Aplicação e Análise Comparativa do Desempenho de Classificadores de Padrões para o Sistema de Detecção de Intrusão Snort

Luan N. Utimura[1], **Kelton A. Costa**[1]

[1]Departamento de Computação
Faculdade de Ciências
Universidade Estadual Paulista "Júlio de Mesquita Filho"

luan.utimura@gmail.com, kelton.costa@gmail.com

Abstract. *With the evolution of the Internet, the volume of data that travels in computer networks grows day by day, challenging the security of computer systems around the world. Among the main tools used to ensure these systems, stand out the Intrusion Detection Systems (IDS). In an environment where new vulnerabilities are discovered every week [Symantec 2015], anomaly-based detections can reduce damage from unknown attacks. Thus, this work proposes the application of two pattern classifiers to replace the default detection scheme of the open-source Snort IDS. The comparative study focused on the performance of both techniques presented high accuracy rates in the connection classifications.*

Resumo. *Com a evolução da Internet, o volume de dados que trafegam nas redes de computadores cresce dia após dia, desafiando a segurança de sistemas computacionais do mundo todo. Dentre as principais ferramentas utilizadas para assegurar esses sistemas, destacam-se os Sistemas de Detecção de Intrusão (SDI). Em um ambiente onde novas vulnerabilidades são descobertas toda semana [Symantec 2015], detecções por anomalia podem reduzir os danos de ataques desconhecidos. Desta forma, este trabalho propôs a aplicação de dois classificadores de padrões para substituir o esquema de detecção do SDI código-aberto Snort. O estudo comparativo focado no desempenho de ambas as técnicas apresentou altas taxas de acurácias nas classificações de conexões.*

1. Introdução

A Internet, considerada uma das maiores invenções da humanidade, atribuiu diversos valores intrínsecos ao modo como os homens vivem, comunicam e interagem na sociedade. A difusão da informação multimídia foi, e ainda é, imprescindível para o crescimento, manutenção e funcionamento desse grande sistema, e por esse motivo, garantir a sua integridade e segurança é uma tarefa importante para profissionais da área de Segurança de Redes de Computadores.

A rede de computadores é definida por [Tanenbaum 2003] como "um conjunto de computadores autônomos interconectados por uma única tecnologia. Dois computadores estão interconectados quando podem trocar informações". Essa troca implica necessariamente em uma comunicação, que sob a perspectiva de [Kurose and Ross 2010], deve

abranger quatro pontos para ser considerada segura: confidencialidade, autenticação do ponto final, integridade de mensagem e segurança operacional.

Existem diversos mecanismos que podem ser utilizados para garantir a segurança de uma rede, dentre eles, pode-se destacar o uso de *firewalls*, antivírus, protocolos de segurança, criptografias, credenciais e sistemas de detecção e prevenção de intrusão.

No escopo dos Sistemas de Detecção de Intrusão (SDI), segundo [Planquart 2001], existem três categorias básicas quanto ao tipo de detecção: baseada em *host*, rede ou avaliação de vulnerabilidade.

Dependendo do tipo de análise utilizada, os sistemas podem ser classificados como baseados em assinatura ou anomalia [Kumar and Punia 2013]. Esquemas baseados em assinatura buscam padrões pré-definidos, também chamados de assinaturas, que poderiam representar uma atividade maliciosa. Já os esquemas baseados em anomalia buscam traçar (ou estimar) o que acredita-se ser o comportamento normal da rede. Assim, quando o sistema demonstra comportamentos que ultrapassam os limites vistos como normais, entende-se que existe uma anormalidade [Garcia-Teodoro et al. 2009].

Sistemas que baseiam-se apenas em detecção por assinatura passam por dificuldades ao tentar detectar novas ameaças, isto é, aquelas que possuem assinaturas desconhecidas. Em contrapartida, sistemas de detecção por anomalia conseguem detectar essas ameaças com maior facilidade, entretanto, deve-se atentar na ocorrência de falsos positivos, que podem ocorrer com maior frequência dependendo do quão bem o sistema está calibrado e sensível à anomalias.

Nos últimos anos, diversos estudos foram realizados para integrar técnicas de Inteligência Artificial (IA) aos SDIs baseados em anomalia. Neste trabalho, o enfoque é voltado para a extensibilidade de Sistemas de Detecção de Intrusão baseados em Rede (SDIR) nos quais a detecção é feita, a princípio, por assinatura, de modo a fornecer a capacidade de detecção por anomalia por meio do emprego de algoritmos de classificação de padrões, como Redes Neurais Artificiais (RNA) e Floresta de Caminhos Ótimos (OPF). Para ambas as técnicas, na etapa de treinamento, o sistema é exposto a diversos ataques conhecidos para que possa aprender seus padrões e classificá-los. Com isso, a etapa seguinte de testes possibilitará que muitos dos novos ataques sejam classificados com base em experiências aprendidas anteriormente pelo sistema.

