Universidade Estadual de Londrina

RAFAEL SEIDI OYAMADA

DETECÇÃO DE ANOMALIAS EM REDES SDN UTILIZANDO ALGORITMO BASEADO EM SISTEMAS IMUNOLÓGICOS ARTIFICIAIS

LONDRINA
2018

RAFAEL SEIDI OYAMADA

DETECÇÃO DE ANOMALIAS EM REDES SDN UTILIZANDO ALGORITMO BASEADO EM SISTEMAS IMUNOLÓGICOS ARTIFICIAIS

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Mario Lemes Proença Jr.

Coorientador: Prof. Dr. Luiz Fernando Carvalho

LONDRINA

2018

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema de Bibliotecas da UEL

Oyamada, Rafael.
Detecção de anomalias em redes SDN utilizando algoritmo baseado em Sistemas Imunológicos Artificiais / Rafael Oyamada. - Londrina, 2018.
61 f. : il.

Orientador: Prof. Dr. Mario Lemes Proença Jr..
Coorientador: Prof. Dr. Luiz Fernando Carvalho.
Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) - Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2018.
Inclui bibliografia.

1. Sistema Imunológico Artificial - TCC. 2. Redes Definidas por Software - TCC. 3. Sistema de Detecção de Intrusão - TCC. I. Proença Jr., Prof. Dr. Mario Lemes . II. Carvalho, Prof. Dr. Luiz Fernando. III. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

RAFAEL SEIDI OYAMADA

DETECÇÃO DE ANOMALIAS EM REDES SDN UTILIZANDO ALGORITMO BASEADO EM SISTEMAS IMUNOLÓGICOS ARTIFICIAIS

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof. Dr. Mario Lemes Proença Jr.
Universidade Estadual de Londrina

Prof. Dr. Luiz Fernando Carvalho
Universidade Estadual de Londrina

Prof. Dr. Elieser Botelho Manhas Jr.
Universidade Estadual de Londrina

Londrina, 24 de novembro de 2018.

AGRADECIMENTOS

Primeiramente agradeço à minha família, em especial minha avó que sempre cuidou de mim e me incentivou a seguir os meus sonhos.

Meus agradecimentos vão também ao meu orientador Prof. Dr. Mário Lemes Proença e ao meu coorientador Prof. Dr. Luiz Fernando Carvalho que me auxiliaram ao longo do ano e me acompanharam nessa fase de conclusão de curso. Aos mestrandos Cinara Brenda Zerbini e Matheus Pereira de Novaes que me ajudaram o ano inteiro compartilhando seus conhecimentos e experiências. À Universidade Estadual de Londrina, principalmente, por ter proporcionado todo o aprendizado durante esta caminhada.

Agradeço imensamente aos amigos que fiz ao longo da graduação, em especial aos da Turma 23 de Ciência da Computação. Às amizades proporcionadas pela Associação Atlética Acadêmica Exatas UEL, instituição da qual fiz parte da diretoria desde o seu ano de fundação em 2013 e que me proporcionou momentos gratificantes que jamais esquecerei, e pela Lolloteria, grupo de ritmistas que também fiz parte desde sua fundação e que me concedeu sentimentos únicos e inexplicáveis. Por fim, deixo os agradecimentos também aos amigos do H14 que sempre me ofereceram bons momentos.

"Tempos difíceis nos aguardam e em breve teremos que escolher entre o que é certo e o que é fácil."
(Alvo Dumbledore)

"Everybody wants to go to heaven, but nobody wants to die."
(Albert King)

"As pessoas não são más. Elas só estão perdidas. Ainda há tempo."
(Criolo)

OYAMADA, RAFAEL SEIDI. **Detecção de anomalias em redes SDN utilizando algoritmo baseado em Sistemas Imunológicos Artificiais**. 2018. 60f. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2018.

RESUMO

Devida à rápida expansão dos dispositivos interconectados, tem se tornado necessário aderir a novos paradigmas de gerenciamento de redes. Uma Rede Definida por Software (SDN) é considerada uma nova solução para este cenário. SDN trata-se de uma nova arquitetura que oferece robustez e praticidade ao gerente de uma rede. Porém, a nova proposta ainda não garante integridade para o ambiente. Para isso, é preciso investir em ferramentas e políticas de segurança, como por exemplo, Sistemas de Detecção de Intrusão. Portanto, o presente trabalho tem como objetivo combinar heurísticas de aprendizado de máquina para o desenvolvimento de um sistema de detecção de intrusões.

Palavras-chave: Sistema de Detecção de Intrusão, Redes Definidas por Software, Sistema Imunológico Artificial

OYAMADA, RAFAEL SEIDI. **Detecção de anomalias em redes SDN utilizando algoritmo baseado em Sistemas Imunológicos Artificiais**. 2018. 60p. Final Project (Bachelor of Science in Computer Science) – State University of Londrina, Londrina, 2018.

ABSTRACT

Due to the rapid expansion of interconnected devices, it becomes necessary to adhere to new paradigms of network management. A Software Defined Network (SDN) is a solution for this scenario. It is a new architecture that offers robustness and practicality to the manager of a network. However, the new proposal does not guarantee the protection of the environment. In order to achieve this protection, it is necessary to invest in security tools and policies, such as Intrusion Detection Systems. Therefore, the present work aims to merge machine learning techniques to develop an intrusion detection system.

Keywords: Intrusion Detection System, Software Defined Network, Artificial Immune System

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AG	*Antigen Agent*
AIS	*Artificial Immune System*
API	*Application Programming Interface*
ARIMA	*Autoregressive Integrated Moving Average*
BLGBA	*Baseline for Automatic Backbone Management*
CPM	*Correlation Paraconsistent Machine*
DC	*Dendritic Cell*
DDoS	*Distributed Denial of Service*
DoS	*Denial of Service*
DSNS	*Digital Signature of Network Segment*
DSNSF	*Digital Signature of Network Segment Using Flow Analysis*
DT	*Danger Theory*
FA	*Firefly Algorithm*
GA	*Genetic Algorithm*
IDS	*Intrusion Detection System*
IP	*Internet Protocol*
KHM	*K-Harmonic Means*
NSA	*Negative Selection Algorithm*
ONF	*Open Networking Foundation*
RF	*Random Forest*
RP	*Responding Agent*
SA	*Simulated Annealing*
SDN	*Software Defined Network*
SIA	Sistema Imunológico Artificial
SVM	*Support Vector Machine*

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos houve um crescimento intenso do uso da tecnologia em diversos seguimentos de serviços e atividades rotineiras. Estamos vivendo em uma era onde a dependência de computadores e dispositivos móveis vem se tornando cada vez mais frequente. Em razão disto, o acesso à Internet e a geração de tráfego também cresce exponencialmente [6]. Conforto e praticidade podem ser considerados os pontos positivos desta nova era, por exemplo, já não há necessidade de sair de casa para realizar simples tarefas, tais como efetuar compras e transações bancárias.

Tendo ciência desse enorme crescimento de aparelhos interconectados, é preciso aderir a novos paradigmas de rede que atendam a essa demanda fornecendo mais praticidade e simplicidade quando se tratar de gerenciamento. Desta maneira, Rede Definida por Software (*Software Defined Network, SDN*) emerge como principal solução proposta [7], [8] na área atualmente. Trata-se de uma tecnologia que separa os mecanismos de encaminhamento e de controle, visando uma maior flexibilidade e robustez. Entretanto, apesar do paradigma proposto apresentar diversos benefícios, ainda continua vulnerável a ataques, o que torna imprescindível o constante investimento em segurança.

Com essa dependência excessiva de redes, é iminente o surgimento de atividades maliciosas vindas de usuários de má-fé, que estão sempre buscando roubar dados pessoais e informações sigilosas que possam beneficiá-los de alguma maneira, ou até mesmo causar danos severos a um determinado sistema. Em virtude deste avanço acelerado da tecnologia e o crescimento da Internet das Coisas [9], um ataque bastante popular que vem acompanhando esse cenário é o DDoS (*Distributed Denial of Service*) [10]. Atualmente existe uma imensa base de ferramentas automatizadas [11], que podem ser facilmente encontradas na Internet, cujo objetivo é realizar ataques como este ou explorar vulnerabilidades. Assim sendo, proteger uma grande infraestrutura de redes tem se tornado uma tarefa interminável, já que novas fraquezas são encontradas diariamente em sistemas computacionais.

Uma proposta estudada na área de Segurança da Informação são os Sistemas de Detecção de Intrusão (*Intrusion Detection System, IDS*) [9], [12], [13]. Vale ressaltar que um IDS não substitui demais *softwares* e políticas de segurança, como por exemplo *firewalls*, anti-vírus e credenciais, mas sim, complementa todo este conjunto. O método abordado pelo IDS é baseado no fundamento de que qualquer tipo de invasão pode ser detectado pelo desvio do comportamento esperado, onde este comportamento é dado como como uma base de dados contendo atributos de tráfego, coletados em tempo real ou gerados artificialmente, classificados como normais. A fim de se evitar falsos negativos e diminuir taxas de erro [14], [15], o desenvolvimento desta ferramenta pode ser realizado

utilizando diversas heurísticas de aprendizado de máquina, e a combinação delas pode gerar resultados ainda mais precisos [16], [17], [18].

O ramo de pesquisa na área de segurança vem buscando constantemente apresentar novas técnicas e métodos para desenvolver os melhores IDS para seus devidos cenários. Este trabalho, portanto, irá propor o desenvolvimento de um IDS utilizando o algoritmo de Seleção Negativa, que é baseado em Sistemas Imunológicos Artificiais, e será aplicado em tráfegos de redes SDN emulados por ferramentas propícias para pesquisas na área. O modelo proposto é uma adaptação do Sistema Imunológico Biológico formado por meios computacionais, o qual tem como objetivo resolver problemas do mundo real.

O trabalho está organizado da seguinte forma: a Seção 2 aborda um paradigma emergente de Gerência de Redes, as redes SDN. A seção 3 descreve a ferramenta a ser desenvolvida no trabalho. A Seção 4 descreve a fundamentação teórica da meta-heurística utilizada no desenvolvimento. A Seção 5 apresenta trabalhos correlatos nas áreas de computação evolutiva e redes. A Seção 6 expõe a proposta de implementação, os métodos utilizados e avaliação do modelo. E por fim, a Seção 7 contém a conclusão do trabalho

2 REDES DEFINIDAS POR SOFTWARE

As Redes Definidas por Software têm sido a principal proposta abordada nos últimos tempos para atender a explosão de crescimento de dispositivos interconectados [8], [19]. Para a ONF (*Open Networking Foundation*), uma organização sem fins lucrativos, que tem como missão transformar a área de redes em plataformas ágeis para prestações de serviços, uma rede SDN pode ser definida como uma arquitetura de rede emergente no mercado, onde o controle da rede é desacoplado do mecanismo de encaminhamento e diretamente programável.

Considerado como o sistema operacional de uma rede, SDN é um paradigma que visa separar as funções de encaminhamento de dados na rede, executadas pelo plano de dados, das funções de controle de rede, executadas pelo plano de controle [20]. Em redes tradicionais, quem decide o caminho que os pacotes devem seguir aplicando algoritmos de roteamento é o roteador. Para um operador, essa centralização de controle e simplificação no gerenciamento são propriedades vantajosas e que garantem uma maior robustez da rede.

O desacoplamento das funções citadas pode ser organizado em camadas e alguns trabalhos recentes [21], [22] focam suas pesquisas apenas em alguma específica, porém no trabalho [23] os autores afirmam que o entendimento profundo de cada parte da arquitetura e um investimento balanceado na pesquisa de cada uma delas é importante para maximizar o potencial de uma rede SDN.

2.1 Arquitetura

A arquitetura de uma rede SDN, ilustrada na Figura 1, pode ser dividida em três camadas: aplicação, plano de controle e plano de dados, onde cada uma delas possui componentes próprios. A camada de aplicação explora ambos os planos para atingir objetivos específicos, trata-se das possíveis aplicações de uma rede SDN. As aplicações se comunicam com o controlador através do lado norte da interface, que é localizado no plano de controle, e o plano de controle comunica-se com o plano de dados pelo lado sul, que é localizado nos *switches*, roteadores e pontos de acesso.

A separação do plano de controle e do plano de dados é quem faz uma SDN ser mais eficiente e prática, pois assim o gerente de redes consegue realizar ajustes, modificações e inserções de novos recursos no plano de controle sem alterar o plano de dados.

