computing: The convergence of emerging reasoning technologies. *Soft Computing*, v. 1, n. 1, p. 6–18, 1997.

BRADEN, R. (Ed.). *Requirements for Internet Hosts – Communication Layers*. Internet Engineering Task Force (IETF), out. 1989. (Request for Comments: 1122, STD0003). Disponível em: `http://www.ietf.org/`.

BRAGA, A. de P.; CARVALHO, A. C. P. d. L. F.; LUDERMIR, T. B. *Redes Neurais Artificiais: Teoria e Aplicações*. Rio de Janeiro: LTC, 2000.

BRAGA, A. de P.; CARVALHO, A. C. P. d. L. F.; LUDERMIR, T. B. *Redes Neurais Artificiais: Teoria e Aplicações*. Rio de Janeiro: LTC, 2000.

BRASIL. Decreto-Lei No. 2.848, de 7 de Dezembro de 1940: Código Penal. *Diário Oficial da União*, 31 dez. 1940. Disponível em: `http://www.planalto.gov.br/CCIVIL/Decreto-Lei/Del2848compilado.htm`.

BRUSIC, V.; PETROVSKY, N. Immunoinformatics The New Kid in Town. In: BOCK, G.; GOODE, J. (Ed.). *Imunoinformatics: Bioinformatic Strategies for Better Understanding Of Immune Function*. New Jersey: John Wiley & Sons, 2003, (Novartis Foundation Symposium, v. 254). p. 3–22.

BURGISS, H. *Security Quick-Start HOWTO for Linux, v.1.2*. 1 jul. 2002. The Linux Documentation Project [WWW]. Disponível em: `http://www.tldp.org/HOWTO/text/Security-Quickstart-HOWTO`. Acesso em: 15/05/2008.

BURGISS, H. *Security Quick-Start HOWTO for Red Hat Linux, v. 1.2*. 21 jul. 2002. The Linux Documentation Project [WWW]. URL: `http://www.ibiblio.org/pub/Linux/docs/HOWTO/Security-Quickstart-Redhat-HOWTO`. Disponível em: `http://www.tldp.org/HOWTO/text/Security-Quickstart-Redhat-HOWTO`. Acesso em: 15/05/2008.

BURNET, F. *The Clonal Selection Theory of Acquired Immunity*. Cambridge: Cambridge Press, 1959. Cambridge Press.

BUSH, V. As we may think. *Atlantic Monthly*, v. 176, n. 1, p. 101–108, jul. 1945. Disponível em: `http://www.theatlantic.com/doc/194507/bush`.

CALICH, V.; VAZ, C. (Ed.). *Imunologia*. Rio de Janeiro: Revinter, 2001.

CAMPELO, F.; AES, F. G.; IGARASHI, H.; RAMIREZ, J. A clonal selection algorithm for optimization in electromagnetics. *Magnetics, IEEE Transactions on*, v. 41, n. 5, p. 1736–1739, May 2005. ISSN 0018-9464.

CAMPELO, F.; NOGUCHI, S.; IGARASHI, H. A new method for the robust design of high field, highly homogenous superconducting magnets using an immune algorithm. *Applied Superconductivity, IEEE Transactions on*, v. 16, n. 2, p. 1316–1319, June 2006. ISSN 1051-8223.

CAMPELO, F.; WATANABE, K.; IGARASHI, H. 3D topology optimization using an immune algorithm. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, v. 26, n. 3, p. 677–688, 2007.

CARNEIRO, J.; COUTINHO, A.; FARO, J.; STWART, J. A Model of The Immune Network with B-T Cell Co-operation. I – Prototypical Structures and Dynamics. v. 182, p. 513–29, 1996.

CARNEIRO, J.; COUTINHO, A.; FARO, J.; STWART, J. A Model of The Immune Network with B-T Cell Co-operation. II – The Simulation Of Ontogenesis. v. 182, p. 531–47, 1996.

CARRANO, E.; AES, F. G.; TAKAHASHI, R.; NETO, O.; CAMPELO, F. Electric distribution network expansion under load-evolution uncertainty using an immune system inspired algorithm. *Power Systems, IEEE Transactions on*, v. 22, n. 2, p. 851–861, May 2007. ISSN 0885-8950.

CARVALHO, M. S. R. M. de. *A Trajetória da Internet no Brasil: do Surgimento das Redes de Computadores À instituição dos Mecanismos de Governança*. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro - UFRJ, RJ, set. 2006.

CASTELLS, M. *A Sociedade em Redes*. São Paulo: Paz e Terra, 2001.

CASTIGLIONE, F.; TOSCHI, F.; BERNASCHI, M.; SUCCI, S.; BENEDETTI, R.; FALINI, B.; LISO, A. Computational Modeling of the Immune Response to Tumor Antigens. *Journal of Theoretical Biology*, Elsevier, v. 21, jul. 2005.

CELADA, F.; SEIDEN, P. E. A Computer Model of Cellular Interactions in the Immune System. *Immunology Today*, v. 13, n. 2, p. 56–62, 1992.

CERF, V. G.; KAHN, R. E. A Protocol for Paket Network Intercommunication. *IEEE Transctions on Communications*, com-22, n. 5, p. 627–641, maio 1974.

CHAKRABORTY, A. K.; DUSTIN, M. L.; SHAWN, A. S. In Silico Models for Cellular and Molecular Immunology: Successes, Promises and Chalenges. *Nature Immunology*, v. 4, n. 10, p. 933–6, out. 2003.

CHAO, D. L. *Modeling the cytotoxic T cell response*. Tese (Doutorado) — University of New Mexico, Albuquerque, New Mexico, dez. 2004.

CHAO, D. L.; DAVENPORT, M. P.; FORREST, S.; PERELSON, A. S. Stochastic Stage-structured Modeling of the Adaptive Immune System. In: *Proceedings of the Computational Systems Bioinformatics (CSB'03)*. Los Alamitos, California: IEEE, 2003. p. 124–131.

CHAO, D. L.; DAVENPORT, M. P.; FORREST, S.; PERELSON, A. S. A Stochastic Model of Cytotoxic T Cell Responses. *Journal of Theoretical Biology*, Elsevier, v. 228, p. 227–40, 2004.

CHESWICK, W. R.; BELLOVIN, S. M.; RUBIN, A. D. *Firewalls e Segurança na Internet: Repelindo o hacker ardiloso*. 2. ed. Porto Alegre: Bookman, 2005.

CHURCHLAND, P. S.; SEJNOWSKI, T. J. *The Computational Brain*. Massachusetts: MIT Press, 1996.

COHEN, I. R. Discrimination and Dialogue in The Immune System. *Seminars in Immunology*, v. 12, n. 3, p. 215–9, 2000.

COMER, D. E. *Redes de Computadores e Internet: Abrange Transmissão de Dados, Ligações Inter-Redes, web e Aplicações*. 4. ed. Porto Alegre: Bookman, 2007.

COPELLI, M.; SANTOS, R. M. Z. dos; STARIOLO, D. A. On the Aging Dynamics in an Immune Network Model. *European Physical Journal B*, v. 34, n. 1, p. 119–29, jul. 2003.