Na próxima seção é apresentado um breve resumo sobre os trabalhos relacionados. Na Seção 3 e 4, os classificadores de padrões abordados são descritos em maiores detalhes. Na Seção 5, é discutido o método de pesquisa, a base de dados utilizada e o desenvolvimento do *plug-in* do Snort++. Na Seção 6 e 7 são apresentados os experimentos e resultados. Na Seção 8 são apresentadas as considerações finais do trabalho.

2. Trabalhos Relacionados

O Snort, uma das principais ferramentas tomadas como referência no escopo dos SDIs, já foi alvo de diversos estudos que visaram a sua extensibilidade por meio do desenvolvimento de pré-processadores.

Em [Biles 2003], um *plug-in* conhecido por SPADE (*Statistical Packet Anomaly Detection Engine*) foi desenvolvido com o intuito de detectar intrusões a partir de análises estatísticas sobre o comportamento do sistema. Para detectar comportamentos anormais,

o SPADE mantém internamente uma tabela de probabilidades que contém informações a respeito do número de ocorrências de diversos tipos de pacotes ao longo do tempo na rede. A partir dessas probabilidades, um "escore" é calculado, permitindo a classificação desses comportamentos como "normais" ou "anômalos".

Uma outra linha de trabalho, como mostrada em [Devi and Nagpal 2012], explora o aperfeiçoamento das taxas de detecções realizadas pela ferramenta com a utilização de Algoritmos Genéticos. Com a implementação da técnica, o sistema de detecção proposto é capaz de gerar e atualizar as regras da ferramenta em tempo real, fazendo-se o uso do algoritmo genético para reduzir as taxas de falsos positivos ao longo do tempo. Posteriormente, um trabalho semelhante foi desenvolvido em [Kumar and Punia 2013].

Tal aperfeiçoamento também tem sido almejado com o uso de RNAs, como mostrado em [da Silva 2007], onde características de pacotes individuais eram processadas com o objetivo de detectar intrusos. De modo similar, em [Jaggim and Sangade 2014], o Snort é aliado ao Perceptron Multicamadas (*Multilayer Perceptron* - MLP) para que ataques que utilizam o protocolo ICMP sejam não apenas detectados mas também classificados em seis diferentes categorias: *Smurf*, *Teardrop*, *Satan*, *Guest*, *Warezclient* e *Buffer Overflow*.

Diferente dos trabalhos apresentados anteriormente, este trabalho propôs, experimentalmente, a substituição completa do esquema de detecção por assinatura do Snort++ (ou Snort 3), a última versão lançada da ferramenta. No lugar da detecção convencional, são utilizadas duas técnicas de classificação de padrões: a MLP e o classificador baseado em Floresta de Caminhos Ótimos (*Optimum-Path Forest* - OPF). Paralelamente à implementação do novo esquema de detecção, um estudo comparativo foi realizado com base no desempenho observado de ambas as técnicas sobre a base de dados ISCX IDS 2012.

3. Perceptron Multicamadas (MLP)

Pelo fato de Perceptrons possuírem apenas uma camada, existem restrições que impedem o aprendizado de importantes mapeamentos. Tal limitação foi demonstrada em [Minsky and Papert 1969], com o clássico problema do *XOR*, onde foi observado não ser possível garantir a separabilidade linear das instâncias com uma Rede Neural de Camada Única.

A essência da restrição encontra-se na ausência de representações internas em Redes Neurais de Camada Única [de Leon F de Carvalho 2009]. Por outro lado, RNAs que possuem uma ou mais camadas ocultas contornam esse problema com o algoritmo de retropropagação (*backpropagation*).

A primeira etapa do algoritmo consiste em apresentar um padrão de entrada à RNA, propagando pulsos camada por camada, até obter o resultado final. Na segunda etapa, o resultado obtido é comparado com o esperado e, no caso de divergências, um erro é calculado. Com isso, o erro é propagado de volta para a camada de entrada, sendo que, nas camadas intermediárias, todas as conexões são reajustadas de acordo com o erro observado.

4. Floresta de Caminhos Ótimos (OPF)

Criado por [Papa et al. 2009], o classificador baseado em Floresta de Caminhos Ótimos (OPF) tem por objetivo a eficiência na etapa de treinamento e a eficácia na etapa de testes, de modo a reunir, em sua abordagem multiclasse, rapidez e simplicidade.

O OPF abrange as três abordagens de aprendizado: supervisionado, não-supervisionado e reforçado (semi-supervisionado), sendo que, no escopo do aprendizado supervisionado, existem três casos típicos de classificação binária em espaços de características bidimensionais: (a) linearmente separáveis; (b) linearmente separáveis por partes; e (c) classes não separáveis com formas arbitrárias [Papa and Falcão 2010]. A técnica em si baseia-se em um grafo, que dependendo da relação de adjacência escolhida, pode ser completo ou k-NN, diferindo-se quanto à relação de adjacência, função de custo de caminho e metodologia de estimação de protótipos utilizada. Neste trabalho, é utilizada a abordagem de aprendizado supervisionado com grafo completo.