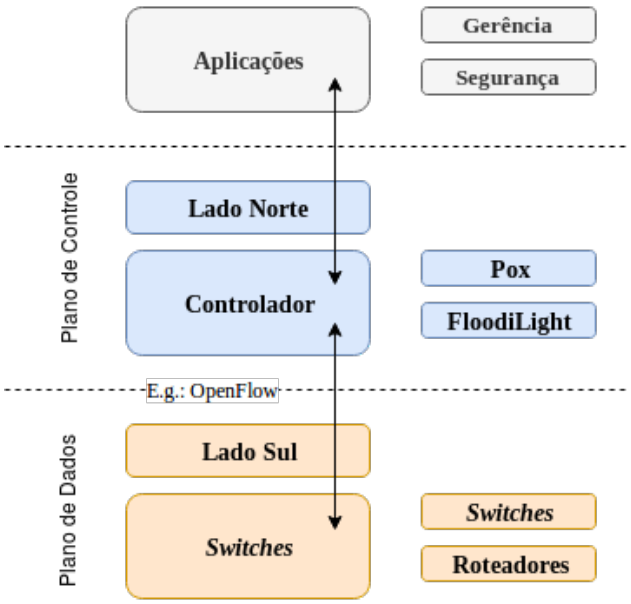

Figura 1 – Arquitetura geral de uma rede SDN. Adaptado de [1].

2.1.1 Plano de Dados

O funcionamento do plano de dados acontece por meio da comunicação entre o *switch* e o plano de controle, que ocorre no lado sul da interface, onde tal conexão é estabelecida através de protocolos de compartilhamento [24].

Um exemplo de protocolo que deve ser citado é o OpenFlow [25]. Este foi criado como um padrão aberto, fazendo com que os fabricantes de equipamentos de redes possam habilitar seus produtos a ele. Com tal tecnologia é possível modificar um comportamento de uma rede inteira de uma maneira barata, dinâmica e independente dos fabricantes. É ele quem define uma Interface de Programação de Aplicações (*Application Programming Interface*, API) para a comunicação entre as partes. Outros exemplos de ferramentas capazes de atingir resultados semelhantes são o JunOS SDK e Cisco One.

A infraestrutura de uma rede SDN é composta por dispositivos da rede como roteadores, *switches* e pontos de acesso, e o protocolo OpenFlow faz com que eles sejam programáveis pelo controlador. Em geral, o plano de dados é responsável pelo encaminhamento de pacotes para portas específicas do equipamento, e as operações realizadas sobre eles, como modificação de cabeçalho e descarte, são feitas pelo controlador a partir de mensagens recebidas do OpenFlow.

Alguns tipos de serviços como armazenamento em cache e transcodificação na rede também são suportados pelo plano de dados.

2.1.2 Plano de Controle

Em uma rede SDN o plano de controle, onde fica situado o controlador, age como uma camada intermediária entre as aplicações e o plano de dados, e pode atuar como *firewall* ou executando operações de roteamento e priorização de pacotes. Seu isolamento cria uma entidade logicamente centralizada, tornando o gerenciamento da rede mais simples e fazendo com que ela seja menos propensa a erros, já que é capaz de supervisionar o comportamento de encaminhamento de dados. Uma rede pode ter mais de um controlador, onde eles podem atuar em paralelo ou ter um principal e os demais destinados a *backup*.

Uma tabela de fluxo tem como objetivo determinar como os pacotes de dados serão tratados. Cada entrada da tabela contém um conjunto de campos do pacote (endereço IP e portas de origem/destino por exemplo) e um conjunto de ações a serem aplicadas sobre o pacote. Tais ações são realizadas pelo plano de controle, que controla a tabela de fluxos do *switch*, podendo adicionar, atualizar e remover entradas de fluxos. Esse processo pode reativo ou proativo dependendo da implementação do protocolo de compartilhamento.

Quando um pacote recebido pelo *switch* gera algum erro no encaminhamento, ele pode ser encaminhado para o controlador, passado para a próxima tabela de fluxo ou ser descartado. Quando o controlador é seu novo destino, além de poder ser descartado ou modificado, o pacote também pode receber uma nova entrada de fluxo permitindo que o *switch* saiba como agir em situações similares futuramente.

Atualmente existem diversas implementações de controladores. Seguem alguns exemplos e as visões gerais sobre os mesmos:

- NOX [26]: é o primeiro controlador OpenFlow. Começou sendo escrito em C++ e Python, mas atualmente é usado apenas em aplicações de controle implementadas em C++;

- POX [27]: é um controlador SDN *open-source* que suporta os protocolos OpenFlow e OVSDB. Implementado em Python, é bastante popular nas áreas de ensino e pesquisa;

- SNAC [28]: é um controlador OpenFlow baseado na versão 0.4 do NOX. Fornece uma interface gráfica amigável que permite o usuário gerenciar a rede e configurar seus dispositivos de uma maneira bastante prática;

- Floodlight [29]: oferece suporte a uma ampla variedade de *switches* OpenFlow virtuais e físicos e pode lidar com redes mistas OpenFlow e não OpenFlow. Implementado em Java;

- OpenMUL [30]: é um controlador OpenFlow que suporta uma infraestrutura *Multithreading* e vários níveis do lado norte da interface.

Existem também ferramentas e ambientes de desenvolvimento que simulam comportamentos de redes, no intuito de incentivar pesquisas permitindo protótipos de redes SDN. Abaixo alguns exemplos:

- Mininet [31]: é um emulador de rede que virtualiza hospedeiros, *switches*, controladores e *links*. Tais componentes agem como os de uma rede real e seus *switches* suportam o OpenFlow.
- NS-3 [32]: é um *software* livre que simula um ambiente de desenvolvimento voltado para pesquisas na área de redes. Suporta o protocolo OpenFlow.
- EstiNet [33]: este simulador inclui camada física, camada de controle de acesso à mídia, camada de rede, camada de transporte e camada de aplicação. Possui uma interface amigável para observação de depuração de resultados. Suporta o protocolo OpenFlow.

2.1.3 Camada de Aplicação

Uma rede SDN possui diferentes técnicas que podem ser abordadas em diferentes áreas, e é a camada de aplicação quem explora a combinação delas para alcançar diversos objetivos.

Coleta de informações sobre o comportamento da rede permite ao usuário saber como ela está se comportando e como pode ser melhorada, e é esse o diferencial de um sistema de gerência faz com que ele seja mais eficiente e dinâmico. Agarwal et al. [34] utilizam informações da rede coletadas por um controlador SDN centralizado para buscar melhorias reduzindo os atrasos e perdas de pacotes. Eles mostram também que o aperfeiçoamento da rede pode ser alcançado com uma implantação incremental de uma SDN em uma rede já existente. No trabalho [35] os autores propõem um *framework* para gerenciar e controlar a rede, o Procera. Essa ferramenta tem como objetivo simplificar e melhorar diversos aspectos, como as operações sobre o encaminhamento de pacotes e o gerenciamento dos mesmos. Ela também permite alterações sobre o estado da rede a qualquer momento, fornecendo um suporte para a sua configuração através de uma linguagem de alto nível.

Uma outra vantagem de ter um controle centralizado em redes SDN é a capacidade de supervisionar e monitorar comportamentos de usuários, o que torna possível a detecção de ataques podendo ter uma rápida resposta para assim evitar maiores detrimentos. O trabalho [36] mostra como o OpenFlow e o NOX garantem uma alta acurácia na detecção

de anomalias em redes domésticas. Eles afirmam que a flexibilidade de redes SDN garantem que suas aplicações mantenham-se atualizadas conforme vão surgindo novas ameaças de segurança.

2.2 Considerações sobre o capítulo

O paradigma de redes SDN está se tornando cada vez mais popular devido às suas características de desacoplamento de funções que permitem ao usuário inovar na hora estruturar, configurar e gerenciar uma rede. Isso graças aos protocolos de comunicação, como, por exemplo, o OpenFlow (o mais promissor em trabalhos e pesquisas recentes [37]), que realizam a conexão entre os planos de controle e de dados.

A tecnologia além de apresentar melhorias no desempenho da rede, também permite, graças a sua estrutura centralizada, maior flexibilidade no seu gerenciamento, que pode servir de incentivo no desenvolvimento de sistemas de detecção de intrusões, que serão abordados no próximo capítulo.

3 SISTEMA DE DETECÇÃO DE INTRUSÃO

Junto do aumento da dependência da *internet* também tem-se o crescimento de atividades maliciosas na rede. Algumas técnicas de segurança tradicionais, tais como *firewalls*, criptografia e autenticação, vêm atuando em diversos sistemas nos últimos anos devido a esse grande avanço no desenvolvimento de ataques. Visto isso, torna-se cada vez mais imprescindível investir constantemente em estratégias de segurança para garantir a integridade dos dados.

Este capítulo descreverá os ataques mais conhecidos atualmente e apresentará a ferramenta proposta como solução, começando com uma visão geral a seu respeito e finalizando com a explicação de como ela pode ser implementada.

3.1 Tipos de ataques

Um ataque é uma ação que tem como objetivo infectar, danificar ou roubar informações de um determinado sistema e pode ser provado por programas maliciosos (*malware*). Geralmente ataques causam alterações significativas no tráfego de rede, alguns deles são bastante conhecidos e serão descritos a seguir:

- Ataque de negação de serviço (*denial of service*, DoS): esse tipo de ataque tem como objetivo gerar uma sobrecarga para inutilizar um serviço. Pode ser causado através do envio de diversas requisições simultâneas para o alvo. Um ataque DoS também pode ser distribuído (DDoS), onde uma máquina toma posse de inúmeras outras máquinas e as utiliza para realizar um DoS a partir de cada uma delas;

- *Flash Crowd*: semelhante ao ataque de negação de serviço, este interrompe o funcionamento de um serviço quando o mesmo recebe inúmeros acessos simultâneos. O que os difere é que as requisições são realizadas por usuários não maliciosos;

- Escaneamento de portas (*port scan*): essa técnica consiste em varrer todas as portas de um sistema trazendo informações sobre elas e os serviços, caso existam, operando nas mesmas. Geralmente administradores de redes fazem uso desta para tarefas de gerência, porém ela pode muito bem ser aplicada por agentes maliciosos no intuito de encontrar vulnerabilidades no sistema.

3.2 Visão geral

Monitorar o comportamento de uma rede é crucial para garantir o seu bom funcionamento. Tal tarefa faz parte do conjunto de políticas e práticas de Segurança da

Informação, e é através dela que torna-se possível discriminar um desvio de comportamento no tráfego da rede. Essa discriminação pode ser realizada por um IDS, que além de detectar a anomalia é capaz de tratar e classificar o problema. Vale ressaltar que um desvio de comportamento na rede é considerado uma anomalia mas não é necessariamente uma intrusão, e cabe ao IDS diferenciar tais conceitos.

Um IDS pode ser divido em algumas categorias [38]. Um *Host-Based Intrusion Detection System* realiza observações nas atividades internas de um computador, como, por exemplo, chamadas de sistema e modificações de arquivos. De forma semelhante, porém realizando um monitoramento nas atividade de protocolos e serviços em uma rede inteira, buscando qualquer comportamento suspeito, o sistema que atua dessa forma é classificado como *Network-Based Intrusion Prevention System*. Um *Network Behavior Analysis* consiste em encontrar padrões nos tráfegos anômalos de uma rede analisando diversas características, como volume de tráfego e largura de banda.

O processo de detecção dessas ferramentas podem ser classificados em duas categorias:

- Detecção baseada em assinaturas: esse método consiste em comparar uma base de dados contendo ataques conhecidos com os dados coletados do tráfego da rede e, caso haja alguma semelhança entre as amostras, uma ameaça é detectada. Existem vantagens e desvantagens quando utiliza-se essa abordagem. A taxa de alarmes falsos e a de acertos tendem a ser baixa e alta, respectivamente. Por outro lado, o sistema falhará naturalmente quando se deparar com um ataque desconhecido. Este problema pode ser solucionado atualizando frequentemente a base de dados de assinaturas, contudo, essa é uma tarefa bastante custosa já que as bases devem ser sempre alimentadas devido ao constante desenvolvimento de novos ataques.

- Detecção baseada em anomalias: este busca encontrar qualquer desvio de comportamento no tráfego da rede. Existem alguns passos para se construir este modelo. Primeiro, o sistema precisa realizar a coleta de dados a partir do processo de monitoramento. Em seguida, é realizada uma extração de padrões das informações obtidas, possibilitando uma caracterização de comportamento normal, que pode ser descrito como o perfil ou assinatura da rede. Por fim, a detecção de intrusão será realizada quando uma nova amostra de tráfego divergir do padrão de comportamento traçado na etapa anterior. Esse trabalho, inclusive, realizará uma abordagem baseada nesse tipo de sistema.