COUTINHO, A. Beyond Clonal Selection and Network. *Immunological Reviews*, v. 110, p. 63–87, ago. 1989.

COUTINHO, A. Will the idiotypic network help to solve natural tolerance? *Trends in Immunology*, v. 24, n. 2, p. 53–54, Feb 2003.

DASGUPTA, D. (Ed.). *Artificial Immune Systems and Their Applications*. London: Springer, 1998.

DASGUPTA, D. Advances in Artificial Immune Systems. *IEEE Computational Inteligence Magazine*, v. 1, n. 4, p. 40–49, nov. 2006.

DASGUPTA, D.; ATTOH-OKINE, N. Immunity-based systems: a survey. In: *IEEE International Conference on Systems, Man and Cybernetics*. Orlando: IEEE, 1997.

DE CASTRO, L. N. *Engenharia imunológica: desenvolvimento e aplicação de ferramentas computacionais inspiradas em sistemas imunológicos artificiais*. Tese (tese de doutorado) — DCA/FEEC/Unicamp, Campinas, 2001.

DE CASTRO, L. N. Immune, Swarm, and Evolutionary Algorithms - Part I: Basic Models. In: *Proc. Of The ICONIP Conference (International Conference on Neural Information Processing), Workshop on Artificial Immune Systems*. Singapura: IEEE, 2002. v. 3, p. 1464–8.

DE CASTRO, L. N. Immune, Swarm, and Evolutionary Algorithms - Part II: Philosophical Comparison. In: *Proc. Of The ICONIP Conference (International*

Conference on Neural Information Processing), Workshop on Artificial Immune Systems. Singapura: IEEE, 2002. v. 3, p. 1469–73.

DE CASTRO, L. N. *Fundamentals of Natural Computing*. Boca Raton: Chapman & Hall, 2006.

DE CASTRO, L. N. Immunocomputing. In: *Fundamentals of Natural Computing*. New York: Chapman & Hall, 2006. cap. 6, p. 267–323.

DE CASTRO, L. N.; TIMMIS, J. An artificial immune network for multimodal function optimization. In: *Proceedings of IEEE Congress on Evolutionary Computation (CEC'02)*. Hawaii: IEEE, 2002. v. 1, p. 669–674.

DE CASTRO, L. N.; TIMMIS, J. *Artificial Immune Systems: A New Computational Intelligence Approach*. London: Springer, 2002.

DE CASTRO, L. N.; TIMMIS, J. Artificial Immune Systems: a novel paradigm to pattern recognition. In: CORCHADO, J. M.; ALONSO, L.; FYFE, C. (Ed.). *Artificial Neural Networks in Pattern Recognition*. Paisley (UK): University of Paisley, 2002. p. 67–84.

DE CASTRO, L. N.; TIMMIS, J. I. Artificial immune system as a novel soft computing paradigm. *Soft Computing*, v. 7, n. 8, p. 526–44, 2003.

DE CASTRO, L. N.; VON ZUBEN, F. J. *Artificial Immune Systems: Part i – basic theory and applications [technical report]*. Campinas: DCA / Unicamp, 1999. (TR DCA 01/99).

DE CASTRO, L. N.; VON ZUBEN, F. J. *Artificial Immune Systems: Part ii – a survey of applications [technical report]*. Campinas: DCA / Unicamp, 2000. (TR DCA 02/00).

DE CASTRO, L. N.; VON ZUBEN, F. J. An evolutionary immune network for data clustering. In: *Proc. of the IEEE SBRN (Brazilian Symposium on Artificial Neural Networks)*. Rio de Janeiro: SBRN, 2000. p. 84–89.

DE CASTRO, L. N.; VON ZUBEN, F. J. An Immunological Approach do Initialize Centers of Radial Basis Function Neural Networks. In: *Proceedings of V Brasilian Conference on Neural Networks - V Congresso Brasileiro de Redes Neurais*. Rio de Janeiro: SBRN, 2001. p. 79–84.

DE CASTRO, L. N.; VON ZUBEN, F. J. An Immunological Approach to Initialize Feedforward Neural Network Weights. In: *Proc. of ICANNGA (Int. Conf. on Artificail Neural Networks and Genetic Algorithms)*. Prague: Springer, 2001. p. 126–129.

DE CASTRO, L. N.; VON ZUBEN, F. J. Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems*, IEEE, v. 6, p. 239–251, 2002.

DE CASTRO, L. N.; VON ZUBEN, F. J.; DEUS JR., G. A. de. The construction of a Boolean competitive neural network using ideas from immunology. *Neurocomputing*, v. 50, p. 51–85, jan. 2003.

DEERING, S.; HINDEN, R. *Internet Protocol, Version 6 (IPv6) Specification*. Internet Engineering Task Force (IETF), dez. 1998. (Request for Comments: 2460). Disponível em: `http://www.ietf.org/rfc/rfc2640.txt`.

DUBOIS, D.; PRADE, H. Soft computing, fuzzy logic, and artificial intelligence. *Soft Computing*, v. 2, n. 1, p. 7–11, 1998.

DURBIN, R.; MIALL, C.; MITCHISON, G. (Ed.). *The Computing Neuron*. Redwood: Addison-Wesley, 1991.

ERCILIA, M. *A Internet*. São Paulo: Publifolha, 2000.

ERLICH, P. On immunity with special reference to cell life. In: BIDEL, D. J. (Ed.). *Milestones in Immunology: A Historical Exploration*. Berlin: Springer-Verlag, 1988. p. 166–169. Ano de Publicação Original do Artigo: 1900.

FANELLI, R. L. A hybrid model for immune inspired network intrusion detection. In: BENTLEY, P. J.; LEE, D.; JUNG, S. (Ed.). *Artificial Immune Systems, 7th International Conference, ICARIS 2008, Phuket, Thailand, August 10-13, 2008. Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2008. p. 107–118.

FARMER, J. D.; PACKARD, N.; PERELSON, A. The immune system, adaptation and machine learning. *Physica D*, v. 22, p. 187–204, 1986.

FAUSETT, L. *Fundamentals of Neural Networks*. New Jersey: Prentice-Hall, 1994.

FÉLIX, R.; USHIO, T. Rough Lymphocytes for Approximate Binding in Artificial Immune Systems. In: *Proceedings of the 15th International Conference on Electronics, Comunications and Computers (CONIELECOMP 2005)*. Puebla (Mexico): IEEE, 2005. p. 272–277.

FENZI, K. *Linux Security HOWTO, v. 2.3*. 22 jan. 2004. The Linux Documentation Project [WWW]. URL: `http://www.ibiblio.org/pub/`

Linux/docs/HOWTO/Security-HOWTO. Disponível em: http://www.tldp.org/HOWTO/text/Security-HOWTO. Acesso em: 15/05/2008.

FERRUGEM, A. P.; LUCAS, D. C.; RODRIGUES, M.; BARONE, D. A. C. Autômatos Celulares. In: BARONE, D. (Ed.). *Sociedades Artificiais*. Porto Alegre: Bookman, 2003. cap. 3, p. 41–60.