A técnica em si tem por objetivo segmentar o espaço de características, isto é, agrupar amostras de acordo com suas classes. Portanto, em classificadores OPF com grafos completos, os vetores de características das amostras são representados por nós de um grafo, onde todos eles estão conectados entre si por meio de arestas (Figura 2.a). Para iniciar o processo de segmentação, protótipos[1] são escolhidos para competir entre si, conquistando amostra por amostra. A escolha dos protótipos se dá através de Árvores Geradoras Mínimas (*Minimum Spanning Tree - MST*), com o intuito de selecionar aqueles elementos que se encontram nas fronteiras das classes (Figura 1).

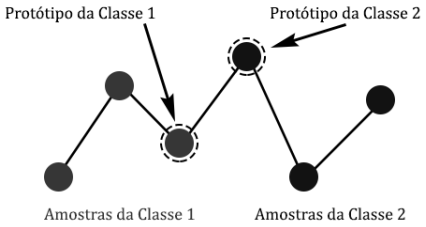

Figura 1. Escolha de Protótipos

Na prática, cada amostra oferece um possível caminho com custo não-negativo para o protótipo. A escolha leva em consideração o caminho de menor custo, de acordo com a função de custo de caminho utilizada. No final do processo, observam-se diversas Árvores de Caminhos Ótimos (*Optimum-Path Trees - OPT*) (Figura 2.b). A etapa de teste visa validar o treinamento do classificador por meio do cálculo de acurácia das classificações realizadas sobre um conjunto de teste. Nesta etapa, uma amostra desconhecida é apresentada aos protótipos do classificador (Figura 2.c) que, por sua vez, oferecem caminhos de diversos custos com o intuito de conquistá-la. O caminho que tiver menor custo determina a classificação da amostra (Figura 2.d).

5. Método de Pesquisa

Conforme mencionado anteriormente, neste trabalho optou-se pela utilização de duas técnicas de classificação de padrões baseadas, respectivamente, em RNA e OPF. A seguir,

[1] Protótipos correspondem às amostras que melhor representam suas respectivas classes.

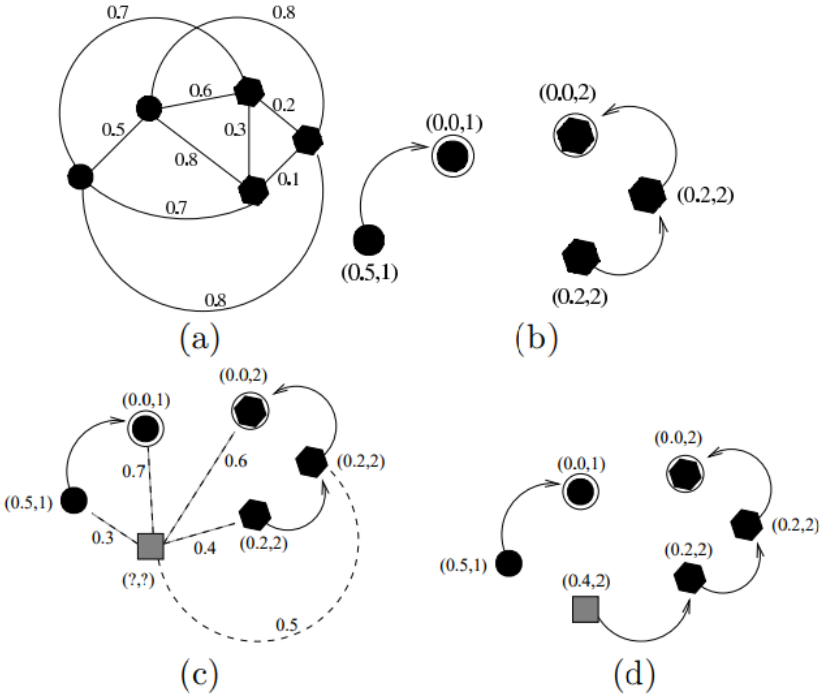

Figura 2. Etapas do OPF

são apresentadas as qualidades de cada uma das técnicas que, eventualmente, levaram às suas escolhas.

No caso das RNAs, elas têm sido um dos principais instrumentos de estudo na área da Inteligência Artificial (IA), tendo em vista a diversidade de problemas passíveis de resolução com essa técnica e, também, dos resultados obtidos que demonstraram-se promissores. Neste trabalho, a RNA utilizada foi o Perceptron Multicamadas (MLP).

Em pesquisas voltadas para a implementação de SDIs, a detecção de anomalias é comumente feita por meio de MLPs. Estudos recentes como o de [Amato et al. 2017] mostram que a técnica ainda é uma das principais referências na classificação de padrões e, sob o ponto de vista de trabalhos correlatos, isso deve-se ao fato de MLPs possuírem habilidade de se adaptar a mudanças, resiliência a informações com ruído e tolerância a falhas [Van Efferen and Ali-Eldin 2017].