Ambos métodos têm utilizado técnicas de inteligência artificial para realizar a detecção de intrusões ou caracterização de tráfego. Em [17] por exemplo, Carvalho et al. propõem o ACODS (*Ant Colony Optimization for Digital Signature*). Trata-se de um

esquema que realiza modificações na metaheurística de otimização da colônia de formigas (ACO) para gerar a assinatura digital usando a abordagem de clusterização. O ACO é um seguimento de computação evolutiva, que faz parte do campo de inteligência artificial, e, de forma sucinta, simula o comportamento de um conjunto de formigas tendo em vista que, de acordo com Deneubourg et al. [39], elas são capazes de encontrar o menor caminho existente entre sua colônia e uma fonte de comida.

Hamamoto et al. [18] desenvolveram um sistema de detecção de intrusão utilizando Algoritmo Genético e lógica *fuzzy*. O sistema desenvolvido é capaz de coletar o tráfego da rede e traçar sua assinatura ao mesmo tempo. A extração de padrões dos dados coletados é realizada pelo GA, que posteriormente possibilita a predição de comportamento da rede em um determinado momento. A lógica *fuzzy* é aplicada para discriminar se uma determinada amostra é anomalia ou não. O método abordado demonstrou um bom desempenho comparado com outros métodos testados, obtendo 96,53% de acurácia.

As técnicas abordadas nos dois trabalhos citados anteriormente são baseadas em algoritmos de otimização, mas existem pesquisas que fazem uso de outros métodos para desenvolver um IDS. Por exemplo, em [40] os autores aplicam a *Random Forest* (RF), um algoritmo de classificação capaz de detectar e classificar diferentes tipos de ataques. O modelo consiste em gerar um conjunto contendo várias árvores de decisão, formando um classificador robusto e com uma baixa taxa de erro. No trabalho os autores também fazem uso de uma técnica de pré-processamento chamada discretização para em seguida realizar a seleção de características mais relevantes através do método *Symmetrical uncertainty*, que mede a correlação das amostras de tráfego.

3.3 Arquitetura

Como já abordado anteriormente, porém de maneira sucinta, um sistema de detecção de intrusão baseado em anomalias precisa seguir algumas etapas para a sua construção. A Figura 2 ilustra esse processo de maneira geral.

Os dados de tráfego primeiramente são coletados a partir de um sistema monitorado. Em uma rede SDN, por exemplo, tal procedimento pode ser realizado pelo protocolo OpenFlow, que é capaz de trazer diversas informações sobre o comportamento da rede, como por exemplo a quantidade de *bits* ou pacotes em uma determinada faixa de tempo.

A fase de pré-processamento é responsável por limpar ou preparar as informações coletadas para a fase de caracterização do tráfego. Pode aplicar algumas técnicas de processamento de dados, como por exemplo uma redução de dimensionalidade na base de dados [41], [42] ou uma seleção de características mais relevantes [43]. Esse pode ser considerado o passo mais importante na implementação da ferramenta, pois quanto maior a qualidade na base de dados, melhor será o desempenho na fase aprendizado.

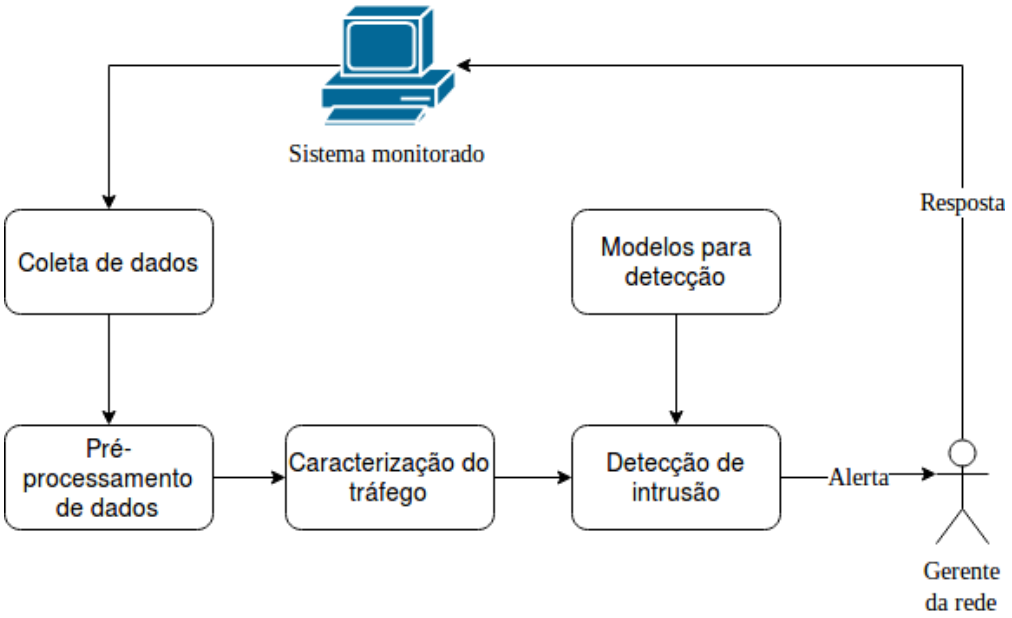

Figura 2 – Arquitetura geral um Sistema de Detecção de Intrusão.

Na caracterização do tráfego é gerado o perfil da rede que descreve o comportamento considerado normal. As intrusões são detectadas quando um comportamento difere significativamente desse perfil traçado. Em ambas as fases podem ser aplicados modelos estocásticos [44] ou do ramo de inteligência artificial [45] para efetuar tais tarefas.

Um IDS ao detectar uma intrusão pode gerar um alerta para o gerente da rede decidir o que deve ser feito. Mas em [19], Carvalho et al. desenvolveram um sistema capaz de mitigar ameaças detectadas a partir de regras preestabelecidas, o que torna o sistema ainda mais autônomo e independente de intervenções humanas.

Este capítulo apresentou uma ferramenta proposta para auxiliar na gerência e segurança de redes. Foram descritos também alguns trabalhos que realizaram sua implementação utilizando diferentes modelos de aprendizado de máquina. O capítulo seguinte introduzirá a heurística empregada para desenvolver a ferramenta neste trabalho.

4 SISTEMA IMUNOLÓGICO

Este capítulo aborda uma descrição introdutória do Sistema Imunológico e aborda suas principais características que podem ser simuladas computacionalmente.

4.1 Visão biológica

O SI protege o corpo humano contra agentes infecciosos, conhecidos também como patógenos. Seu principal objetivo é diferenciar as células do organismo de indivíduos externos, e esse processo é chamado de distinção *self/non-self* [46], [3]. Pode ser um problema bastante complexo, já que o número de invasores (*non-self*) costuma ser muito maior que o de células próprias (*self*). A eficiência do sistema, portanto, é medida baseando-se nesta habilidade de distinção [47].

Esse sistema é estruturado em camadas, como mostra a Figura 3. A camada fisiológica e a física são barreiras naturais do corpo, sendo a pele e barreiras bioquímicas alguns exemplos. As duas formam o mecanismo de defesa do ser humano junto com o sistema imunológico inato e o sistema imunológico adaptativo.

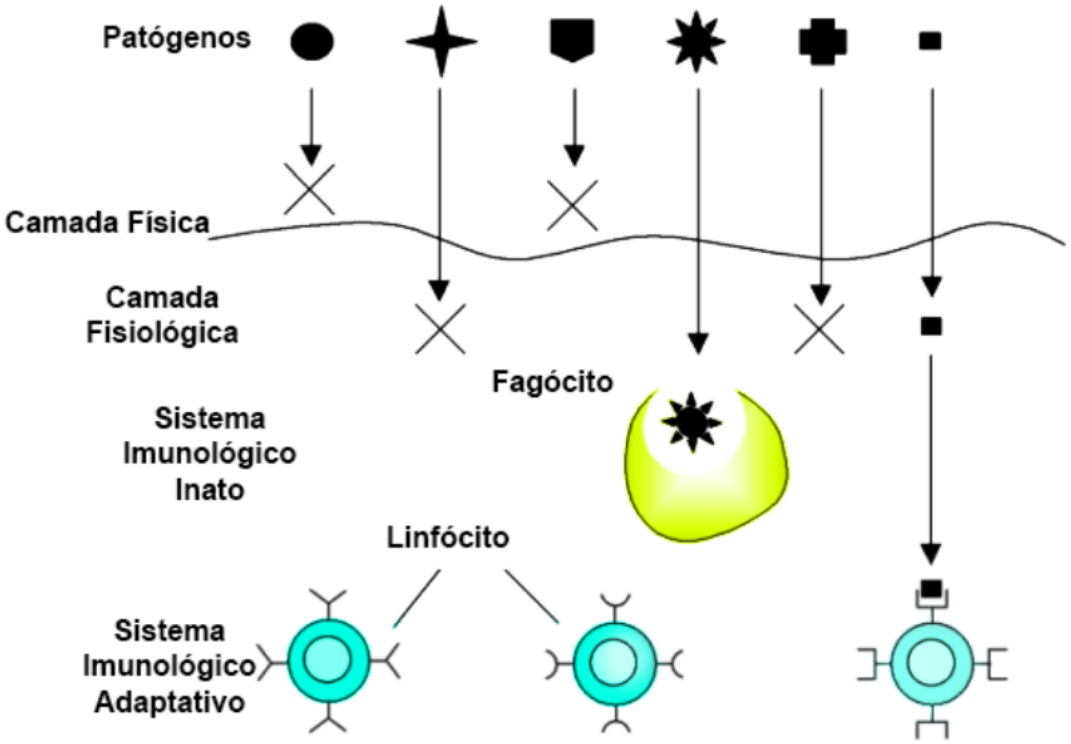

Figura 3 – Estrutura do Sistema Imunológico [2].

Existem dois tipos de imunidade no SI, a inata e a adaptativa. A imunidade

inata é o mecanismo de defesa do organismo que já nasce com o indivíduo e ela nunca é modificada. Esta não é direcionada a um tipo específico de invasor, e sim contra qualquer patógeno que entra no corpo. Quando encontrada alguma infecção, ela é a primeira linha defensiva a ser ativada e posteriormente comunica o sistema imune adaptativo que por sua vez ativa seu sistema defensivo. A imunidade adaptativa se desenvolve com o tempo, pois suas células, conforme entram em contato direto com algum agente infeccioso, são capazes de memorizar tal acontecimento e aprender como neutralizar ou eliminar a infecção. Essa habilidade pode ser descrita como memória imunológica.

Qualquer substância capaz de gerar uma resposta imunológica é considerada um antígeno. Um antígeno não é exatamente um indivíduo, e sim uma toxina ou enzima que se encontra no invasor. As células do sistema imune são responsáveis por combater tais substâncias maliciosas encontradas no organismo, e elas fazem parte de dois grandes grupos: os macrófagos, responsáveis pela fagocitose, e os linfócitos, que produzem anticorpos e destroem os antígenos. A resposta imune, processo que detecta e destrói um agente patogênico, envolve a cooperação celular entre elas.

A fagocitose é o processo de englobamento e digestão de substâncias no organismo, como por exemplo microrganismos invasores e células mortas. No sistema imune os responsáveis por detectar e fagocitar partículas anômalas são os macrófagos, que atuam no sistema imunológico inato. Essas células são as primeiras a perceber a presença de agentes patogênicos e também são quem orientam as respostas imunes, ativando os linfócitos quando necessário.

Os linfócitos podem ser classificados como células B ou T. O linfócito B tem como principal função a produção e secreção de anticorpos que se conectam na superfície do invasor neutralizando o mesmo para posteriormente ele ser fagocitado. Atua no sistema imune adaptativo e é produzido na medula óssea assim como o linfócito T. O que difere o linfócito T do B é que ele migra para o timo para amadurecer e aprender a diferenciar o que faz parte do próprio organismo e o que não faz. Durante o processo de amadurecimento que ocorre nesse órgão, essas células são classificadas como auxiliares (CD4) ou citotóxicas (CD8), dependendo do receptor de membrana que a célula específica desenvolver. Quando maduras elas saem do timo para circular pelo corpo. Os CD4 coordenam todo o mecanismo de defesa do corpo. A partir de informações recebidas dos macrófagos eles solicitam a produção de anticorpos no linfócito B caso seja necessário, senão ativam os CD8 que por sua vez destroem as células infectadas pelos antígenos.