FITZGERALD, J.; DENNIS, A. *Comunicações de Dados Empresariais e Redes*. 7. ed. Rio de Janeiro: LTC, 2005.

FORREST, S.; BEAUCHEMIN, C. Computer Immunology. *Immunological Reviews*, v. 216, n. 1, p. 176–97, abr. 2007.

FORREST, S.; HOFMEYR, S. A. Immunology as Information Processing. In: SIEGEL, L. A.; COHEN, I. R. (Ed.). *Design Principles for the Immune System and Other Distributed Autonomous Systems*. New York: Oxford University Press, 2001, (Santa Fe Institute Studies in the Sciences of Complexity). p. 361–87.

FORREST, S.; HOFMEYR, S. A.; SOMAYAJI, A. Computer Immunology. *Communications of the ACM*, v. 40, n. 10, p. 88–96, 1997. Disponível em: http://citeseer.ist.psu.edu/forrest96computer.html.

FORREST, S.; PERELSON, A. S.; ALLEN, L.; CHERUKURI, R. A.S. Perelson, L. Allen, R. and Cherukuri. In: *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy*. Los Angeles: IEEE Computer Society Press, 1994. p. 202–12.

FRAMPTON, S. *Linux Administration Made Easy*. [S.l.]: The Linux Documentation Project, 1999. URL: http://www.tldp.org/guides.html.

FREITAS, A. A.; TIMMIS, J. Re-visiting the Foundations of Artificial Immune Systems for Data Mining. *IEEE Transactions on Evolutionary Computation*, v. 11, n. 4, p. 521–40, 2007. Disponível em: http://www-users.cs.york.ac.uk/jtimmis/pubs.html.

FURLAN, L. B.; GAGLIARDI, H. F.; ALVES, D. Estudo do Conceito de Memória Imunológica em um Modelo para a Competição e Co-evolução de Microparasitas e o Sistema Imune de um Hospedeiro. In: *SBC2004 – XXIV Congresso da Sociedade Brasileira de Computação*. Salvador: SBC/UFBA, 2004.

GARCIA, A. C. B.; SICHMAN, J. S. Agentes e Sistemas Multiagentes. In: REZENDE, S. O. (Ed.). *Sistemas Inteligentes - Fundamentos e Aplicações*. Barueri: Manole, 2003. p. 269–306.

GARRETT, S. M. A Paratope is Not an Epitope: Implications for Immune Network Models and Clonal Selection. In: TIMMIS, J.; BENTLEY, P. J.; HART, E. (Ed.). *Artificial Immune Systems, Second International Conference, ICARIS 2003, Edinburgh, UK*. New York: Springer, 2003. (Lecture Notes in Computer Science, v. 2787), p. 217–28.

GÓMEZ, J.; GONZÁLEZ, F.; DASGUPTA, D. An Immuno-Fuzzy Approach to Anomaly Detection. In: *Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZIEEE)*. St. Louis: IEEE, 2003. v. 2, p. 1219–1224.

GONT, F. *Security Assessment of Internet Protocol*. CPNI, jul. 2008. 63 p. Disponível em: `http://www.cpni.gov.uk/WhatsNew/3680.aspx`.

GREENSMITH, J. *New Frontiers For An Artificial Immune System*. Dissertação (MSc Thesis) — School of Computing, University of Leeds, Hewlett-Packard Labs, Bristol, UK, 5 set. 2003.

GREENSMITH, J.; AICKELIN, U. Dendritic Cells for SYN Scan Detection. In: LIPSON, H. (Ed.). *Genetic and Evolutionary Computation Conference, GECCO 2007, Proceedings*. London: ACM, 2007. p. 49–56.

GREENSMITH, J.; AICKELIN, U.; CAYZER, S. Introducing Dendritic Cells as a Novel Immune-Inspired Algorithm for Anomaly Detection. In: *Proceedings of the 4th International Conference on Artificial Immune Systems (ICARIS 2005)*. Banff, Canada: Springer-Verlag, 2005. (LNCS, 3627), p. 153–167.

GREENSMITH, J.; AICKELIN, U.; CAYZER, S. Detecting Danger: The Dendritic Cell Algorithm. In: SCHUSTER, A. (Ed.). *Robust Intelligent Systems*. London: Springer, 2008. cap. 5, p. 89–112.

GREENSMITH, J.; AICKELIN, U.; TWYCROSS, J. Articulation and Clarification of the Dendritic Cell Algorithm. In: *Proceedings of the 5th International Conference on Artificial Immune Systems (ICARIS 2006), Oeiras, Portugal*. Oeiras: Springer, 2006. (LNCS, v. 4163), p. 404–417.

GREENSMITH, J.; TWYCROSS, J.; AICKELIN, U. Dendritic Cells for Anomaly Detection. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2006), Vancouver, Canada*. Vancouver: IEEE, 2006. p. 664–671.

GRILO, A.; CAETANO, A.; ROSA, A. Agent Based Artificial Immune System. In: GOODMAN, E. D. (Ed.). *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers (Proc. GECCO 2001, Vol. LBP)*. San Francisco, California, USA: [s.n.], 2001. p. 145–151.

GROOT, A. S. de; SBAI, H.; AUBIN, C. S.; MCMURRY, J.; MARTIN, W. Immuno-informatics: Mining Genomes for Vaccine Components. *Immunology and Cell Biology*, v. 80, p. 255–69, 2002.

GU, F.; GREENSMITH, J.; AICKELIN, U. Further exploration of the dendritic cell algorithm: Antigen multiplier and time windows. In: BENTLEY, P. J.; LEE, D.; JUNG, S. (Ed.). *Artificial Immune Systems, 7th International Conference, ICARIS 2008, Phuket, Thailand, August 10-13, 2008. Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2008. p. 142–153.

GUIMARÃES, F. G.; PALHARES, R. M.; CAMPELO, F.; IGARASHI, H. Design of mixed H_2/H^∞ control systems using algorithms inspired by the immune system. *Information Sciences*, Elsevier Science Inc., New York, NY, USA, v. 177, n. 20, p. 4368–4386, 2007. ISSN 0020-0255.

GUZELLA, T. dos S.; UCHÔA, J. Q.; SANTOS, T. A. M.; CAMINHAS, W. M. Proposta de um Modelo de Classificação de Padrões Baseado no Sistema Imune: uma Aplicação para a Identificacao de SPAM. In: UFRN/SBRN. *CBRN 2005 - VII Congresso Brasileiro de Redes Neurais*. Natal, 2005. v. 1.

GUZELLA, T. S.; MOTA-SANTOS, T. A.; UCHÔA, J. Q.; CAMINHAS, W. M. Identification of SPAM messages using an approach inspired on the immune system. *Biosystems*, v. 92, n. 3, p. 215–225, jun. 2008.

HART, E.; TIMMIS, J. Application areas of AIS: the past, the present and the future. *Appl. Soft Comput.*, v. 8, n. 1, p. 191–201, 2008.