Já o OPF é uma técnica que busca conciliar acurácia e velocidade no processo de classificação, destacando-se também pela economia de processamento quando em execução. A técnica, apesar de recente, demonstrou-se poderosa ao comparar-se, de igual para igual, com outras técnicas clássicas de classificação de padrões. Alguns estudos já foram realizados com o OPF no contexto de detecção de anomalias, em bases de dados conhecidas como a IDS Bag, KDDCup e NSL-KDD. Na Tabela 1, observa-se os resultados atingidos por [Pereira 2012] com o OPF sobre a base de dados NSL-KDD.

Portanto, a aplicação da MLP e do OPF tem por objetivo complementar as capacidades de um SDI popularmente conhecido, ao mesmo tempo em que, com a utilização de uma base de dados, a performance de ambas as técnicas no processo de classificação de anomalias é comparada com o propósito de instigar o desenvolvimento de trabalhos

Tabela 1. Resultados de classificação de padrões na base de dados NSL-KDD

Classificador	Acurácia	Treinamento [s]	Teste [s]
OPF	99.50 ± 0.017	125.8236	425.9925
Bayes	99.50 ± 0.021	33.9259	219.1123
SVM-RBF	89.56 ± 3.91	5018.6606	1.9853
SOM	99.38 ± 0.025	9650.4257	23.2324

futuros acerca deste tema.

5.1. Base de Dados ISCX IDS 2012

É comum, em pesquisas sobre SDIs, a utilização de bases de dados conhecidas para realizar análises comparativas (*benchmarking*) entre técnicas de detecção de intrusão. Entretanto, com o passar dos anos, muitos dos ataques contidos nessas bases não refletem mais a realidade das redes de computadores atuais, tornando-se discutível os resultados alcançados por novas técnicas que, de alguma forma, basearam-se em uma infraestrutura de rede ultrapassada.

A base de dados ISCX IDS 2012 [Canadian Institute for Cybersecurity 2012] conta com uma série de características que a torna ideal para a avaliação de SDIs. Uma característica importante é a sua tentativa de replicar tanto a arquitetura como o tráfego de uma rede real, evitando quaisquer ajustes posteriores à captura que possam, eventualmente, torná-la excessivamente sintética. Além disso, por ser rotulada, classificadores podem aprender os padrões de atividades normais e anômalas da rede, sem a necessidade de rotular milhares de amostras previamente.

Na base de dados também encontram-se arquivos com a captura completa do tráfego de rede que deu origem à base de dados rotulada. Com eles, é possível reproduzir as conexões normais e anômalas, inclusive com os *payloads* dos pacotes, dando maior liberdade ao pesquisador para avaliar e comparar seus sistemas. A base de dados é composta por sete dias de atividade de rede, onde diversos cenários de intrusão foram simulados.

Em relação aos dados que compõem a base de dados, existem no total 2.071.657 conexões, sendo 96.67% rotuladas como "normais" e 3.32% como "ataques". Por conta do hardware utilizado e também do cronograma de desenvolvimento do projeto, foram utilizados, aproximadamente, 10% da ISCX IDS 2012, resultando em 207.172 conexões (96.68% rotuladas como "normais" e 3.31% rotuladas como "ataques"). Na Seção 6, as proporções utilizadas para treinamento e teste são apresentadas.

5.2. *Plug-in* do Snort++

O Snort++, por natureza, baseia-se em um esquema de detecção voltado para a análise de assinaturas e regras. Assim, para que a ferramenta passasse a operar com um esquema de detecção por anomalias, com técnicas de classificação de padrões, foi preciso desenvolver um *plug-in* para estender as capacidades do programa. A extensibilidade do Snort++, de modo geral, se dá por meio de diversos tipos de *plug-ins*, que atuam em diferentes partes do processamento de pacotes.

Tendo em vista que o treinamento das técnicas de classificação de padrões ocorreu por meio das características de conexões da base de dados ISCX IDS 2012, o *plug-in* escolhido não deveria atuar apenas ao nível de pacotes individuais, e sim de conexões, uma vez que as características de conexões são passadas, por meio de vetores, às técnicas de classificação de padrões.

Portanto, utilizando como referência a metodologia de [Salem and Buehler 2013] para transformar fluxos de dados em vetores de características de conexões, para facilitar a aplicação de técnicas inteligentes em SDIs, foi desenvolvido um *Inspector* para o Snort++ que, paralelamente, desempenha duas funções: gerenciamento e classificação de conexões. A escolha do *plug-in* tipo *Inspector* deve-se ao fato de que, dentre todos os outros tipos de *plug-ins*, o *Inspector* encontra-se em um patamar intermediário entre baixos níveis de abstrações (codificação/decodificação de pacotes) e altos níveis de abstrações (manipulação e configuração de regras), e por lidar diretamente com as características de um pacote, é adequado para gerenciar características de conexões (múltiplos pacotes).

5.2.1. Gerenciamento de Conexões

O gerenciamento de conexões torna-se uma etapa primordial no desenvolvimento deste projeto, uma vez que é preciso manter controle de todas as conexões que estão sendo criadas, atualizadas e finalizadas. Nesse contexto, uma conexão é vista como sendo uma janela de tempo em que trocas de pacotes ocorrem entre dois *hosts*. Sendo assim, o objetivo do *Inspector* é extrair características desse agrupamento de pacotes e repassá-las para as técnicas de classificação, categorizando-as como normais ou anômalas.