4.2 Visão computacional

O sistema imunológico humano tem proporcionado inspiração para os campos de ciência da computação e engenharia. Definido na década de 80 [48], o Sistema Imunológico

Artificial (SIA) é uma heurística que faz parte da área de computação evolutiva. Esse campo de estudo é um ramo da inteligência artificial e é usado na resolução de problemas, otimização e detecção de anomalias.

Um SIA tem como principais características sua robustez, tolerância a erros e capacidade adaptativa. Existem diversos modelos para uma grande variedade de aplicações, por exemplo, o reconhecimento de novos padrões, onde tal processo é simulado baseado na capacidade que as células defensivas apresentam de diferenciar organismos próprios e não próprios do corpo. No corpo humano essa é uma função do sistema imune adaptativo e quem a realiza são os linfócitos B e linfócitos T, que produzem anticorpos e eliminam os agentes patogênicos. Outros campos que podem fazer uso de tal heurística são segurança, detecção de anomalias, resolução de problemas e análise de dados.

4.2.1 Seleção Negativa

Proposto por Forrest [3], o Algoritmo de Seleção Negativa (*Negative Selection Algorithm, NSA*) é o algoritmo mais popular no ramo de Sistema Imunológico Artificial e suas principais aplicações são voltadas para classificação e reconhecimento de padrões. Seu desenvolvimento é inspirado no processo que ocorre no timo durante o amadurecimento das células T, que possuem receptores em suas superfícies capazes de detectar agentes patogênicos. Tal reconhecimento da origem ao conceito de distinção *self/nonself*. Simula-se a reação das células defensivas quando expostas a qualquer substância ou organismo invasor capaz de gerar uma resposta delas.

O ponto de partida do algoritmo é definir o conjunto de amostras *self*, que representam o comportamento normal do sistema e referem-se aos atributos de entrada. Em seguida é realizada a fase de aprendizado, que consiste na geração do conjunto *non-self* e é representado por detectores que reconhecem o complemento *self*. A Figura 4 ilustra o funcionamento descrito.

Atualmente existem muitas versões do algoritmo representando os dados de diferentes formas e cada um variando a complexidade computacional para gerar os detectores de forma eficiente. O método original, proposto por Forrest et al. [3] e descrito como geração exaustiva de detectores por Araya et al. [49], seguia os seguintes passos: (1) dado o conjunto *self* (2) gera-se um candidato a detector aleatoriamente; e (3) se tal detector combinar com qualquer amostra *self* ele é descartado, se não inserido no conjunto *non-self*.

Após a geração do conjunto *non-self* ele é posto para monitorar o sistema. A ideia consiste em ficar combinando toda nova amostra recebida com os detectores. Se houver combinação entre qualquer um deles significa que está ocorrendo uma anomalia.

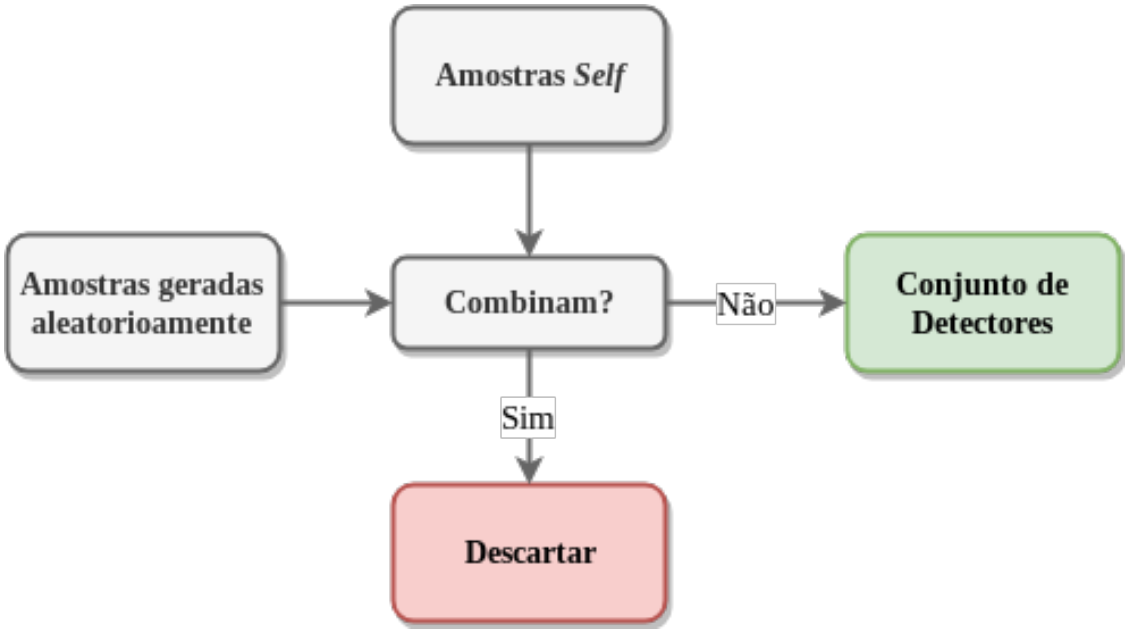

Figura 4 – Algoritmo Seleção Negativa. Adaptado de Forrest et al. [3].

4.2.1.1 Regra de combinação

Os dados observados podem ser representados como *strings* ou valores reais. Quando definido o conjunto *self* como S, o conjunto *non-self* R é gerado a partir de amostras aleatórias que são comparadas com as amostras em S. Nesse processo utilizam-se regras de combinações dependendo do tipo de dado.

No trabalho [50] por exemplo, Dasgupta et al. representam os dados como *strings* de tamanho l e usam uma regra baseada no grau de similaridade denominada *r-contiguous matching rule*. A ideia consiste em verificar se existem pelo menos r caracteres contínuos em qualquer posição das *strings* comparadas, sendo $r \leq l$. O autor também ressalta que um número ótimo de r não deve ser próximo de 1 nem próximo de l, pois no primeiro caso os detectores gerados serão combinados com muitas *strings self* causando falsas detecções; já no segundo os símbolos serão idênticos em todas as posições das *strings* comparadas, o que faz necessário que seja produzido um grande número de detectores para detectar padrões no conjunto *non-self*.

O exemplo ilustrado na Figura 5 apresenta X e Y sendo duas palavras definidas por quatro letras do alfabeto (a, b, c e d). X e Y combinam em três posições contínuas. Logo, *combinam(X, Y)* é verdadeiro para $r \leq 3$ e falso para $r > 3$.

Geralmente a maioria dos trabalhos que aplicam seleção negativa utilizam *strings* binárias para representar seus dados, e existem outras regras de combinação para atender a demanda, tais como *r-chunks* e distância *Hamming* [51], [52]. Segundo Ji et al. o motivo da escolha de representação binária é a facilidade de analisar e classificar os dados, já que o método fornece um espaço finito do problema. No entanto, em [53] o autor utiliza valores

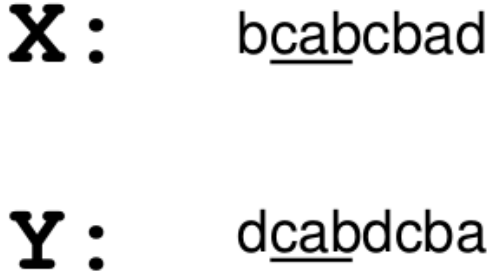

Figura 5 – Ilustração da regra de combinação apresentada por Dasgupta et al. [4].

reais para representar os dados [54]. Como regra de combinação é utilizada a distância Euclidiana entre a coordenada de uma amostra e a de um detector.

Independente da regra utilizada, os detectores são gerados seguindo basicamente a mesma ideia, definindo um certo limiar: o número de caracteres para representação binária e a distância limite para representação de valores reais.

4.2.2 Seleção Clonal

Esse conceito foi sugerido pela primeira vez por Burnet (1959), e sua proposta é apresentar uma memória evolutiva para responder uma grande variedade de antígenos. Isso se dá devido à capacidade que as células B têm de reconhecer microrganismos anômalos no sistema de forma que as células que possuírem uma maior reação em relação ao antígeno seja clonado para combatê-lo.

A heurística também faz uso do conceito de memória imunológica do sistema imune. As informações contidas no sistema em relação aos antígenos são passadas das células pais para as filhas através de uma herança genética. Isso faz com que, quando o corpo é exposto a um mesmo vírus pela segunda vez, por exemplo, o sistema imunológico terá uma resposta mais rápida. No entanto, isso não significa imunidade total ao vírus, mas sim que o indivíduo terá pelo menos mais resistência.

O princípio do funcionamento da seleção clonal, ilustrado na Figura 6 consiste em duas condições: (1) se os linfócitos reconhecem amostras do próprio organismo são eliminados e (2) aqueles que reconhecem as *non-self* são estimulados a se proliferar e diferenciar em células de memória e plasmócitos.

Existem variações em questão de implementação quando se trata da teoria da seleção clonal. Em [55], por exemplo, é abordado um algoritmo denominado Operador Clonal de Anticorpos. Trata-se de um mapeamento estocástico, visto que são abordados os princípios básicos de um SIA, onde é realizada uma media de afinidade entre os antígenos e anticorpos.

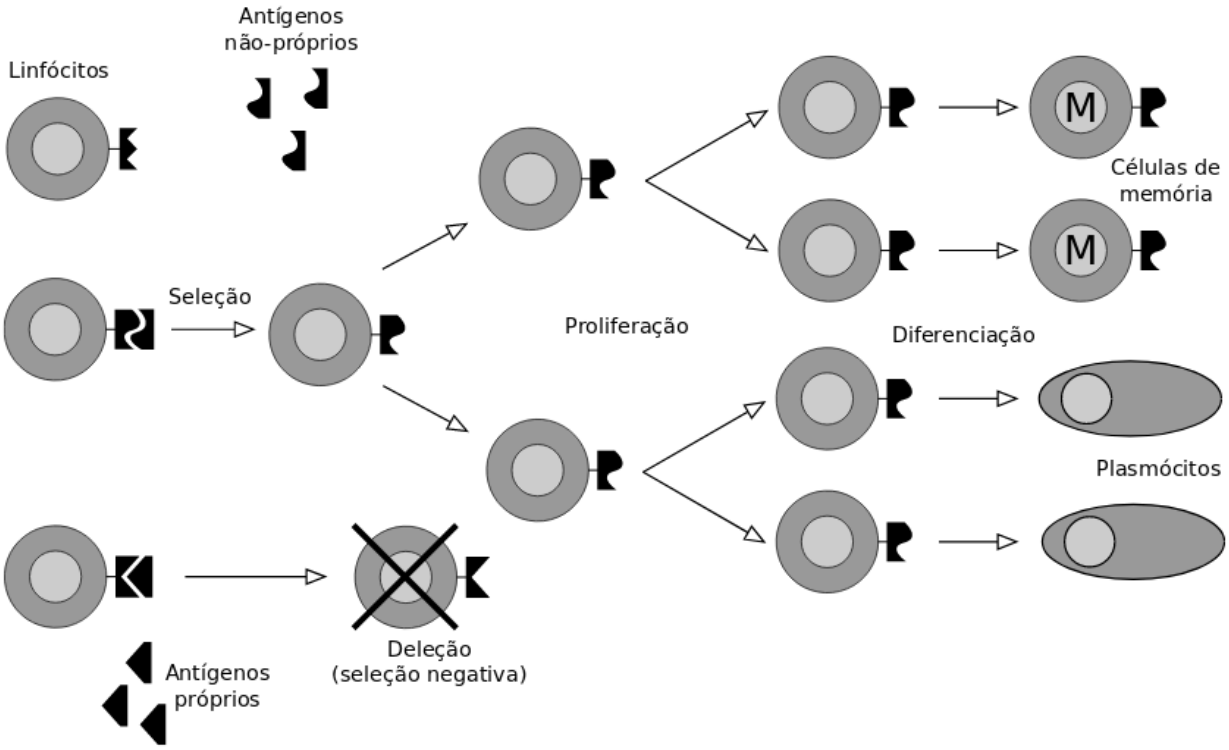

Figura 6 – Esquematização do funcionamento da seleção clonal [5].