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

HAYKIN, S. *Redes Neurais: Princípios e Prática*. Porto Alegre: Bookman, 2001.

HERZOG, P. *OSSTMM 3 Lite: Introdution and Sample to the Open Source Security Testing Methodology Manual*. ISECOM (Institute for Security and Open Methodologies), ago. 2008. Disponível em: `http://www.isecom.org/osstmm/`.

HETTICH, S.; BAY, S. D. *The UCI KDD Archive*. Irvine, CA: University of California, Department of Information and Computer Science., 1999. WWW. Disponível em: `http://kdd.ics.uci.edu/`.

HOFMEYR, S. A. *An Immunological Model of Distributed Detection and its Application to Network Security*. Tese (Doutorado) — University of New Mexico, New Mexico, maio 1999. Disponível em: `http://www.cs.unm.edu/~steveah/steve_diss.pdf`.

HOFMEYR, S. A.; FORREST, S. Architecture for an Artificial Immune System. *Evolutionary Computation*, v. 7, n. 1, p. 1289–1296, 2000. Disponível em: `http://citeseer.ist.psu.edu/hofmeyr00architecture.html`.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. 2. ed. Massachusetts: MIT Press, 1992.

HOOD, L.; PERLMUTTER, R. M. The Impact Of Systems Approaches On Biological Problems in Drug Discovery. *Nature Biotechnology*, v. 22, n. 10, p. 1215–7, out. 2004.

JANEWAY, C. A.; TRAVERS, P.; WALPORT, M.; SHLONMCHIK, M. *Immunobiology: The Immune System In Health and Disease*. 5. ed. New York: Garland Publishing, 2001. Possui tradução (JANEWAY *et al.*, 2002).

JANEWAY, C. A.; TRAVERS, P.; WALPORT, M.; SHLONMCHIK, M. *Imunobiologia: O Sistema Imune na Saúde e na Doença*. 5. ed. Porto Alegre: Artes Médicas, 2002.

JANEWAY JR., C. How the immune system works to protect the host from infection: A personal view. *PNAS*, v. 98, n. 13, p. 7461–8, Jun 19 2001.

JANEWAY JR., C.; MEDZHITOV, R. Innate Immune Recognition. *Annu. Rev. Immunol.*, v. 20, p. 197–216, 2002.

JENNER, E. An inquiry into the causes and effects of the variolae vaccine. In: BIDEL, D. J. (Ed.). *Milestones in Immunology: A Historical Exploration*. Berlin: Springer-Verlag, 1988. p. 16–19. Ano de Publicação Original do Artigo: 1798.

JERNE, N. K. The Immune System. *Sci. Am.*, v. 229, n. 1, p. 52–60, 1973.

JERNE, N. K. Towards a Network Theory Of The Immune System. *Ann. Immunol. (Inst. Pasteur)*, v. 125C, n. 1-2, p. 373–89, 1974.

JI, Z.; DASGUPTA, D. Estimating the detector coverage in a negative selection algorithm. In: BEYER, H.-G.; O'REILLY, U.-M. (Ed.). *Proceedings of Genetic and Evolutionary Computation Conference, GECCO 2005*. Washington DC: ACM, 2005. p. 281–288.

JI, Z.; DASGUPTA, D. Applicability issues of the real-valued negative selection algorithms. In: CATTOLICO, M. (Ed.). *Proceedings of enetic and Evolutionary Computation Conference, GECCO 2006*. Seattle: ACM, 2006. p. 111–118.

JI, Z.; DASGUPTA, D. Revisiting Negative Selection Algorithms. *Evolutionary Computation*, v. 15, n. 2, p. 223–251, 2007.

KAYACIK, H. G.; ZINCIR-HEYWOOD, A. N.; HEYWOOD, M. I. A hierarchical som-based intrusion detection system. *Engineering Applications of Artificial Intelligence*, v. 20, n. 4, p. 439 – 451, 2007. ISSN 0952-1976.

KEPHART, J. O. A biologically inspired immune systems for computers. In: BROOKS, R. A.; MAES, P. (Ed.). *Artificial Life IV Proc. of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*. Massachusetts: MIT Press, 1994. p. 130–139.

KIM, D. H.; CHO, J. H. Intelligent Tuning of PID Controller With Disturbance Function Using Immune Algorithm. In: *2004 Annual Meeting of the North American Fuzzy Information Processing Society - NAFIP 2004*. [S.l.]: IEEE, 2004. v. 1, p. 286–91.

KIM, D. H.; LEE, H. Intelligent Control of Nonlinear Power Plant Using Immune Algorithm Based Multiobjectiv Optimization. In: *Proceedings of The 2004 IEEE International Conference on Networking, Sensing & Control*. Taipei, Taiwan: IEEE, 2004. p. 1388–1393.

KIM, J.; BENTLEY, P. J.; AICKELIN, U.; GREENSMITH, J.; TEDESCO, G.; TWYCROSS, J. Immune System Approaches to Intrusion Detection - a Review. *Natural Computing*, v. 6, n. 4, p. 413–466, jan. 2007.

KIRCH, O.; DAWSON, T. *Linux Network Administrator's Guide*. 2. ed. Sebastopol: O'Reilly, 2002. Disponível em: `http://www.tldp.org/guides.html`.

KITANO, H. Computational Systems Biology. *Nature*, v. 420, p. 206–210, 14 nov. 2002.

KLEINROCK, L. *Information Flow in Large Communication Nets*. MIT, RLE Quarterly Progress Report, jul. 1961.

KLEINSTEIN, S. H.; SEIDEN, P. E. Simulating the Immune System. *Computing in Science and Engineering*, IEEE, p. 69–77, ago. 2000.

KOHLER, B.; PUZONE, R.; SEIDEN, P. E.; CELADA, F. A Systematic Approach to Vaccine Complexity Using an Automaton Model of the Cellular and Humoral Immune System – I. Viral Characteristics and Polarized Responses. *Vaccine*, Elsevier, v. 19, p. 862–76, 2001.

KOPP, E.; MEDZHITOV, R. Recognition of Microbial Infection by Toll-Like Receptor. *Current Opinion in Immunology*, n. 15, p. 396–401, 2003.

LANZAVECCHIA, A. Antigen-specific interaction between T and B cells. *Nature*, v. 314, n. 6011, p. 11–17, 11 abr. 1985.

LANZAVECCHIA, A.; SALLUSTO, F. Dynamics of T Lymphocyte Responses: Intermediates, Effectors, and Memory Cells. *Science*, v. 290, n. 5489, p. 92–97, 2000.

LANZAVECCHIA, A.; SALLUSTO, F. Regulation of T Cell Immunity by Dendritic Cells. *Cell*, v. 106, n. 3, p. 263–6, out. 2001.

LANZAVECCHIA, A.; SALLUSTO, F. Progressive Differentiation and Selection of the Fittest in the Immune Response. *Nature Reviews Immunology*, v. 2, n. 12, p. 982–7, dez. 2002.

LANZAVECCHIA, A.; SALLUSTO, F. Understanding the Generation and Function of Memory T Cell Subsets. *Current Opinion in Immunology*, v. 17, n. 3, p. 326–32, 2005.