Uma vez que conexões são criadas e atualizadas pelo gerenciador de conexões, é preciso definir uma política que, em meio a tantas conexões, escolha algumas para o processo de classificação. Naturalmente, para protocolos não orientados à conexão, como UDP e ICMP, uma solução seria definir um tempo limite de ociosidade para a conexão. Assim, quando esse limite é ultrapassado, considera-se que a conexão ficou inativa e portanto está apta a ser despachada. Já para o protocolo de estados TCP, uma primeira alternativa seria aguardar a conexão atingir o estado *Closed*. Todavia, em ambientes reais de redes de computadores, não é garantido que toda conexão TCP vá alcançar esse estado, seja por eventos inesperados ou pelo simples prolongamento da conexão. Portanto, a abordagem de *timeouts* utilizadas para protocolos UDP e ICMP também deve abranger o protocolo TCP, atribuindo a cada estado um tempo limite adequado. A política de *timeout* utilizada neste trabalho, proposta por [Salem and Buehler 2013], é mostrada na Tabela 2.

Tabela 2. Política de *timeout* para conexões UDP, ICMP e TCP

Protocolo - Estado da Conexão	***Timeout* [s]**
UDP - N/A	180
ICMP - N/A	180
TCP - *Handshake*	20
TCP - *Established*	720
TCP - *Termination*	675
TCP - *Closed*	240

A verificação de *timeouts* das conexões é feita por uma *thread* que, periodicamente, checa a lista de conexões ativas no *Inspector*. Cada conexão possui um campo chamado *timestamp* (marcação de horário), utilizado para registrar o tempo em que o último pacote da conexão foi adicionado. Dessa forma, a *thread*, que também possui um campo *timestamp* com o tempo atual de execução, percorre sequencialmente a lista de conexões, despachando todas aquelas cuja diferença entre os *timestamps* seja maior que os valores pré-definidos na Tabela 2.

5.2.2. Classificação de Conexões

Com a política preemptiva de *timeouts* de conexões, evita-se a sobrecarga do sistema com múltiplas conexões ativas na memória e, simultaneamente, cria-se um espaço para a classificação de conexões. Toda conexão despachada pelo sistema é, obrigatoriamente, classificada pela MLP e o OPF, treinados previamente com a base de dados ISCX IDS 2012. Para a etapa de classificação, as conexões despachadas têm suas características (Tabela 3) salvas na forma de um vetor que, na prática, possui 21 características (devido a adaptação de valores alfanuméricos para valores numéricos), servindo de entrada para as técnicas utilizadas neste projeto.

Tabela 3. Características de conexões

#	Característica	Descrição
1	*totalSourceBytes*	Total de *bytes* enviados pela origem
2	*totalDestinationBytes*	Total de *bytes* enviados pelo destino
3	*totalDestinationPackets*	Total de pacotes enviados pelo destino
4	*totalSourcePackets*	Total de pacotes enviados pela origem
5	*direction*	Direção da conexão
6	*sourceTCPFlagsDescription*	Descrição de flags TCP da origem
7	*destinationTCPFlagsDescription*	Descrição de flags TCP do destino
8	*protocolName*	Nome do protocolo
9	*sourcePort*	Porta da origem
10	*destinationPort*	Porta do destino
11	*duration*	Duração da conexão em segundos

Todo o funcionamento do *Inspector* no que diz respeito à criação, atualização e finalização de conexões é impresso na tela, para facilitar a visualização e compreensão das informações que representam o comportamento do sistema. Assim, foram definidos alertas padronizados que indicam a ocorrência desses três eventos:

1. **Criação de Conexão**
   [+] `TCP-192.168.0.102:5050-192.168.0.105:80`
2. **Atualização de Conexão**
   [*U*] `UDP-192.168.0.109:2500-192.168.0.104:2401`
3. **Finalização de Conexão**
   [−] `ICMP-192.168.0.102:0-192.168.0.109:0-61562`

Sendo que, na finalização de conexões, o resultado das classificações são impressos logo abaixo, com o formato:

```
[OPF] Result:  Normal
[MLP] Result:  Normal
```

6. Experimentos

No processo de experimentação do projeto, foi utilizado um notebook Dell Inspiron 14R 5421 com sistema operacional Ubuntu 16.04 LTS (64 bits), processador Intel i7-3537U de 2.00GHz e 8GB de memória RAM DDR3 de 1600MHz. Nesta etapa, os experimentos dividiram-se em duas fases, sendo a primeira voltada para a análise comparativa da performance das técnicas de classificação de padrões, utilizando a base de dados ISCX IDS 2012 e, a segunda, voltada para a validação da extensão do SDI Snort++.

Na primeira fase, tendo em vista o tamanho da base de dados e o hardware disponível, optou-se pela utilização de 10% da ISCX IDS 2012. Desta forma, experimentos foram realizados com diferentes proporções de conjuntos de treinamento e teste, para ambas as técnicas, conforme mostrado na Tabela 4.