4.2.3 Teoria do Perigo

O método mais recente na área de imunologia, a Teoria do Perigo foi introduzida por Matzinger [56]. O conceito ainda é controverso, já que desafia o conceito de que uma resposta imunológica é estimulada através do reconhecimento de qualquer amostra *non-self* encontrada no organismo [57]. A autora defende que o processo defensivo ocorre a partir de danos causados por agentes patogênicos no corpo humano, ou seja, uma anomalia pode ser detectada mas não necessariamente ela apresenta perigo e desencadeia uma ação de macrófagos e linfócitos.

Muitos autores já apresentaram diferentes abordagens dessa heurística aplicando-a em sistemas de detecção de intrusão, mas a ideia a princípio foi proposta por Aickelin et al. [58]. Ele afirma que a Teoria do Perigo pode substituir o conceito de distinção *self/non-self* que fora abordado em [3], [47], [50]. Em seu trabalho, o autor relata que sistemas imunológicos artificiais já obtiveram sucesso em pequenas aplicações apresentando os principais benefícios extraídos do sistema imunológico humano, que são tolerância a erro, adaptação e monitoramento próprio. A proposta ainda não foi aceita entre os imunologistas, porém o autor afirma que a Teoria do Perigo será a próxima geração do Sistema Imunológico Artificial.

Um modelo baseado em sistema multiagentes utilizando Teoria do Perigo é proposto por Ou et al. [59]. Antígenos, células dendríticas e células T compõe a comunidade

de agentes que coordenam mutuamente para realizarem respostas imunes. Agentes AG (*Antigen Agent*) são responsáveis por receber os dados de entrada, que se tratam dos pacotes de *IP* (endereço, porta, protocolo, etc), e enviá-los para os agentes DC (*Dendritic Cell Agent*), que são quem calculam o valor do perigo que as informações recebidas apresentam. Caso um antígeno ultrapasse um limiar de perigo os agentes TC (*T Cell Agent*) informam um quarto agente, o RP (*Responding Agent*). Este está instalado em uma Central Operacional de Segurança e é quem elimina a atividade maliciosa.

Tendo em vista os principais benefícios dos modelos oferecidos pelo SIA, o desenvolvimento deste trabalho tratará de aplicar o NSA no desenvolvimento de um IDS em um ambiente caracterizado por tráfegos de redes SDN a fim de detectar intrusões e anomalias.

5 TRABALHOS RELACIONADOS

Atualmente, com a constante expansão de redes de computadores e dispositivos interconectados, a Segurança da Informação tem se tornado uma área de pesquisa contínua junto com o desenvolvimento de novos ataques aos componentes e informações da rede. Detecção de Intrusão é um dos principais assuntos abordados dentro dessa área de estudo e tem como objetivo identificar atividades maliciosas a partir de monitorações em um determinado sistema. Técnicas de Aprendizado de Máquina têm sido amplamente aplicadas nos desenvolvimentos dessas ferramentas.

Um Algoritmo Genético (*Genetic Algorithm, GA*) é um método de análise de dados que funciona analogamente à Teoria da Evolução de Darwin e é comumente usado na computação em problemas de otimização. Owais et al. [60] mostram, por meio de de diversos trabalhos relacionados, que o GA pode ser aplicado de diferentes formas para o desenvolvimento de um Sistema de Detecção de Intrusão baseado em Inteligência Artificial. Nesse trabalho são apresentados resultados impressionantes, com alta taxa de detecção e baixa taxa de falso positivo. Mas este não é o único, existem diversas publicações nessa mesma área de computação evolutiva apresentando ótimas precisões quando se trata de detecção de anomalias.

O Sistema Imunológico Humano possui diversas células que realizam diferentes tarefas, dessa forma, algoritmos computacionais podem ser desenvolvidos baseando-se em técnicas dentro dessa área. Os modelos inspirados nessa linha de pesquisa são chamados de Sistemas Imunológicos Artificiais (*Artificial Immune Systems, AIS*) e comumente usados para reconhecimento de padrões, segurança de computadores, detecção de anomalias e muitas outras aplicações. Gonzalez et al. [61] descrevem uma abordagem inspirada em Sistemas Imunológicos que é capaz de detectar uma grande variedade de atividades anômalas em redes de computadores. A técnica abordada trata-se do mecanismo de Seleção Negativa que consegue gerar um conjunto de detectores de comportamentos anormais. Tais detectores são elementos contrários às amostras do conjunto inicial, presentes no complemento do espaço, e são otimizados através de Algoritmos Genéticos, onde estes conseguem distinguir diversos níveis de anomalias no tráfego de rede.

Em [62], o autor também implementou um IDS aplicando o conceito de Seleção Negativa, onde GA é aplicado para gerar detectores de anomalia para o sistema. Porém, no trabalho [63], o conjunto de treinamento que possui comportamentos considerados normais da rede passa por um processo de seleção de características para gerar um subconjunto melhor e obter detectores mais eficientes. Tal processo tem como principal objetivo reduzir a dimensão do problema e selecionar apenas informações mais relevantes.

O MILA (*Multilevel Immune Learning Algorithm*) é um algoritmo inspirado na resposta imune humoral da célula T. Proposto em [64], esse algoritmo mede o grau de relacionamento entre antígenos e anticorpos a partir da distância Euclidiana, e pode ser dividido em fases: (1) de inicialização, onde dados *self* são recolhidos; (2) reconhecimento, onde existem grupos de conjuntos de detectores para realizar um reconhecimento multinível; (3) evolucionária, processo no qual os detectores são refinados; e (4) resposta, que consiste na eliminação de antígenos.

A Teoria do Perigo é um segmento do campo de Sistemas Imunológicos. Inspirados nela, Haidong Fu et al. [65] apresentaram o modelo de um Sistema Imunológico Artificial Multiagente para a detecção de anomalias, chamado MAAIS (*Multi-agents Artificial Immune System*). O MAAIS é um sistema de defesa robusto que tem como objetivo central a capacidade de se adaptar aos ataques recebidos em uma rede, por mais distintos que sejam.

Chowdhury [16] combinou duas técnicas de Aprendizado de Máquina (*Machine Learning, ML*) para detecção de intrusão em redes. A proposta é utilizar a meta-heurística *Simulated Annealing* (SA) para reduzir a dimensão da base de dados, onde um determinado número de atributos é combinado aleatoriamente e então aplicado no algoritmo de Máquinas de Vetores Suporte (*Support Vector Machine, SVM*) para detectar comportamentos anômalos nos dados de tráfego. Nesse trabalho foi possível mostrar que a combinação de SA e SVMs pode aumentar a acurácia de um sistema de detecção de intrusão.

A caracterização do tráfego é a fase mais importante no desenvolvimento de um sistema de detecção de intrusão pois permite ao gerente de redes identificar limitações e pontos cruciais em uma rede [66]. As pesquisas nessa área crescem constantemente e existem diversos trabalhos que realizam diferentes abordagens a respeito. Adaniya et al. [67], por exemplo, propuseram a meta-heurística *Firefly Harmonic Clustering* para gerar um sistema de detecção de anomalias. A proposta consiste em combinar a técnica de clusterização *K-Harmonic Means* (KHM) com o Algoritmo dos Vagalumes (*Firefly Algorithm*, FA) para caracterizar o tráfego. O KHM é uma técnica que busca encontrar padrões na base de dados através de agrupamentos de amostras para alcançar resultados melhores no final. O FA é um algoritmo baseado na intensidade das luzes emitidas pelos vagalumes. Sabendo disso, os autores aplicaram essa combinação dessas técnicas para detectar anomalias em redes baseado no volume de tráfego.

A ferramenta CPM (*Correlational Paraconsistent Machine*) apresentada nos trabalhos [68] e [69] foi usada na construção de um modelo não supervisionado para caracterização de tráfego a partir da sua lógica paraconsistente. O perfil da rede pôde ser traçado utilizando o modelo *Autoregressive Integrated Moving Average* (ARIMA) e a meta-heurística *Ant Colony Optimization* (ACO). A detecção de anomalias ocorre a

partir da análise de irregularidades no sistema, como por exemplo falta de informações ou ruídos nos fluxos de dados.

Em [70] os autores também focaram na caracterização de tráfego, buscando traçar uma assinatura digital (*Digital Signature of Network Segment*, DSNS) em servidores de rede. Para isso eles fazem uso do modelo BLGBA (*Baseline for Automatic Backbone Management*), que foi desenvolvido baseado em análises estatísticas para cada segundo do dia durante todos os dias da semana. Com isso, foi possível calcular a variação da moda nos dados coletados. O trabalho [71] também contribuiu com um estudo sobre a geração do DSNS, no qual esse processo era realizado através de limiares que refletiam no volume de tráfego esperado em determinada faixa de tempo ao longo do dia.

Em [72], a assinatura digital foi abordada de uma maneira diferente das anteriores. Os autores apresentaram três métodos para criar uma assinatura baseada no fluxo da rede (*Digital Signature of Network Segment Using Flow Analysis*, DSNSF): o procedimento estatístico de Análise de Componentes Principais, a metaheurística de otimização da colônia de formigas o método de previsão *Holt-Winters*. Uma análise realizada a partir de sete dimensões do fluxo de rede foi realizada para gerar a assinatura digital em [73], [74]. Os trabalhos analisaram cada dimensão para caracterizar o tráfego no intuito de gerar um perfil de comportamento mais robusto para assim melhorar o gerenciamento de redes com um detector de anomalias eficaz.

6 ESTUDO DE CASO

A meta-heurística de Sistemas Imunológicos Artificiais pode ser aplicada em diferentes áreas da computação, como a segurança da informação por exemplo. Ela pode ser utilizada na implementação de uma ferramenta para auxiliar na gerência de uma Rede Definida por Software que trabalhe na prevenção de ataques. O algoritmo de Seleção Negativa em específico apresenta alta eficiência quando se trata de detecção de anomalias. Sabendo disso, é possível construir um sistema que detecte intrusos a partir de bases de dados que contenham informações sobre o tráfego da rede e que sejam consideradas normais. A ideia central é gerar, a partir desse conjunto de entrada classificado com comportamento padrão, o complemento, que seria todo o comportamento anômalo, do tráfego e realizar a distinção *self/non-self* para detectar anomalia.

Com o objetivo de avaliar o modelo, o sistema foi desenvolvido utilizando a linguagem *Python*, que é uma linguagem de programação de alto nível, interpretada, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte [75]. As bibliotecas utilizadas foram o *Pandas*, para manipulação de dados, *Matplotlib*, para plotagem de gráficos e o *Scikit-Learn*, que possui funções para avalias o desempenho do IDS proposto.

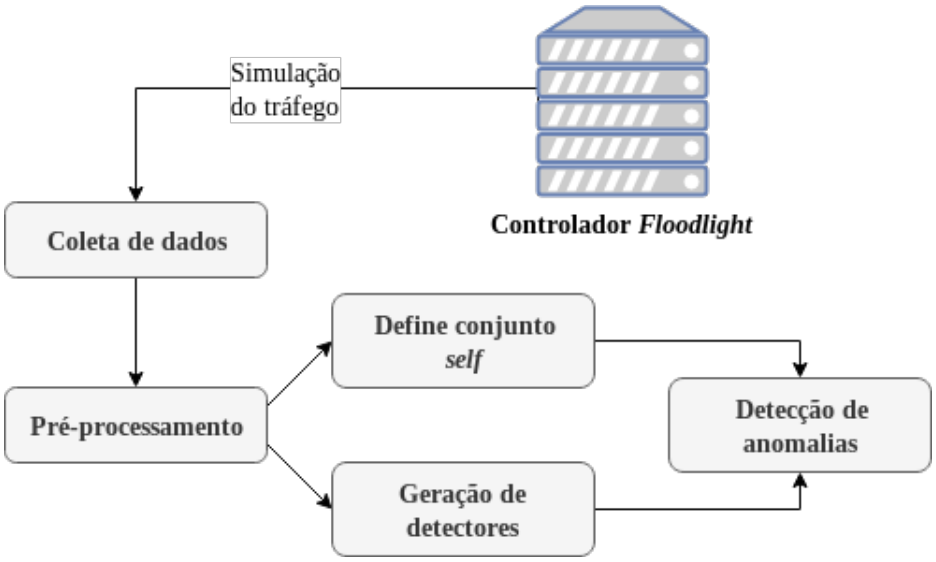

Figura 7 – Fluxograma demonstrando a arquitetura da implementação.

A arquitetura da implementação pode ser observada na Figura 7. Cada uma das etapas será explicada a seguir.