LAZEBNIK, Y. Can a Biologist Fix a Radio? – Or, What I Learned Studying Apoptosis. *Cancer Cell*, v. 2, p. 179–82, set. 2002.

LEINER, M. B.; CERF, G. V.; CLARK, D. D.; KAHN, E. R.; KLEINROCK, L.; LYNCH, C. D.; POSTEL, J.; ROBERTS, G. L.; WOLFF, S. *Histories of the Internet: A Brief History of the Internet, version 3.32*. Internet Society, 10 dez. 2003. WWW. Disponível em: `http://www.isoc.org/internet/history/brief.shtml`. Acesso em: 15/10/2008.

LEINER, M. B.; CERF, V. G.; CLARK, D. D.; KAHN, R. E.; KLEINROCK, L.; LYNC, D. C.; POSTEL, J.; ROBERTS, L. G.; WOLFF, S. S. The Past and Future History of the Internet. *Comunications of the ACM*, v. 40, n. 2, p. 102–108, fev. 1997.

LÉVY, P. *As tecnologias da inteligência*. 2. ed. São Paulo: Editora 34, 1995.

LI, Y.; HU, Z.; HE, X.; ZHANG, W. Optimal Design of Self-Tuning Fuzzy Controler Based on Immune Principle. In: *Proceeding of the 5th World Congress on Intelligent Control and Automation*. Hangzhou, P.R. China: IEEE, 2004. p. 2593–7.

LICKLIDER, J. C. R. On-Line Man-Computer Communication, Spring Joint Computer Conference. *National Press*, California, v. 21, n. 7, p. 113–128, 1962.

LIPPMANN, R.; HAINES, J. W.; FRIED, D. J.; KORBA, J.; DAS, K. The 1999 darpa off-line intrusion detection evaluation. *Comput. Netw.*, Elsevier North-Holland, Inc., New York, NY, USA, v. 34, n. 4, p. 579–595, 2000. ISSN 1389-1286.

LOUZON, Y.; SOLOMON, S.; ATLAN, H.; COHEN, I. R. Proliferation and Competition in Discrete Biological Systems. *Bulletin of Mathematical Biology*, v. 65, p. 375-96, 2003.

MAGRO, C.; VAZ, N. Falando do corpo que fala. In: DÓRIA, F. A.; KATZ, C. S. (Ed.). *Razão / Desrazão*. Petrópolis: Vozes, 1991. p. 121–31.

MANN, S.; MITCHELL, E. L. *Linux System Security: An Administrator's Guide to Open Source Security Tools*. New Jersey: Prentice-Hall, 2000.

MANZ, R. A.; HAUSER, A. E.; HIEPE, F.; RADBRUCH, A. Maintenance of serum antibody levels. *Annual Review of Immunology*, v. 23, p. 367–86, abr. 2005.

MANZ, R. A.; RADBRUCH, A. Plasma cells for a lifetime? *European Journal of Immunology*, v. 32, n. 4, p. 923–7, abr. 2002.

MATTHEWS, J. *Rede de Computadores: Protocolos de Internet em Ação*. Rio de Janeiro: LTC, 2006.

MATZINGER, P. Tolerance, Danger, And The Extended Family. *Annu. Rev. Immunology*, v. 12, p. 991–1045, abr. 1994.

MATZINGER, P. Essay 1: The Danger Model in Its Historical Context. *Scand. J. Immunol.*, v. 54, p. 4–9, ago. 2001.

MATZINGER, P. The Danger Model: A Renewed Sense of Self. *Science*, v. 296, n. 5566, p. 301–5, 12 abr. 2002.

MEDZHITOV, R. Toll-Like Receptors and Innate Immunity. *Nature Reviews*, v. 1, p. 135–145, nov. 2001.

MEDZHITOV, R.; JANEWAY JR., C. Innate Immune Recognition and Control of Adaptive Immune Responses. *Seminars in Immunology*, v. 10, p. 351–3, 1998.

MEDZHITOV, R.; JANEWAY JR., C. Innate Immune Recognition: Mechanisms and Pathways. *Immunological Reviews*, v. 173, p. 89–97, 2000.

MEDZHITOV, R.; JANEWAY JR., C. A. Decoding the Patterns of Self and Nonself by the Innate Immune System. *Science*, v. 296, n. 5566, p. 298–300, 12 abr. 2002.

MEDZHITOV, R.; PRESTON-HURLBURT, P.; JANEWAY, C. A. A human homologue of the Drosophila Toll protein signals activation of adaptive immunity. *Nature*, v. 388, p. 394–397, 24 jul. 1997.

MELO, S. *Exploração de Vulnerabilidades em Redes TCP/IP*. Rio de Janeiro: Alta Books, 2004.

MITCHELL, M. *An Introduction to Genetic Algorithms*. Massachusetts: MIT Press, 1996.

MITNICK, K. L.; SIMON, W. L. *A Arte de Invadir: As verdadeiras histórias por trás das ações de hackers, intrusos e criminosos eletrônicos*. São Paulo: Prentice-Hall, 2006.

MUSHEGIAN, A.; MEDZHITOV, R. Evolutionary Perspective on Innate Immune Recognition. *The Journal of Cell Biology*, v. 155, n. 5, p. 705–10, nov. 2001.

NAKAMURA, E. T.; GEUS, P. L.-c. d. *Segurança de redes*. São Paulo: Berkeley, 2002.

NASAROUI, O.; GONZALEZ, F.; DASGUPTA, D. The fuzzy artificial immune system: motivations, basic concepts,and application to clustering and Web profiling. In: *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, 2002 - FUZZ-IEEE'02.* Honolulu: IEEE, 2002. v. 1, p. 711–716.

NEMETH, E.; SNYDER, G.; HEIN, T. R. Linux Administration Handbook. Prentice-Hall, New Jersey, 2002.

NEMETH, E.; SNYDER, G.; SEEBASS, S.; HEIN, T. R. *UNIX System Administration Handbook*. 2. ed. New Jersey: Prentice-Hall, 1995.

NEMETH, E.; SNYDER, G.; SEEBASS, S.; HEIN, T. R. *UNIX System Administration Handbook*. 3. ed. New Jersey: Prentice-Hall, 2001.

NEY, C. Invasão!: Entendendo as técnicas de análise com o Nmap. *Linux Magazine*, n. 17, p. 62–71, fev. 2006.

NIC BR SECURITY OFFICE. *Práticas de Seguranca para Administradores de Redes Internet*. NIC BR Security Office, 16 maio 2003. Disponível em: `http://www.cert.br/docs/seg-adm-redes/seg-adm-redes.pdf`.

NORTHCUTT, S.; NOVAK, J.; MCLACHLAN. *Network Intrusion Detection: An Analyst's Handbook*. 2. ed. Indianapolis: New Riders, 2001.

NOVÁK, V. Towards formal theory of soft computin. *Soft Computing*, v. 2, n. 1, p. 4–6, 1998.