Tabela 4. Configuração dos experimentos realizados com a MLP e o OPF.

%	Proporção de Treinamento/Teste (%)
10%	50/50
10%	60/40
10%	70/30
10%	80/20

Para garantir maior consistência nos resultados, cada configuração de experimento foi realizada dez vezes, sendo calculadas as respectivas médias e desvios padrão dos resultados obtidos. Além disso, para a MLP, os dados dos conjuntos de treinamento e teste foram normalizados para o intervalo de [0, 1]. Sem a normalização, os pesos podem alcançar valores altos, saturando as funções de transferência da RNA.

Por outro lado, o processo de normalização não é necessário para o OPF, pelo fato dos pesos dos arcos serem calculados com o log da distância euclidiana das amostras. Assim, se as bases fossem normalizadas, as distâncias entre as amostras seria menor ainda, aumentando o processo de competição entre elas e, por fim, dificultaria o processo como um todo para o OPF.

Tanto para a MLP, como para o OPF, os conjuntos de treinamentos tiveram 20% de suas amostras reservadas para os conjuntos de validação, com o intuito de melhorar a acurácia das técnicas. No caso da MLP, os erros de validação obtidos na fase de treinamento derivam-se do método *trainUntilConvergence* da biblioteca PyBrain que, como o próprio nome já diz, treina o módulo até a sua convergência, retornando-o com os parâmetros que oferecem o erro de validação mínimo. A configuração da rede neural e de seus respectivos treinamentos levaram em consideração a definição de alguns parâmetros. A quantidade de neurônios nas camadas de entrada e saída, por exemplo, foram definidas com base na quantidade de características e classes presentes na base de dados ISCX IDS 2012, enquanto que os demais parâmetros foram definidos arbitrariamente, sempre optando pela combinação que produzia melhores resultados.

1. Neurônios na camada de entrada: 21.
2. Neurônios na camada de saída: 1.
3. Quantidade de camadas ocultas: 1.
4. Neurônios na camada oculta: 11.
5. Taxa de aprendizado: 0.01.
6. Taxa de *Momentum*: 0.99.
7. Função de ativação: Sigmoidal.
8. Quantidade máxima de *Epochs* (Épocas): 25.
9. *continueEpochs*: 10.
10. Decaimento de pesos: 0.00.
11. Proporção para validação: 0.2.

No caso do OPF, a etapa de treinamento foi realizada com o programa `opf_learn` da biblioteca LibOPF, que executa o procedimento de aprendizado a partir de erros de classificação no conjunto de validação. Já o cálculo da acurácia, de ambas as técnicas, foi realizada com o programa `opf_accuracy`, aplicável a qualquer técnica de classificação em que se deseja comparar com o OPF. A acurácia é medida levando em consideração que as classes podem ter diferentes tamanhos. Assim, se um classificador sempre atribui as amostras à classe de maior tamanho, sua acurácia cai drasticamente devido à grande taxa de erros em classes menores [Papa and Falcão 2009].

Na segunda fase, os experimentos realizados focaram-se na validação do funcionamento dos modos de operação *online* e *offline* do *Inspector*. Para a configuração dos testes no modo *online*, foram realizadas simples capturas de pacotes na rede local, sem a reprodução dos ataques contidos na base de dados ISCX IDS 2012, devido à impossibilidade de reproduzí-los com o hardware disponível. Todavia, apesar de não reproduzir os ataques com o hardware utilizado no projeto, diversos arquivos de captura de tráfego da base ISCX IDS 2012 foram lidos no modo de operação *offline*, contendo alguns dos ataques da base de dados. A seguir, são apresentados os resultados técnicos obtidos em ambas as fases de experimentação do projeto.

7. Resultados

Seguindo o modelo de configuração de experimentos da Tabela 4, os resultados obtidos com as técnicas de classificação de padrões são mostrados na Tabela 5, por meio de uma relação que associa o tempo de treinamento, teste e acurácia alcançada.

Os resultados apresentados mostram que o OPF, independente da configuração de experimentos utilizada, é capaz de obter maior consistência nos resultados, com baixos desvios padrão. A MLP, por outro lado, dentro do limite de 25 épocas, mostrou-se suscetível a alcançar diversas acurácias, indicando que a convergência pode ser facilmente influenciada pela matriz de pesos inicializada com valores aleatórios.

Outro fator que exige atenção é o tempo médio de treinamento e teste das técnicas. A MLP, por estar limitada a um número fixo de épocas, teve, de forma significativa, tempos médios de treinamento e teste mais curtos que o OPF. Tal restrição pode ter limitado a eficácia da técnica que, popularmente, é conhecida por obter bons resultados em detrimento de processamentos de dados exaustivos. É reconhecido, portanto, que melhores resultados poderiam ser obtidos com computadores mais robustos ou plataformas distribuídas.