6.1 Dados

Os dados de tráfego pertencem a uma rede SDN e foram emulados pelo controlador *Floodlight* utilizando o *OpenFlow*. A partir da análise do fluxo *IP* foram coletados os seguintes atributos: número de *bits*, quantidade de pacotes, entropia *IP* origem, entropia *IP* destino, entropia porta origem e entropia porta destino. Em seguida estes foram armazenados em arquivos de texto.

6.2 Pré-processamento

Os fluxos são agrupados em intervalos de 30 segundos, o que totaliza 2880 amostras do tráfego por dia. As informações são divididas por dimensão, contendo variáveis quantitativas (pacotes e *bits*) e qualitativas (entropias). Os atributos não numéricos (portas e *IPs*) precisaram ser abordados utilizando o conceito de entropia para encontrar a concentração e a representação numérica para as suas informações.

Para este trabalho foi utilizado apenas o atributo de quantidade de pacotes por segundo, e o mesmo foi normalizado entre 0 e 1 seguindo a Equação 6.1.

$$X_i = \frac{X_i}{X_{max}} \tag{6.1}$$

6.3 Conjunto *Self* e Geração de Detectores

A definição do conjunto *self* deu-se a partir de 4 dias (4 bases) de tráfego, calculando a média de seus intervalos respectivos. A Figura 8 apresenta a plotagem dos gráficos referentes aos dias 1 ao 4 do tráfego contendo um comportamento considerado normal, ao ataque DDoS simulado no Mininet e ao conjunto *self* gerado a partir das quatro bases de tráfego normal.

Em seguida, o conjunto *self* foi dividido utilizando o conceito de janela deslizante no intuito de reduzir a dimensionalidade do problema e aumentar o poder preditivo do sistema. Cada janela possui um tamanho fixo de 20 intervalos, o que representa 10 minutos de tráfego; e um deslocamento de 4 intervalos de tempo, o que significa que as janelas se movem de 2 em 2 minutos.

Tendo isso, um detector foi gerado para cada uma das 720 janelas. Para este processo foram realizados os seguintes passos: (1) geração de de um detector aleatório; (2) comparação deste com cada amostra da janela determinada; (3) inserção do detector no conjunto *non-self* caso o mesmo não combine com nenhuma das amostras comparadas, se não o processo volta para o passo (1).

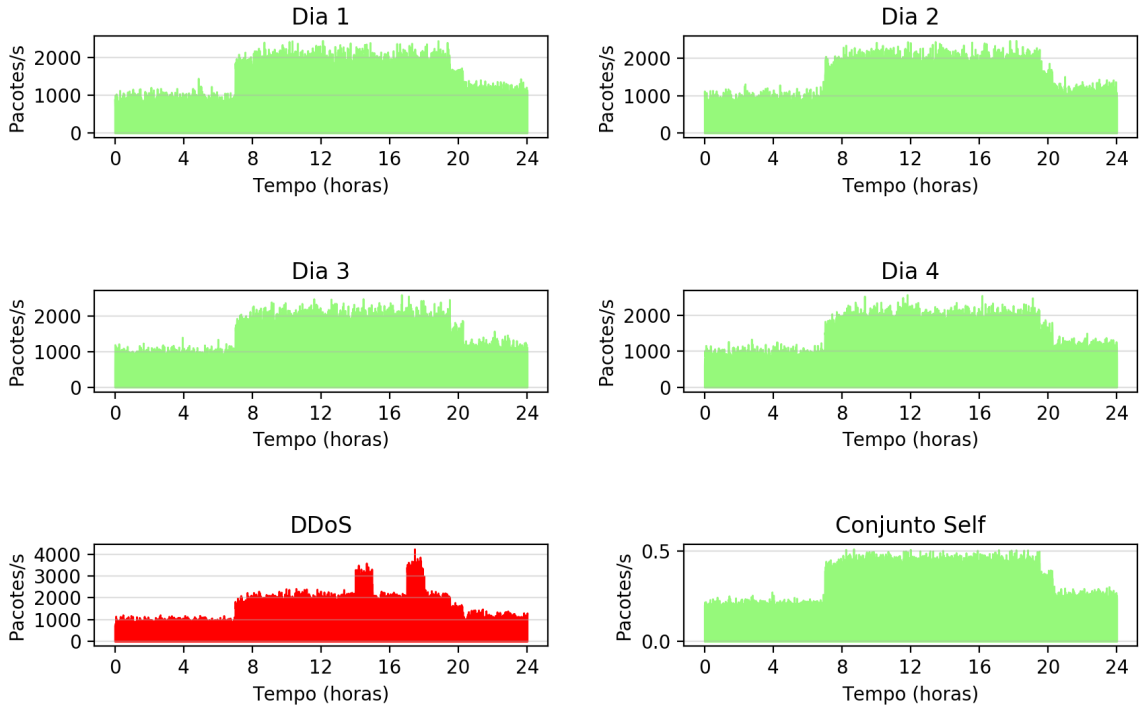

Figura 8 – Representação dos tráfegos e do conjunto *self*.

A regra de combinação utilizada foi a distância Euclidiana, e ela é dada pela fórmula apresentada na Equação (6.2).

$$\sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{6.2}$$

6.4 Detecção de Anomalia

Na fase que seria equivalente ao monitoramento na Seleção Negativa, foi utilizada uma base de teste pré-definida contendo tráfego com ataques injetados em uma rede emulada no Mininet. Apenas o ataque DDoS foi abordado.

Cada amostra de teste monitorada é comparada com quatro detectores referentes a suas quatro próximas janelas, dado seu intervalo de tempo. A regra de comparação é a mesma usada na geração de detectores. Logo, caso a combinação seja positiva a amostra é classificada como *non-self*, ou seja, anomalia.

6.5 Avaliação do modelo

A base de teste contém um tráfego onde foi injetado um ataque do tipo DDoS nos intervalos de tempo das 13 às 15 horas e das 17 às 19 horas. Trata-se de um ataque onde inúmeras requisições de diferentes origens são enviadas ao servidor, fazendo com que a quantidade de pacotes torne-se um valor elevado.

Na fase de geração de detectores, utiliza-se um limiar para diferenciar as amostras a partir de suas distâncias euclidianas. Neste trabalho foram utilizados quatro limiares distintos no intuito de comparar seus resultados. Essa comparação é importante pois ela permite realizar um balanceamento na performance do modelo.

O algoritmo de Seleção Negativa foi testado com diferentes parâmetros, o que gerou diferentes conjuntos de detectores: a Figura 9 apresenta os detectores gerados com limiar de 0.05; a Figura 10 apresenta os detectores gerados com limiar de 0.10; a Figura 11 apresenta os detectores gerados com limiar de 0.15; e, por fim, a Figura 12 apresenta os detectores gerados com limiar de 0.20.

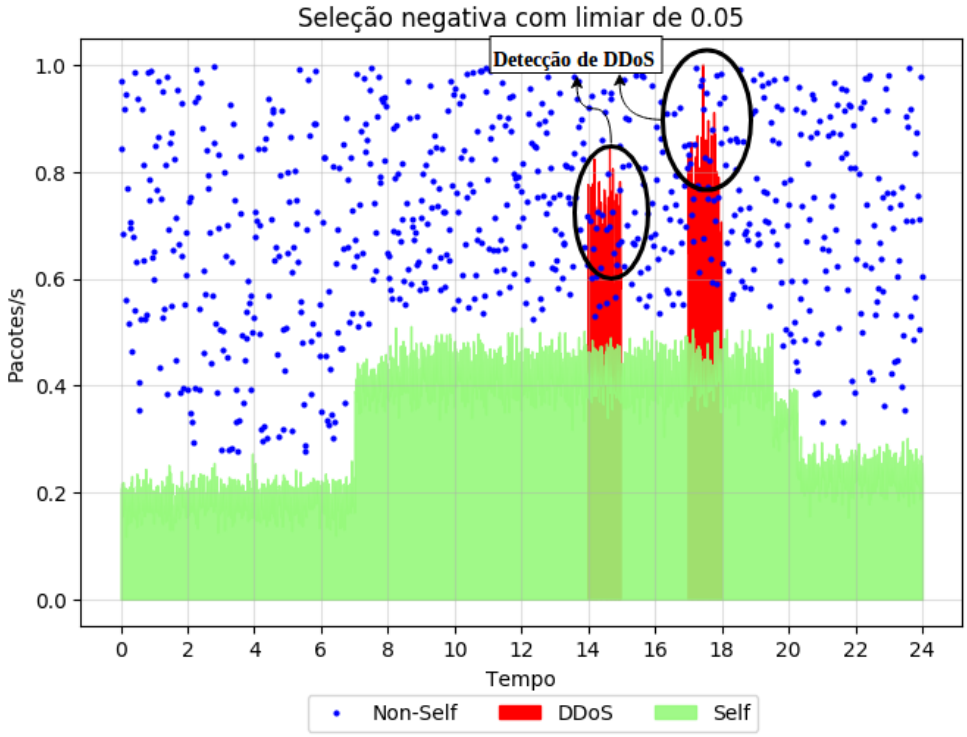

Figura 9 – Detecção de anomalia a partir do conjunto *non-self* gerado com limiar igual a 0.05.

Esses valores serviram para ajustar a regra de combinação. Durante o processo de geração de detectores, caso a distância das amostras fosse menor que o limiar, o detector candidato era descartado, caso contrário ele era inserido no conjunto *non-self*. Já na fase de monitoramento, caso a distância entre as amostras analisadas e os detectores fosse menor que o limiar, uma anomalia era detectada.

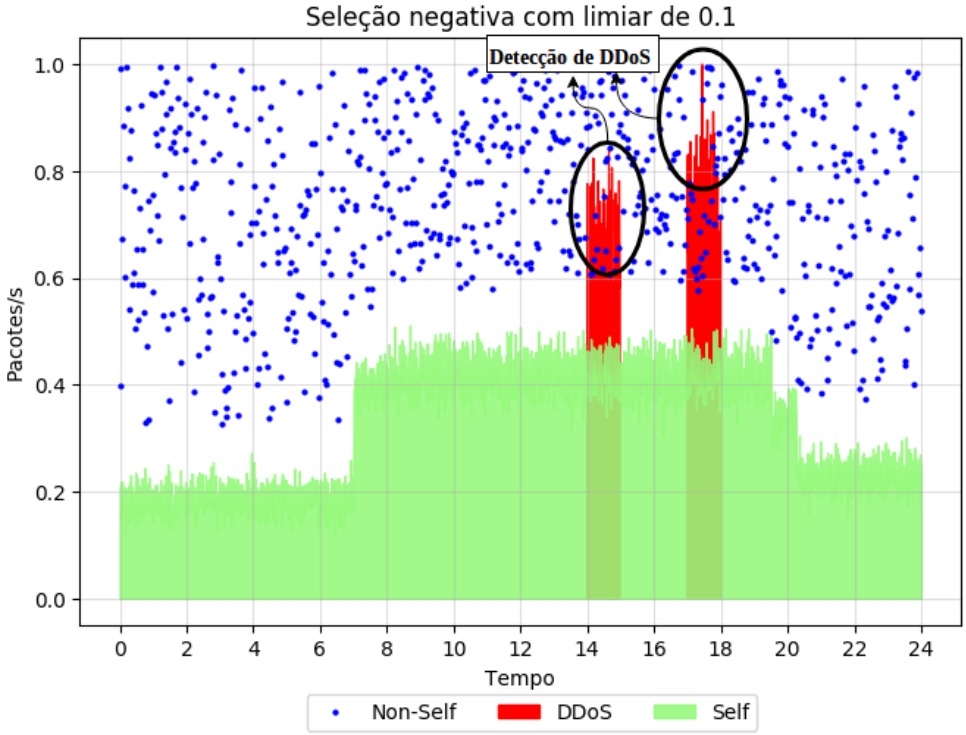

Figura 10 – Detecção de anomalia a partir do conjunto *non-self* gerado com limiar igual a 0.10.