NUSSENZVEIG, H. M. (Ed.). *Complexidade e Caos*. Rio de Janeiro: Edirota UFRJ/Copea, 2003.

NUSSENZVEIG, H. M. Introdução à Complexidade. In: NUSSENZVEIG, H. M. (Ed.). *Complexidade e Caos*. Rio de Janeiro: Edirota UFRJ/Copea, 2003.

OCHSENBEIN, A. F.; PINSCHEWER, D. D.; SIERRO, S.; HORVATH, E.; HENGARTNER, H.; ZINKERNAGEL, R. M. Protective long-term antibody memory by antigen-driven and T help-dependent differentiation of long-lived memory B cells to short-lived plasma cells independent of secondary lymphoid organs. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, v. 97, n. 24, p. 13263–8, 21 nov. 2000.

ODA, T.; WHITE, T. Developing an Immunity to Spam. In: *Proceedings of Genetic and Evolutionary Computation - GECCO 2003 - Genetic and Evolutionary Computation Conference, Chicago, IL, USA, July 12-16, 2003*. Chicago: Springer, 2003. (LNCS, v. 2723), p. 231–242.

ODA, T.; WHITE, T. Increasing the Accuracy of a Spam-Detecting Artificial Immune System. In: *Proceedings of the Congress on Evolutionary Computation (CEC 2003), Canberra, Australia, December 2003*. Australia: IEEE, 2003. v. 1, p. 390–396.

OLIVEIRA, P. M. C. de. Autômatos Celulares. In: NUSSENZVEIG, H. M. (Ed.). *Complexidade e Caos*. Rio de Janeiro: Edirota UFRJ/Copea, 2003.

O'REILLY, T. *What Is Web 2.0 Design Patterns and Business Models for the Next Generation of Software*. 30 set. 2005. WWW. Disponível em: `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html`. Acesso em: 24/10/2008.

OTTINO, J. M. Engineering Complex Systems. *Nature*, v. 427, n. 29, p. 399, 29 jan. 2004.

PAPPALARDO, F.; LOLLINI, P.; CASTIGLIONE, F.; MOTTA, S. Modeling and Simulation of Cancer Immunoprevention Vaccine. *Bioinformatics*, Oxford University Press.

PASARE, C.; MEDZHITOV, R. Toll-like Receptors and Acquired Immunity. *Seminars in Immunology*, v. 16, p. 23–6, 2004.

PAWLAK, Z. Rough Sets. *International Journal of Computer and Information Sciences*, v. 11, n. 5, p. 341–356, 1982.

PAWLAK, Z. *Rough Sets: Theoretical Aspects of Reasoning About Data*. London: Kluwer, 1991.

PETER, I. *Internet History Resource Centre*. The Internet History Project, 2007. Internet History Project (WWW). Disponível em: `http://www.nethistory.info/`. Acesso em: 19/10/2008.

PETROVSKY, N.; BRUSIC, V. Computational immunology: The coming of age. *Immunol Cell Biol.*, v. 80, n. 3, p. 248–54, Jun 2002.

PETROVSKY, N.; SILVA, D.; BRUSIC, V. The future for computational modelling and prediction systems in clinical immunology. In: BOCK, G.; GOODE, J. (Ed.). *Imunoinformatics: Bioinformatic Strategies for Better Understanding Of Immune Function*. New Jersey: John Wiley & Sons, 2003, (Novartis Foundation Symposium, v. 254). p. 23–42.

POSTEL, J. *User Datagram Protocol*. Internet Engineering Task Force (IETF), ago. 1980. (Request for Comments: 768, STD0006). Disponível em: `http://www.ietf.org/rfc/rfc768.txt`.

POSTEL, J. *Internet Control Message Protocol*. Internet Engineering Task Force (IETF), set. 1981. (Request for Comments: 792, STD0005). Disponível em: `http://www.ietf.org/rfc/rfc792.txt`.

POSTEL, J. *Internet Protocol*. Internet Engineering Task Force (IETF), set. 1981. (Request for Comments: 791, STD0005). Disponível em: `http://www.ietf.org/rfc/rfc791.txt`.

POSTEL, J. *Transmission Control Protocol*. Internet Engineering Task Force (IETF), set. 1981. (Request for Comments: 793, STD0007). Disponível em: `http://www.ietf.org/rfc/rfc793.txt`.

POWERS, S. T.; HE, J. A hybrid artificial immune system and self organising map for network intrusion detection. *Information Sciences*, v. 178, n. 15, p. 3024 – 3042, 2008. ISSN 0020-0255. Nature Inspired Problem-Solving.

RAMAKRISHNAN, K.; FLOYD, S.; BLACK, D. *The Addition of Explicit Congestion Notification (ECN) to IP*. Internet Engineering Task Force (IETF), set. 2001. (Request for Comments: 3168). Disponível em: `http://www.ietf.org/rfc/rfc3168.txt`.

RATCLIFF, J. W.; METZENER, D. E. Pattern Matching: The Gestalt Approach. *Dr. Dobbs Journal*, p. 46–51, 1 jul. 1988.

RNP - REDE NACIONAL DE PESQUISA. *Guia do Usuário Internet/Brasil*. Rio de Janeiro, abr. 1996. Disponível em: `http://www.rnp.br/_arquivo/documentos/rpu0013d.pdf`.

RUSSO, M. Propriedades Gerais do Sistema Imune. In: CALICH, V.; VAZ, C. (Ed.). *Imunologia*. Rio de Janeiro: Revinter, 2001. cap. 1, p. 1–9.

SALLUSTO, F.; GEGINAT, J.; LANZAVECCHIA, A. Central Memory and Effector Memory T Cell Subsets: Function, Generation, and Maintenance. *Annu. Rev. Immunol.*, v. 22, p. 745–63, 2004.

SALLUSTO, F.; LANZAVECCHIA, A. Exploring Pathways for Memory T Cell Generation. *J. Clin. Invest.*, v. 108, n. 6, p. 805–6, set. 2001.

SANTOS, R. M. Z. dos; BERNARDES, A. T. Immunization and Aging: a Learning Process in the Immune Network. *Physical Review Letters*, v. 81, n. 14, p. 3034–7, 5 out. 1998.

SANTOS, R. M. Z. dos; COPELLI, M. Long Term and Short Term Effects of Perturbations in a Immune Network Model. *Brazilian Journal of Physics*, Sociedade Brasileira de Física, São Paulo, v. 33, n. 3, p. 628–33, 2003.

SCHAFF, A. *A sociedade da informação*. São Paulo: UNESP, 1992.

SCHNEIER, B. *Applied Cryptography*. New York: John Wisley, Inc., 1996.

SCHNEIER, B. *Segurança.com: Segredos e Mentiras sobre a Proteção na Vida Digital*. Rio de Janeiro: Campus, 2001.

SEGEL, L. A.; COHEN, I. R. (Ed.). *Design Principles for the Immune System and Other Distributed Autonomous Systems*. New York: Oxford University Press, 2001. (Santa Fe Institute Studies in the Sciences of Complexity).