Tabela 5. Resultados obtidos com o OPF e MLP em 10% da base ISCX IDS 2012

Técnica	Proporção Treinamento/Teste (%)	Tempo Treinamento (s)	Tempo Teste (s)	Acurácia (%)
OPF	50/50	1422.93±151.68	318.80±4.86	96.80±0.04
OPF	60/40	2874.97±808.81	314.97±7.60	96.83±0.17
OPF	70/30	3165.79±249.85	263.96±4.31	97.32±0.10
OPF	80/20	4644.67±1189.10	210.50±5.22	96.59±0.14
MLP	50/50	1146.75±15.07	147.67±88.34	90.41±9.74
MLP	60/40	1330.26±8.67	145.14±89.35	89.45±10.06
MLP	70/30	1608.95±20.17	143.58±90.16	87.19±11.27
MLP	80/20	1800.35±34.81	6.02±0.13	95.94±1.81

Para efeito de comparação, as acurácias do OPF e da MLP com a proporção 80/20 de treino/teste é mostrada em detalhes na Tabela 6.

Tabela 6. Acurácia das técnicas com proporção 80/20

# Treinamento	Acurácia OPF (%)	Acurácia MLP (%)
1	96.56	96.95
2	96.24	95.04
3	96.65	97.97
4	96.73	96.97
5	96.50	95.00
6	96.67	97.06
7	96.55	96.47
8	96.67	91.50
9	96.64	96.40
10	96.64	96.03
Média	**96.59**	**95.94**
Desvio Padrão	**0.14**	**1.81**

Foram geradas, ainda, as matrizes de confusão do décimo treinamento de ambas as técnicas, de modo a permitir uma análise focada diretamente nas classificações realizadas sobre as conexões da base de dados, conforme mostrado na Tabela 7 e 8.

As matrizes de confusão indicam que o OPF obteve uma maior taxa de verdadeiros positivos (96.35%), isto é, uma maior quantidade de conexões normais classificadas como "normais", enquanto que a MLP, ligeiramente, obteve uma maior taxa de verdadeiros negativos (3.10%), ou seja, uma maior quantidade de conexões de ataque classificadas como "ataque". É importante ressaltar que, apesar de aparentar ser um ganho benéfico em relação ao OPF, tal resultado vem ao custo de uma maior taxa de falsos negativos, quase 6 vezes maior que o do OPF, como discutido adiante.

Outro aspecto interessante é a taxa de falsos positivos produzidos por cada técnica, que indica a quantidade de conexões normais classificadas como "ataques". Neste quesito, a MLP cometeu um maior número de erros de classificações (1.74%) em relação

Tabela 7. Matriz de confusão do décimo treinamento do OPF com proporção 80/20

	Positivo	Negativo	Total
Positivo	39926 (96.35%)	144 (0.35%)	40070
Negativo	87 (0.21%)	1281 (3.09%)	1368
Total	40013	1425	41438

Tabela 8. Matriz de confusão do décimo treinamento da MLP com proporção 80/20

	Positivo	Negativo	Total
Positivo	39351 (94.96%)	719 (1.74%)	40070
Negativo	84 (0.20%)	1284 (3.10%)	1368
Total	39435	2003	41438

ao OPF (0.35%). Por outro lado, a taxa de falsos negativos, representando a quantidade de conexões de ataques classificadas como "normais", manteve-se próxima em ambas as técnicas (0.21% no OPF e 0.20% na MLP).

Quanto aos resultados obtidos com a execução do Snort++ com o *Inspector* desenvolvido, foi possível observar, no modo de operação *online*, a classificação de conexões em tempo real, na medida em que as mesmas sofriam *timeout*. A Figura 3 ilustra a classificação de uma conexão que foi estabelecida com um servidor da Amazon.

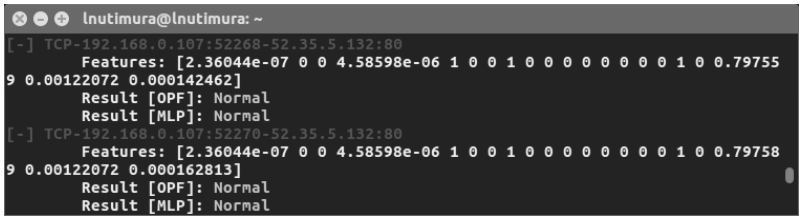

Figura 3. Primeiro Exemplo de Classificação de Conexão

Para validar a classificação de conexões de ataques, no modo de operação *offline*, foram lidos arquivos de captura de tráfego de rede da base de dados ISCX IDS 2012. Por mais difícil que seja encontrar constantes ótimas para o *timeout* das conexões, que produza os melhores resultados para essa base de dados em específico, foi possível observar a convergência das técnicas em determinadas conexões, tal como mostra a Figura 4.