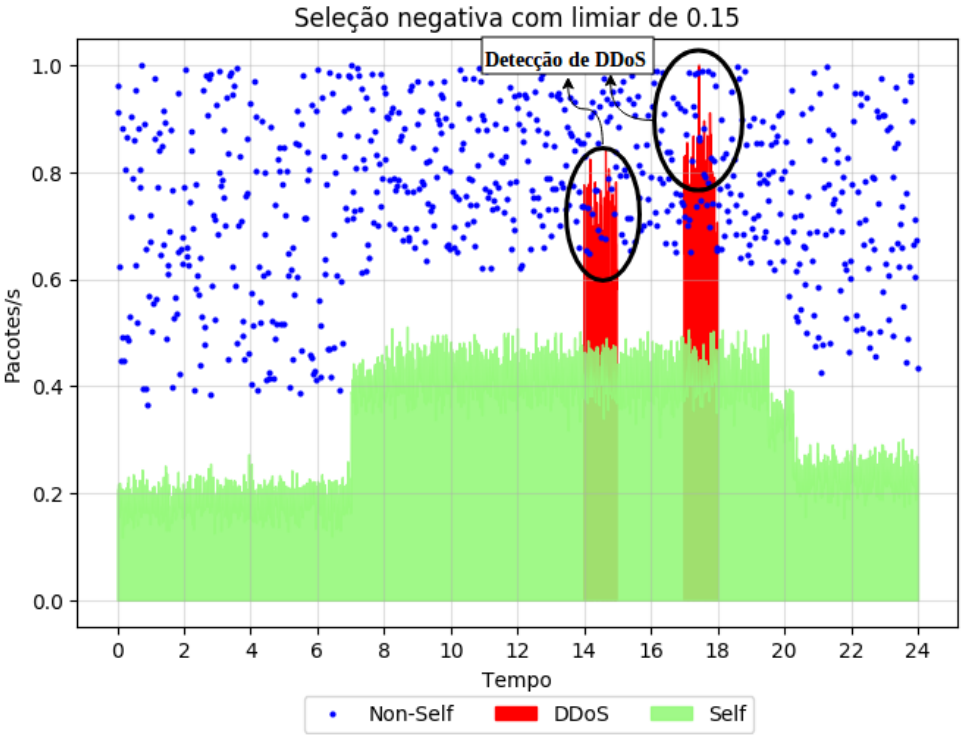

Figura 11 – Detecção de anomalia a partir do conjunto *non-self* gerado com limiar igual a 0.15.

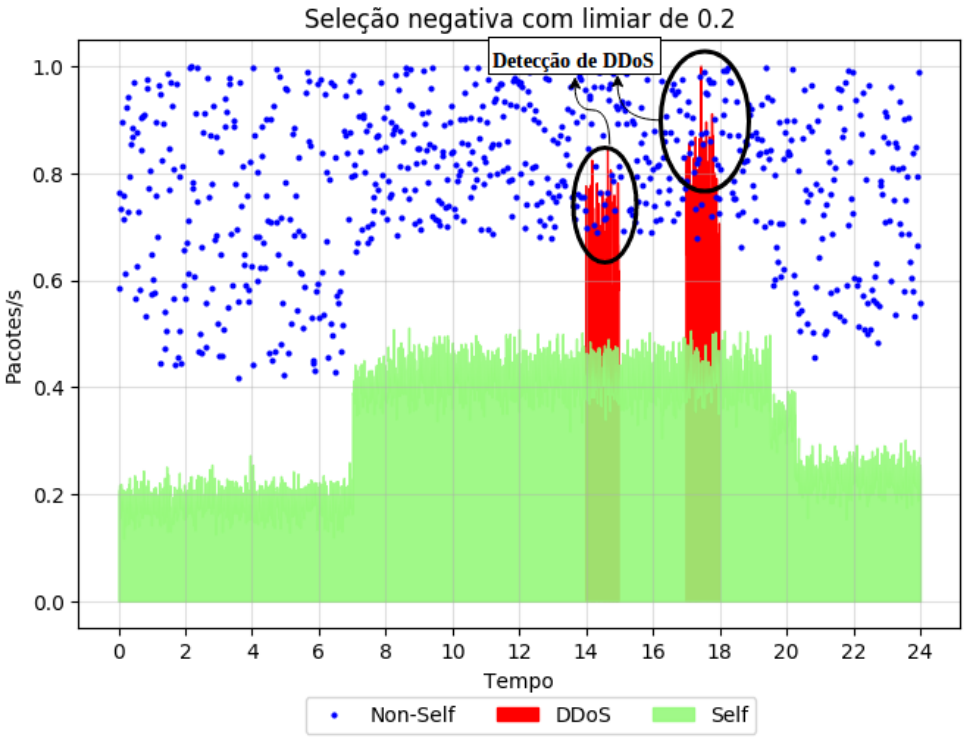

Figura 12 – Detecção de anomalia a partir do conjunto *non-self* gerado com limiar igual a 0.20.

Nas áreas destacadas pelas circunferências é possível visualizar o momento em que as amostras anômalas (em vermelho) foram detectadas pelos detectores (pontos azuis).

Os resultados que cada valor de limiar apresentou é mostrado na Tabela 1.

Limiar	**Acurácia**	**F1**	***Recall***	**Precisão**
0.05	64.20	76.63	99.06	62.48
0.10	92.08	95.56	98.40	92.88
0.15	96.46	98.07	97.08	99.08
0.20	95.10	97.35	94.90	99.92

Tabela 1 – Resultados obtidos com diferentes limiares.

O primeiro limiar apresentou uma performance menor que os demais pois os detectores gerados encontraram-se mais próximos das amostras *self* do que das novas amostras analisadas na fase de monitoramento. Grande parte das amostras de DDoS possuía o dobro do número de pacotes que as amostras *self*, por conta disso os demais limiares apresentaram alta taxa de detecção, pois eles acabaram gerando detectores que encontraram-se distantes das amostras *self*.

Para avaliar a performance do modelo proposto foram utilizadas as métricas de acurácia, que mede o número de predições corretas pelo total de amostras; de precisão, que calcula a taxa de acertos positivos pelo total de amostras; *recall*, onde o seu resultado

é dado pelo número de positivos verdadeiros divididos pelo total de positivos; e o *F1 Score*, uma média harmônica entre precisão e *recall*. Tais valores foram obtidos a partir de funções de medidas de precisão presentes na biblioteca *Scikit-Learn* [76].

7 CONCLUSÃO

O Sistema Imunológico Artificial, inspirado no Sistema Imunológico Humano, possui diversos modelos que podem ser aplicados em detecção de falhas, reconhecimento de padrões, segurança, e diversas outras áreas. A partir de estudos realizados com estes modelos bio-inspirados aplicados em reconhecimento de padrões, este trabalho abordou o algoritmo de Seleção Negativa para o desenvolvimento de um Sistema de Detecção de Intrusão baseado em anomalias em Redes Definidas por Software.

O constante desenvolvimento de controladores deixa nítido o grande incentivo à pesquisa que está ocorrendo dentro da área de Redes Definidas por Software. Fazendo uso de ferramentas relacionadas, foi possível realizar o estudo do modelo proposto pelo trabalho, pois através do controlador Floodlight os tráfegos foram emulados e com o Mininet os ataques puderam ser injetados nos mesmos.

A caracterização do tráfego foi definida pela média dos atributos de pacotes das quatro bases diferentes emuladas pelo controlador Floodlight. Em seguida, o tráfego resultante foi normalizado entre os intervalos de 0 e 1 formando o conjunto *self*. O aprendizado do modelo constituiu-se pela geração de detectores (conjunto *non-self*), e este mostrou-se eficaz na detecção de anomalias utilizando a regra de combinação da distância Euclidiana para a maioria dos limiares estabelecidos.

Uma proposta de melhoria futura para este trabalho seria aplicar ou desenvolver uma técnica de caracterização de tráfego, visto que a abordagem desta etapa deu-se com a média dos tráfegos normalizados e a partir de um atributo apenas. Uma segunda proposta seria aplicar uma técnica diferente apenas para a geração de detectores. A ideia consistiria em gerar um conjunto bem maior e, feito isso, realizar agrupamentos das amostras obtidas no intuito de construir um conjunto *non-self* menor e mais eficaz.

REFERÊNCIAS

[1] MASOUDI, R.; GHAFFARI, A. *Software defined networks: A survey*. 2016. 1–25 p.

[2] CASTRO, L. N. de; ZUBEN, F. J. V. Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 3, p. 239–251, June 2002. ISSN 1089-778X.

[3] FORREST, S. et al. Self-nonself discrimination in a computer. *Proceedings of 1994 IEEE Computer Society Symposium on Research in Security and Privacy*, p. 202–212, 1994. ISSN 1540-7993. Disponível em: <http://ieeexplore.ieee.org/document/296580/>.

[4] AMIRA, A. S.; HANAFI, S. E. O.; HASSANIEN, A. E. Comparison of classification techniques applied for network intrusion detection and classification. *Journal of Applied Logic*, Elsevier B.V., v. 24, p. 109–118, 2017. ISSN 15708683. Disponível em: <http://dx.doi.org/10.1016/j.jal.2016.11.018>.

[5] CASTRO, L. et al. Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation, Special Issue on Artificial. *Immune Systems*, v. 6, n. 3, p. 239–251, 2002. ISSN 1089-778X.

[6] WANG, H.; GU, J.; WANG, S. An effective intrusion detection framework based on SVM with feature augmentation. *Knowledge-Based Systems*, Elsevier B.V., v. 136, p. 130–139, 2017. ISSN 09507051. Disponível em: <http://dx.doi.org/10.1016/j.knosys.2017.09.014>.

[7] De Assis, M. V. et al. A game theoretical based system using holt-winters and genetic algorithm with fuzzy logic for DoS/DDoS mitigation on SDN networks. *IEEE Access*, v. 5, n. c, p. 9485–9496, 2017. ISSN 21693536.

[8] KALKAN, K.; GUR, G.; ALAGOZ, F. Defense Mechanisms against DDoS Attacks in SDN Environment. *IEEE Communications Magazine*, v. 55, n. 9, p. 175–179, 2017. ISSN 01636804.

[9] DIRO, A. A.; CHILAMKURTI, N. Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Generation Computer Systems*, Elsevier B.V., v. 82, p. 761–768, 2018. ISSN 0167739X. Disponível em: <http://dx.doi.org/10.1016/j.future.2017.08.043>.

[10] BEHAL, S.; KUMAR, K.; SACHDEVA, M. Characterizing DDoS attacks and flash events: Review, research gaps and future directions. *Computer Science Review*, Elsevier Inc., v. 25, p. 101–114, 2017. ISSN 15740137. Disponível em: <http://dx.doi.org/10.1016/j.cosrev.2017.07.003>.

[11] TCHAKOUCHT, T. A. I. T.; EZZIYYANI, M. Building A Fast Intrusion Detection System For High-Speed- Building A Fast Intrusion Detection System For High-Speed-Networks : Probe and DoS Attacks Detection Networks : Probe and DoS Attacks Detection. *Procedia Computer Science*, Elsevier B.V., v. 127, p. 521–530, 2018. ISSN 1877-0509. Disponível em: <https://doi.org/10.1016/j.procs.2018.01.151>.

[12] ZHANG, H. et al. Exploring machine-learning-based control plane intrusion detection techniques in software defined optical networks. *Optical Fiber Technology*, Elsevier, v. 39, n. July, p. 37–42, 2017. ISSN 10685200. Disponível em: <http://dx.doi.org/10.1016/j.yofte.2017.09.023>.

[13] HIDAYANTO, B. C. et al. Network Intrusion Detection Systems Analysis using Frequent Item Set Mining Algorithm FP-Max and Apriori. *Procedia Computer Science*, Elsevier B.V., v. 124, p. 751–758, 2017. ISSN 18770509. Disponível em: <https://doi.org/10.1016/j.procs.2017.12.214>.

[14] AMARAL, A. A. et al. Deep IP flow inspection to detect beyond network anomalies. *Computer Communications*, Elsevier B.V., v. 98, p. 80–96, 2017. ISSN 01403664. Disponível em: <http://dx.doi.org/10.1016/j.comcom.2016.12.007>.

[15] M. L. Proença Jr., G. Fernandes, L. F. Carvalho, M. V. O. de Assis, J. J. P. C. R. Digital signature to help network management using flow analysis. *Internation Journal of Network Management*, n. October 2005, p. 17–31, 2014. ISSN 10557148. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1002/nem.604/abstract>.

[16] CHOWDHURY, M. N.; FERENS, K.; FERENS, M. Network Intrusion Detection Using Machine Learning. p. 30–35, 2010. Disponível em: <https://pdfs.semanticscholar.org/a30c/16f5598ba18ffd7d9c533515cf671d54b382.pdf>.

[17] CARVALHO, L. F. et al. Unsupervised learning clustering and self-organized agents applied to help network management. *Expert Systems with Applications*, v. 54, p. 29–47, 2016. ISSN 09574174.

[18] HAMAMOTO, A. H. et al. Network Anomaly Detection System using Genetic Algorithm and Fuzzy Logic. *Expert Systems with Applications*, v. 92, p. 390–402, 2018. ISSN 09574174.