SEIDEN, P. E.; CELADA, F. A Model for Simulating Cognate Recognition and Response in the Immune System. *J. Theoretical Biology*, v. 158, n. 3, p. 329–57, 1992.

SHAFER, G. *A mathemathical theory of evidence*. Princeton: Princeton University Press, 1976.

SOARES, L. F. G.; LEMOS, G.; COLCHER, S. *Redes de Computadores: das LANs, MANs e WANs às Redes ATM*. 2. ed. Rio de Janeiro: Campus, 1995.

SRIKUSALANUKUL, W.; BRUYNE, F. de; MCCULLAGH, P. Modelling of Peripheral Lymphocyte Migration: System Identification Approach. *Immunology and Cell Biology*, v. 78, p. 288–93, 2000.

STANFIELD, V.; SMITH, R. W. *Linux System Administration*. San Francisco: Sybex, 2001. (Craig Hunt Linux Library).

STEWART, J.; VARELA, F. J. Exploring the Meaning of Connectivity in the Immune Network. *Immunological Reviews*, n. 110, p. 37–61, 1989.

STIBOR, T.; MOHR, P.; TIMMIS, J.; ECKERT, C. Is negative selection appropriate for anomaly detection? In: *Proceedings of Genetic and Evolutionary Computation Conference (GECCO 2005)*. Washington DC: ACM Press, 2005. p. 321–328.

STIBOR, T.; TIMMIS, J.; ECKERT, C. A comparative study of real-valued negative selection to statistical anomaly detection techniques. In: JACOB, C.; PILAT, M. L.; BENTLEY, P. J.; TIMMIS, J. (Ed.). *ICARIS*. Canada: Springer, 2005. (LNCS, v. 3627), p. 262–275.

TARAKANOV, A.; DASGUPTA, D. An Immunochip Architecture and Its Emulation. In: *Proceeding of the 2002 NASA/DOD Conference on Evolvable Hardware (EH'02)*. [S.l.]: IEEE Computer Society, 2002. p. 261–5.

TARAKANOV, A. O.; SKORMIN, V. A.; SOKOLOVA, S. P. *Immunocomputing: Principles and Applications*. New York: Springer, 2003.

TARAKANOV, A. O.; TARAKANOV, Y. A. A Comparison of Immune and Genetic Algorithms for Two Real-Life Tasks of Pattern Recognition. *International Journal of Unconventional Computing*, v. 1, n. 4, p. 357–374, 2005.

TEDESCO, G.; TWYCROSS, J.; AICKELIN, U. . Integrating Innate and Adaptive Immunity for Intrusion Detection. In: *Proceedings of the 5th International Conference on Artificial Immune Systems (ICARIS 2006), Oeiras, Portugal*. Oeiras: Springer, 2006. (LNCS, v. 4163), p. 193–202.

TIMMIS, J.; ANDREWS, P.; OWENS, N.; CLARK, E. An interdisciplinary perspective on artificial immune systems. *Evolutionary Intelligence*, v. 1, n. 1, p. 5–26, mar 2008.

TIMMIS, J.; HONE, A.; STIBOR, T.; CLARK, E. Theoretical advances in artificial immune systems. *Theor. Comput. Sci.*, Elsevier Science Publishers Ltd., Essex, UK, v. 403, n. 1, p. 11–32, 2008. ISSN 0304-3975.

TIMÓTEO, G. T. S. *Desenvolvimento de Algoritmos Genéticos para a Resolução do Timetabling*. Monografia (Monografia de Graduação) — DCC/UFLA, Lavras, 2002.

TRAGGIAI, E.; PUZONE, R.; LANZAVECCHIA, A. Antigen Dependent and Independent Mechanisms that Sustain Serum Antibody Levels. *Vaccine*, v. 21 Suppl 2, p. S2/35–S2/37, 2003.

TWYCROSS, J.; AICKELIN, U. Towards a Conceptual Framework for Innate Immunity. In: *Proceedings of the 4th International Conference on Artificial Immune Systems (ICARIS 2005)*. Banff, Canada: Springer-Verlag, 2005. (LNCS 3627), p. 112–25.

UCHÔA, J. Q. *Representação e Indução de Conhecimento Usando Teoria de Conjuntos Aproximados*. Dissertação (Mestrado) — Universidade Federal de São Carlos, São Carlos, jun. 1998.

UCHÔA, J. Q. *Configuração Segura do Serviço de E-Maill via Postfix - Palestra no FISL 6.0*. 2 jun. 2005. Palestra em Vídeo. Disponível em: `http://mirrors.softwarelivre.org/fisl6/fisl6_-_redes_-_joaquim_quinteiro_uchoa_-_configuracao_segura_do_servico_de_e_-_mail_via_postfix.ogg`. Acesso em: 02/11/2006.

UCHÔA, J. Q.; MOTA-SANTOS, T. A.; CAMINHAS, W. M. *An Artificial Innate and Adaptive Immune System Applied to the Problem of Intrusion Detection*. 2009. Submetido para publicação na IEEE Transactions On Systems, Man and Cybernetics — Part B: Cybernetics.

VARELA, F. J.; COUTINHO, A.; DUPIRE, B.; VAZ, N. N. Cognitive Networks: Immune, Neural, and Otherwise. In: PERELSON, A. S. (Ed.). *Theoretical Immunology, Part Two, Studies in the Sciences of Complexity*. Boston: Addison-Wesley, 1988. p. 359–375.

VAZ, N. M.; FARIA, A. M. *Guia Incompleto de Imunobiologia*. Belo Horizonte: Coopmed, 1993.

VAZ, N. M.; FARIA, A. M. de; VERDOLIN, B. A.; NETO, A. F. S.; MENEZES, J. S.; CARVALHO, C. R. The Conservative Physiology of the Immune System. *Brazilian Journal of Medical Biological Research*, v. 36, n. 1, p. 13–22, 2003.

WANG, P.; TAN, S. Soft computing and fuzzy logic. *Soft Computing*, v. 1, n. 1, p. 35–41, 1997.

WARRENDER, C.; FORREST, S.; SEGEL, L. Effective Feedback in the Immune System. In: SMITH, R. E.; BONACINA, C.; HOILE, C.; MARROW, P. (Ed.). *Genetic and Evolutionary Computation Conference Workshop Program (GECCO'2001)*. San Francisco, California, USA: Morgan Kaufman, 2001. p. 329 – 332.

WARRENDER, C. E. *CyCells User Manual*. Albuquerque, 17 set. 2004. Disponível em: `http://cs.unm.edu/~christy/simcode/cycells.html`.

WARRENDER, C. E. *Modeling Intercellular Interaction in The Peripheral Immune System*. Tese (Doutorado) — University of New Mexico, Albuquerque/New Mexico, dez. 2004.

WEBER, R. F. Segurança na Internet. In: *Anais da XIX JAI - Jornada de Atualização em Informática*. Curitiba: PUCPR, 2000. p. 43–82.

WILSON, W. O.; BIRKIN, P.; AICKELIN, U. Motif detection inspired by immune memory. In: DE CASTRO, L. N.; VON ZUBEN, F. J.; KNIDEL, H. (Ed.). *Artificial Immune Systems, 6th International Conference, ICARIS*. Santos, 2007. (Lecture Notes in Computer Science, v. 4628), p. 276–287.