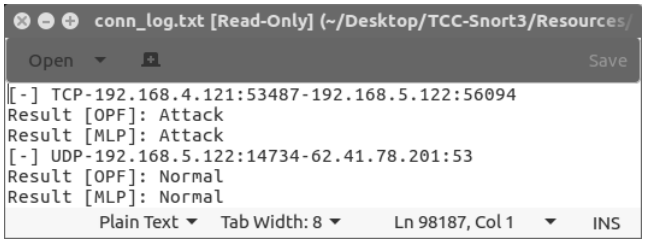

Figura 4. Segundo Exemplo de Classificação de Conexão

Todavia, nem sempre as técnicas convergem para o mesmo resultado de classificação para uma conexão qualquer, como é ilustrado na Figura 5. A frequência de ocorrência de casos como esse é diretamente proporcional às taxas de falsos positivos e falsos negativos observadas nas matrizes de confusão e, portanto, uma possível solução para este problema envolveria a execução de treinamentos mais exaustivos e/ou, na medida do possível, a alteração de parâmetros arbitrários das técnicas, de modo a reduzir ambas as taxas.

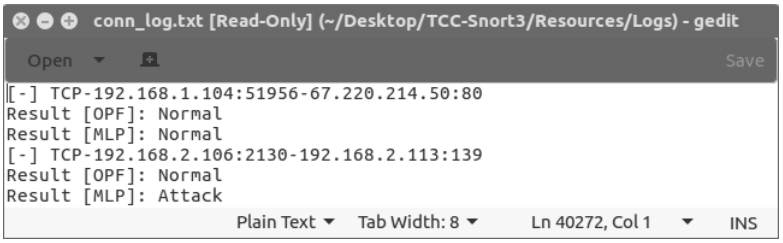

Figura 5. Terceiro Exemplo de Classificação de Conexão

8. Conclusão

Neste artigo apresentou-se um estudo comparativo do desempenho de classificadores de padrões na detecção de intrusões e, ainda, a verificação da aplicabilidade destas técnicas em um Sistema de Detecção de Intrusão existente, o Snort++.

Com a concretização do projeto, ambas as técnicas demonstraram resultados convincentes, tendo em vista as satisfatórias acurácias alcançadas. Também foi possível observar que, apesar de recente, o OPF demonstrou-se majoritariamente mais consistente e resiliente aos erros de classificação, utilizando a base de dados ISCX IDS 2012. Trabalhos dessa natureza fomentam o surgimento de novas pesquisas que, diante de inúmeras possibilidades, podem abordar as técnicas utilizadas em novos ambientes, com outras configurações, contribuindo para o enriquecimento do conhecimento científico acerca da área de Segurança de Redes.

Referências

Amato, F., Mazzocca, N., Moscato, F., and Vivenzio, E. (2017). Multilayer perceptron: An intelligent model for classification and intrusion detection. In *Advanced Information Networking and Applications Workshops (WAINA), 2017 31st International Conference on*, pages 686–691. IEEE.

Biles, S. (2003). Detecting the unknown with snort and the statistical packet anomaly detection engine (spade). *Computer Security Online Ltd., Tech. Rep*.

Canadian Institute for Cybersecurity (2012). Intrusion detection evaluation dataset (iscxids2012).

da Silva, J. R. C. (2007). Redes neurais artificiais para sistemas de detecção de intrusos.

de Leon F de Carvalho, A. P. (2009). Redes neurais artificiais.

Devi, S. and Nagpal, R. (2012). Intrusion detection system using genetic algorithm-a review. *International Journal of Computing & Business Research*.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., and Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1):18–28.

Jaggim, R. and Sangade, J. (2014). Detecting and classifying attacks using artificial neural network. *International Journal on Recent and Innovation Trends in Computing and Communication ISSN*, pages 1136–1142.

Kumar, K. and Punia, R. (2013). Improving the performance of ids using genetic algorithm. *International Journal of Computer Science and Communication*, 4(2).

Kurose, J. F. and Ross, K. W. (2010). *Redes de Computadores e a Internet: uma abordagem top-down*. Pearson.

Minsky, M. and Papert, S. (1969). Perceptrons.

Papa, J. and Falcão, A. (2009). *LibOPF: A library for the design of optimum-path forest classifiers*.

Papa, J. P. and Falcão, A. X. (2010). Optimum-path forest: A novel and powerful framework for supervised graphbased pattern recognition techniques. *Institute of Computing University of Campinas*, pages 41–48.

Papa, J. P., Falcão, A. X., and Suzuki, C. T. (2009). Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19(2):120–131.

Pereira, C. R. (2012). Detecção de intrusão em redes de computadores utilizando floresta de caminhos ótimos.

Planquart, J.-P. (2001). Application of neural networks to intrusion detection. *SANS Institute*.

Salem, M. and Buehler, U. (2013). Reinforcing network security by converting massive data flow to continuous connections for ids. In *Internet Technology and Secured Transactions (ICITST), 2013 8th International Conference for*, pages 570–575. IEEE.

Symantec (2015). A new zero-day vulnerability discovered every week in 2015.

Tanenbaum, A. S. (2003). *Redes de computadoras*. Pearson educación.

Van Efferen, L. and Ali-Eldin, A. M. (2017). A multi-layer perceptron approach for flow-based anomaly detection. In *Networks, Computers and Communications (ISNCC), 2017 International Symposium on*, pages 1–6. IEEE.