[19] CARVALHO, L. F. et al. An ecosystem for anomaly detection and mitigation in software-defined networking. *Expert Systems with Applications*, v. 104, p. 121–133, 2018. ISSN 09574174.

[20] HA, T. et al. Suspicious traffic sampling for intrusion detection in software-defined networks. *Computer Networks*, Elsevier B.V., v. 109, p. 172–182, 2016. ISSN 13891286. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2016.05.019>.

[21] NUNES, B. A. A. et al. A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys Tutorials*, v. 16, n. 3, p. 1617–1634, Third 2014. ISSN 1553-877X.

[22] FEAMSTER, N.; REXFORD, J.; ZEGURA, E. The road to sdn: An intellectual history of programmable networks. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 44, n. 2, p. 87–98, abr. 2014. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/2602204.2602219>.

[23] FARHADY, H.; LEE, H.; NAKAO, A. Software-Defined Networking: A survey. *Computer Networks*, Elsevier B.V., v. 81, p. 79–95, 2015. ISSN 13891286. Disponível em: <http://dx.doi.org/10.1016/j.comnet.2015.02.014>.

[24] PAKZAD, F. et al. Efficient topology discovery in OpenFlow-based Software Defined Networks. *Computer Communications*, Elsevier Ltd., v. 77, p. 52–61, 2016. ISSN 01403664. Disponível em: <http://dx.doi.org/10.1016/j.comcom.2015.09.013>.

[25] LARA, A.; KOLASANI, A.; RAMAMURTHY, B. Network innovation using openflow: A survey. *IEEE Communications Surveys Tutorials*, v. 16, n. 1, p. 493–512, First 2014. ISSN 1553-877X.

[26] GUDE, N. et al. Nox: Towards an operating system for networks. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 38, n. 3, p. 105–110, jul. 2008. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/1384609.1384625>.

[27] MCCAULEY, M. Pox (online), disponível em: <https://github.com/noxrepo/>. 2009.

[28] SNAC. Disponível em: <https://github.com/bigswitch/snac-nox/blob/master/src/include/openflow/openflow/nicira-ext.h>.

[29] FLOODLIGHT. Documentação disponível em: <http://www.projectfloodlight.org/documentation/>.

[30] MUL. Disponível em: <http://www.openmul.org/openmul-controller.html/>.

[31] MININET. An instant virtual network on your laptop (or other pc), disponível em: <http://mininet.org/>.

[32] NS3. Disponível em: <https://www.nsnam.org/>.

[33] ESTINET. Disponível em: <http://www.estinet.com/ns/>.

[34] AGARWAL, S.; KODIALAM, M.; LAKSHMAN, T. V. Traffic engineering in software defined networks. In: *2013 Proceedings IEEE INFOCOM*. [S.l.: s.n.], 2013. p. 2211–2219. ISSN 0743-166X.

[35] KIM, H.; FEAMSTER, N. Improving network management with software defined networking. *IEEE Communications Magazine*, v. 51, n. 2, p. 114–119, February 2013. ISSN 0163-6804.

[36] MEHDI, S. A.; KHALID, J.; KHAYAM, S. A. Revisiting Traffic Anomaly Detection using Software Defined Networking. *The Lancet*, v. 193, n. 4993, p. 808–811, 1919. ISSN 01406736.

[37] HU, F.; HAO, Q.; BAO, K. A survey on software-defined network and openflow: From concept to implementation. *IEEE Communications Surveys Tutorials*, v. 16, n. 4, p. 2181–2206, Fourthquarter 2014. ISSN 1553-877X.

[38] GARCÍA-TEODORO, P. et al. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers and Security*, Elsevier Advanced Technology, v. 28, n. 1-2, p. 18–28, feb 2009. ISSN 01674048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404808000692>.

[39] DENEUBOURGL, J. L. et al. the Dynamics of Collective Sorting Robot . Like Ants and Ant . Likb Robots. n. January, 1990.

[40] FARNAAZ, N.; JABBAR, M. A. Random Forest Modeling for Network Intrusion Detection System. *Procedia Computer Science*, The Author(s), v. 89, p. 213–217, 2016. ISSN 18770509. Disponível em: <http://dx.doi.org/10.1016/j.procs.2016.06.047>.

[41] GNANAPRASANAMBIKAI, L.; MUNNUSAMY, N. Data Preprocessing and Classification for Traffic Anomaly Intrusion Detection using NSLKDD Dataset. *International Journal of Pure and Applied Mathematics*, v. 119, n. 10, p. 847–858, 2018. ISSN 1314-4081.

[42] FERNANDES, G.; RODRIGUES, J. J.; PROENÇA, M. L. Autonomous profile-based anomaly detection system using principal component analysis and flow analysis. *Applied Soft Computing Journal*, v. 34, p. 513–525, 2015. ISSN 15684946.

[43] FERNANDES, G. et al. Network anomaly detection using IP flows with Principal Component Analysis and Ant Colony Optimization. *Journal of Network and Computer Applications*, Elsevier, v. 64, p. 1–11, 2016. ISSN 10958592. Disponível em: <http://dx.doi.org/10.1016/j.jnca.2015.11.024>.

[44] CARVALHO, L. F. et al. A novel anomaly detection system to assist network management in sdn environment. In: *2017 IEEE International Conference on Communications (ICC)*. [S.l.: s.n.], 2017. p. 1–6. ISSN 1938-1883.

[45] FERNANDES, G. et al. Statistical, forecasting and metaheuristic techniques for network anomaly detection. *Proceedings of the 30th Annual ACM Symposium on Applied Computing - SAC '15*, p. 701–707, 2015. Disponível em: <http://dl.acm.org/citation.cfm?doid=2695664.2695852>.

[46] TIMMIS, J.; NEAL, M.; HUNT, J. An artificial immune system for data analysis. *Bio Systems*, v. 55, n. 1-3, p. 143–150, 2000. ISSN 0303-2647.

[47] FORREST, S. et al. A sense of self for Unix processes. *Proceedings 1996 IEEE Symposium on Security and Privacy*, p. 120–128, 1996. ISSN 1081-6011. Disponível em: <http://ieeexplore.ieee.org/document/502675/>.

[48] J. Doyne Farmer, Norman H Packard, A. S. The Immune System, Adaptation, and Machine Learning. p. 187–204, 1986.

[49] AYARA, M. et al. Negative selection: How to generate detectors. *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, v. 1, p. 89–98, 2002.

[50] DASGUPTA, D.; FORREST, S. Tool Breakage Detection in Milling Operations using a 1 Introduction 2 Negative Selection Algorithm. p. 1–17, 1995.

[51] GONZÁLEZ, F.; DASGUPTA, D.; GÓMEZ, J. The Effect of Binary Matching Rules in Negative Selection. In: . [s.n.], 2003. p. 195–206. ISBN 978-3-540-40602-0. Disponível em: <http://link.springer.com/10.1007/3-540-45105-6{_}.>

[52] ESPONDA, F.; FORREST, S.; HELMAN, P. Detection Schemes. v. 34, n. 1, p. 357–373, 2004.

[53] JI, Z.; DASGUPTA, D. Real-Valued Negative Selection Algorithm with Variable-Sized Detectors. p. 287–298, 2004.

[54] MARTINEZ, A. S. An Efficient Algorithm to Generate Random Uncorrelated Euclidean Distances: The Random Link Model. *Brazilian Journal of Physics*, v. 36, n. 1B, p. 232–236, 2006. ISSN 01039733.

[55] DU, H.-f.; JIAO, L.-c. Clonal operator and antibody clone algorithms. n. November, p. 4–5, 2002.

[56] MATZINGER, P. The evolution of the danger theory. *Expert Review of Clinical Immunology*, v. 8, n. 4, p. 311–317, 2012. ISSN 1744666X.

[57] HOSSEINPOUR, F. et al. Distributed Agent Based Model for Intrusion Detection System Based on Artificial Immune System. *International Journal of DIgital Content Technology and Its Applications (JDCTA)*, v. 7, n. 9, p. 206–214, 2013.

[58] AICKELIN, U. et al. Danger Theory: The Link between AIS and IDS? *Ssrn*, p. 147–155, 2016. ISSN 03029743.

[59] OU, C. M.; OU, C. R. Multi-agent artificial immune systems (MAAIS) for intrusion detection: Abstraction from danger theory. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. [S.l.: s.n.], 2009. v. 5559 LNAI, p. 11–19. ISBN 3642016642. ISSN 03029743.

[60] OWAIS, S. et al. Survey: Using Genetic Algorithm Approach in Intrusion Detection Systems Techniques. *2008 7th Computer Information Systems and Industrial Management Applications*, p. 300–307, 2008. Disponível em: <http://ieeexplore.ieee.org/document/4557881/>.

[61] GONZÁLEZ, F.; DASGUPTA, D. An immunogenetic technique to detect anomalies in network traffic. *GECCO'02 Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, p. 1081–1088, 2002.

[62] DASGUPTA, D.; GONZÁLEZ, F. An immunity-based technique to characterize intrusions in computer networks. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 3, p. 281–291, 2002. ISSN 1089778X.

[63] AZIZ, A.; AZAR, A. Genetic algorithm with different feature selection techniques for anomaly detectors generation. *Computer Science ...*, p. 769–774, 2013. Disponível em: <http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=6644>.

[64] DASGUPTA, D.; YU, S.; MAJUMDAR, N. Mila - multilevel immune learning algorithm and its application to anomaly detection. *Soft Computing*, v. 9, p. 172–184, 03 2005.

[65] FU, H.; YUAN, X.; WANG, N. Multi-agents artificial immune system (maais) inspired by danger theory for anomaly detection. In: *2007 International Conference on Computational Intelligence and Security Workshops (CISW 2007)*. [S.l.: s.n.], 2007. p. 570–573.

[66] PROENÇA, M. L. et al. The Hurst Parameter for Digital Signature of Network Segment. *Telecommunications and Networking - ICT 2004: 11th International Conference on Telecommunications, Fortaleza, Brazil, August 1-6, 2004. Proceedings*, p. 772–781, 2004. ISSN 03029743 16113349. Disponível

em: <http://dx.doi.org/10.1007/978-3-540-27824-5{_}103{\%}5Cnhttp://link.springer.com/10.1007/978-3-540-27824>.

[67] ADANIYA, M. H.; ABRÃO, T.; PROENÇA, M. L. Anomaly detection using metaheuristic firefly harmonic clustering. *Journal of Networks*, v. 8, n. 1, p. 82–91, 2013. ISSN 17962056.

[68] PENA, E. H. et al. Anomaly detection using the correlational paraconsistent machine with digital signatures of network segment. *Information Sciences*, Elsevier Inc., v. 420, p. 313–328, 2017. ISSN 00200255. Disponível em: <http://dx.doi.org/10.1016/j.ins.2017.08.074>.

[69] PENA, E. H. M. et al. Correlational paraconsistent machine for anomaly detection. In: *2014 IEEE Global Communications Conference*. [S.l.: s.n.], 2014. p. 551–556. ISSN 1930-529X.

[70] MENDES, L. D. S. Anomaly Detection Using Digital Signature of Network Segment Aiming to Help Network Management. v. 23, n. 1, p. 1–11, 2008.

[71] PROEM, M. L. et al. a Practical Approach for Automatic Generation of Network Segment Traffic Baselines. 2005.

[72] CARVALHO, L. F. et al. Digital signature of network segment for healthcare environments support. *IRBM*, Elsevier Masson SAS, v. 35, n. 6, p. 299–309, 2014. ISSN 18760988. Disponível em: <http://dx.doi.org/10.1016/j.irbm.2014.09.001>.

[73] De Assis, M. V.; RODRIGUES, J. J.; PROENÇA, M. L. A seven-dimensional flow analysis to help autonomous network management. *Information Sciences*, Elsevier Inc., v. 278, p. 900–913, 2014. ISSN 00200255. Disponível em: <http://dx.doi.org/10.1016/j.ins.2014.03.102>.

[74] ASSIS, M. V. O. de; RODRIGUES, J. J. P. C.; PROENçA, M. L. A novel anomaly detection system based on seven-dimensional flow analysis. In: *2013 IEEE Global Communications Conference (GLOBECOM)*. [S.l.: s.n.], 2013. p. 735–740. ISSN 1930-529X.

[75] ROSSUM, G. *Python Reference Manual*. Amsterdam, The Netherlands, The Netherlands, 1995.

[76] PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.