WIRZENIUS, L.; OJA, J.; STAFFORD, S.; WEEKS, A. *The Linux System Administrator's Guide*. Version 0.9. Chapel Hill (North Carolina): The Linux Documentation Project, 2002. Disponível em: `http://www.tldp.org/guides.html`.

WU, S.-Y.; YEN, E. Data mining-based intrusion detectors. *Expert Syst. Appl.*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 36, n. 3, p. 5605–5612, 2009. ISSN 0957-4174.

YAKUWA, F.; SATOH, S.; SHAIKH, M. S.; DOTE, Y. Fault Diagnosis for Dynamical Systems Using Soft Computing. In: *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, 2002 - FUZZ-IEEE'02*. Honolulu, Hawaii (USA): IEEE, 2002. p. 261–6.

YEPES, I.; BARONE, D. A. C. Inteligência Artificial Distribuída: Uma Abordagem ao Comportamento Social Inteligente. In: BARONE, D. (Ed.). *Sociedades Artificiais*. Porto Alegre: Bookman, 2003. cap. 9, p. 231–50.

YU, Z.; TSAI, J. J. P.; WEIGERT, T. J. An automatically tuning intrusion detection system. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, v. 37, n. 2, p. 373–384, 2007.

ZADEH, L. A. Fuzzy Sets. *Information and Control*, v. 8, n. 3, p. 338–353, 1965.

ZADEH, L. A. A theory of approximate reasoning. In: HAYES, J. E.; MICHIE, D.; MIKULICH, L. I. (Ed.). *Machine Inteligence*. New York: Halstead Press, 1979. v. 9, p. 149–194.

ZADEH, L. A. Fuzzy Logic, Neural Networks, and Soft Computing. *Communications of the ACM*, v. 37, n. 3, p. 77–84, mar. 1994.

ZEBULUM, R. S.; PACHECO, M. A. C.; VELLASCO, M. M. B. R. *Evolutionary Electronics: Automatic Design of Eletronic Circuits and Systems by Genetic Algorithms*. New York: CRC Press, 2002.

ZINKERNAGEL, R. M. On differences between immunity and immunological memory. *Current Opinion in Immunology*, v. 14, n. 4, p. 523–36, 1 ago. 2002.

ZUO, X.-Q.; LI, S.-Y.; BAN, X.-J. An Immunity-based optimization algorithm for tuning neuro-fuzzy controller. In: *Proceedings of the Second International Conference on Machine Learning and Cybernetics*. Xi'an: IEEE, 2003. p. 666–71.

APÊNDICE

A

Exemplos de Código

A.1 *Script* para captura de dados de pacotes

```
#! /usr/bin/env python
# encoding: utf-8

from scapy.all import *

PACKAGES_QTD = 500
RAW_BEGIN_END = 5

while True:
    pks = sniff(filter="tcp or udp or icmp", count=PACKAGES_QTD)
    tosave = ""
    tempo = ""
    for i in xrange(len(pks)):

        comum = ""
        tempo = str(int(pks[i].time)) + ":"
        comum += str(pks[i].payload.ihl) + ","
        comum += str(pks[i].payload.len) + ","
        comum += str(pks[i].payload.id) + ","
        comum += str(pks[i].payload.flags) + ","
        comum += str(pks[i].payload.frag) + ","
        comum += str(pks[i].payload.ttl) + ","
```

```
        if pks[i].payload.proto == 6:
            tosave += tempo + "tcp," + comum
            tosave += str(pks[i].payload.payload.sport) + ","
            tosave += str(pks[i].payload.payload.dport) + ","
            tosave += str(pks[i].payload.payload.flags) + ","
            tosave += str(pks[i].payload.payload.window) + ","
            if pks[i].payload.payload.urgptr == 0:
                urgp = 0
            else:
                urgp = 1
            tosave += str(urgp) + ","

        elif pks[i].payload.proto == 17:
            tosave += tempo + "udp," + comum
            tosave += str(pks[i].payload.payload.sport) + ","
            tosave += str(pks[i].payload.payload.dport) + ","
            tosave += str(pks[i].payload.payload.len) + ","

        else:
            tosave += tempo + "icmp," + comum
            tosave += str(pks[i].payload.payload.type) + ","
            tosave += str(pks[i].payload.payload.code) + ","
            tosave += str(pks[i].payload.payload.id) + ","
            tosave += str(pks[i].payload.payload.seq) + ","

        try:
            loadraw = pks[i].payload.payload.payload.load
            loadraw = loadraw[0:RAW_BEGIN_END]
               + loadraw[len(loadraw)-RAW_BEGIN_END:]
        except:
            loadraw = "NONE"
        tosave += loadraw + "\n"

    print tosave
```

A.2 *Script* para captura de sinais

```
#! /usr/bin/env python
import time

SLEEPTIME = 1 # in secs

def read_snmp_data( ):
    vec_var = []
    tempo = ""
    sm = open('/proc/net/snmp','r')
```

```
    tempo = str(int(time.time())) + ":"
    l = sm.readlines()
    ip = l[1].split(" ")
    icmp = l[3].split(" ")
    tcp = l[7].split(" ")
    udp = l[9].split(" ")
    sm.close()
    vec_var.append(ip[3]) # InReceives
    vec_var.append(ip[4]) # InHdrErrors
    vec_var.append(ip[5]) # InAddrErrors
    vec_var.append(ip[8]) # InDiscards
    vec_var.append(ip[11]) # OutDiscards
    vec_var.append(ip[12]) # OutNoRoutes
    vec_var.append(icmp[1]) # InMsgs
    vec_var.append(icmp[2]) # InErrors
    vec_var.append(icmp[8]) # InEchos
    vec_var.append(icmp[15]) # OutErrors
    vec_var.append(icmp[16]) # OutDestUnreachs
    vec_var.append(tcp[7])  #AttemptFails
    vec_var.append(tcp[8]) # EstabResets
    vec_var.append(tcp[9]) # CurrEstab
    vec_var.append(tcp[10]) # InSegs
    vec_var.append(tcp[11]) # OutSegs
    vec_var.append(tcp[12])# RetransSegs
    vec_var.append(tcp[13]) # InErrs
    vec_var.append(tcp[14].replace('\n',"")) # OutRsts
    vec_var.append(udp[1]) # InDatagrams
    vec_var.append(udp[2]) # NoPorts
    vec_var.append(udp[3]) # InErrors
    vec_var.append(udp[4].replace('\n',"")) # OutDatagrams

    return vec_var,tempo

old_data,x = read_snmp_data()
y = 0
while True:
    x = time.time()
    time.sleep(SLEEPTIME - y)
    new_data,tempo = read_snmp_data()
    delta = ""
    delta += tempo
    for i in xrange(len(new_data)):
        delta += str(int(new_data[i]) - int(old_data[i])) + ","
    old_data = new_data
    print delta[:-1]
    y = time.time() - x - SLEEPTIME
```