Universidade Estadual de Londrina

MARCOS VINICIUS OLIVEIRA DE ASSIS

SISTEMA DE DETECÇÃO DE ANOMALIAS PARA REDES *SDN* UTILIZANDO *DEEP LEARNING* E TEORIA DE JOGOS

Londrina
2020

MARCOS VINICIUS OLIVEIRA DE ASSIS

SISTEMA DE DETECÇÃO DE ANOMALIAS PARA REDES *SDN* UTILIZANDO *DEEP LEARNING* E TEORIA DE JOGOS

Tese orientada pelo Prof. Dr. Mario Lemes Proença Jr. intitulada "Sistema de Detecção de Anomalias para Redes SDN utilizando Deep Learning e Teoria de Jogos" e apresentada ao Programa de Pós-Graduação associado em Engenharia Elétrica da Universidade Estadual de Londrina, como requisito parcial para a obtenção do Título de Doutor em Engenharia Elétrica.

Orientador: Prof. Dr. Mario Lemes Proença Jr.

Londrina
2020

Ficha Catalográfica

Marcos Vinicius Oliveira de Assis
Sistema de Detecção de Anomalias para Redes *SDN* utilizando *Deep Learning* e Teoria de Jogos - Londrina, 2020 - 189 p., 30 cm.
Orientador: Prof. Dr. Mario Lemes Proença Jr.
Redes Definidas por *Software*, Teoria de Jogos, Aprendizagem de Máquina, Aprendizagem Profunda, Detecção de Anomalias
I. Universidade Estadual de Londrina. Doutorado em Engenharia Elétrica. II. Sistema de Detecção de Anomalias para Redes *SDN* utilizando *Deep Learning* e Teoria de Jogos.

MARCOS VINICIUS OLIVEIRA DE ASSIS

SISTEMA DE DETECÇÃO DE ANOMALIAS PARA REDES *SDN* UTILIZANDO *DEEP LEARNING* E TEORIA DE JOGOS

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Estadual de Londrina, como requisito parcial para a obtenção do título de Doutor em Engenharia Elétrica.

BANCA EXAMINADORA

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

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

Prof. Dr. José Valdeni de Lima
Universidade Federal do Rio Grande do Sul - UFRGS

Prof. Dr. Alexandre de Aguiar Amaral
Instituto Federal Catarinense – IFC

Prof. Dr. José Carlos Marinello Filho
Universidade Tecnológica Federal do Paraná – UTFPR

Londrina, 11 de setembro de 2020.

Dedico este trabalho à minha esposa Natalie pelo incentivo, apoio e amor incondicional recebido, e à todos aqueles que buscam o aperfeiçoamento pessoal por meio da busca pelo conhecimento.

Agradecimentos

Agradeço à Deus pela vida e pela possibilidade de conhecer cada uma das pessoas que cito a seguir.

Agradeço à minha esposa Natalie pela parceria de vida, por compartilhar todos os momentos comigo, sejam eles bons ou ruins. Agradeço por acreditar em meu potencial, pelo apoio à minhas decisões, por me incentivar quando me desmotivei, por me abraçar quando chorei, por vibrar comigo em minhas vitórias e por não me deixar desistir frente às minhas derrotas. Agradeço pelo seu exemplo de força, dedicação e amor genuíno, que me inspiram e motivam a sempre melhorar e fazer o que me propuser com todo o coração. Embora palavras não sejam suficientes para descrever minha gratidão, espero poder retribuir ao menos um pouco de tudo que me proporciona. Somos um.

Aos meus filhos caninos Goku e Costelinha, por serem fontes inesgotáveis de amor, energia e bagunça, sempre trazendo meu olhar para o agora. Agradeço por contribuírem com o desenvolvimento deste trabalho suportando a distância física por meses com lealdade e paciência.

Aos meus pais Eunilson e Renicler, bem como minha avó Ingrácia (Bá) e minha irmã Mariane, pelo amor, carinho e preocupação comigo, por sempre priorizarem minha educação e por apoiar e respeitar minhas decisões. O carinho, cuidado, risadas, conversas até tarde, broncas e conselhos foram tão importantes para minha formação quanto qualquer fórmula ou experimento. Gratidão eterna!

Agradeço ao meu orientador Prof. Dr. Mario Lemes Proença Jr., por seus ensinamentos e orientação, por confiar em meu trabalho e em meu potencial e pelas oportunidades oferecidas. Agradeço por investir tempo e energia em minha formação tanto acadêmica quanto pessoal ao longo de 10 anos de trabalho em conjunto. Seus exemplos me motivaram a seguir a carreira acadêmica e passar o que aprendi para meus alunos da melhor forma possível.

Al profesor Dr. Jaime Lloret, por aceptar supervisarme durante el período de 6 meses en su laboratorio como investigador visitante. Su receptividad, orientación y apoyo fueron esenciales para mi desarrollo académico y profesional.

Aos professores Elieser Botelho Manhas Jr., Alexandre de Aguiar Amaral, José Valdeni de Lima, e José Carlos Marinello Filho, por aceitarem fazer parte da banca avaliadora deste trabalho, dedicando tempo e esforço para contribuir com meu desenvolvimento acadêmico e pessoal. A valiosa avaliação de vocês contribuiu para o aumento do rigor científico e da qualidade desta tese.

À minha Mestre Renata Balestrini, que sempre confiou em meu trabalho e potencial, me ensinando o significado de amor genuíno e a importância do *kung fu* (trabalho árduo)

em qualquer área da vida. Agradeço por me ensinar que aperfeiçoamento pessoal é atingido através do coletivo, que ninguém cresce sozinho e que juntos somos mais fortes. O *kung fu* me deu a resiliência e foco necessários para o desenvolvimento deste trabalho, funcionando como minha terapia pessoal. Assim, sou grato pela oportunidade de poder aprender *Hung Gar* sob seus direcionamentos e cuidados.

Aos meus alunos, colegas de trabalho e amigos Vilson, Roberta, Luciana, Teodoro e Mateus, por nos apoiarem num período crítico com todo o coração e carinho. Temos uma gratidão inenarrável com vocês por tudo que nos proporcionaram e auxiliaram, contem conosco sempre!

Agradeço ao meu cunhado Max, por dedicar energia e tempo de suas horas de descanso na correção do inglês em textos redigidos durante o desenvolvimento deste trabalho. Seus ricos comentários contribuíram não somente para os resultados acadêmicos obtidos, mas também para meu crescimento pessoal.

Aos amigos e parceiros de laboratório Luiz, Gilberto, Eduardo, Anderson, Matheus e Cinara, pelo companheirismo e apoio que, mesmo com a distância que muitas vezes nos separa, sempre se mostram dispostos a ajudar, compartilhando conhecimento e dividindo vitórias.

Aos meus amigos Felipe, Guilherme e Bruno, por todo o apoio, incentivo e amizade dedicados a mim durante todo o tempo em que nos conhecemos. Nossa amizade, conversas aleatórias e encontros anuais são muito importantes para mim.

A todos os professores que contribuíram para minha formação não somente durante o Doutorado, mas em todas as etapas de minha vida, por toda sua dedicação, paciência e sabedoria na execução dessa tarefa tão nobre e importante que é a educação. Agradeço por garantirem que o método científico e o pensamento racional prevaleçam, mesmo em tempos obscuros.

Aos meus alunos, tanto da Universidade Federal do Paraná, quanto da Escola de Kung Fu Punhos Unidos, por demonstrarem vontade e determinação de evolução, me motivando a me aperfeiçoar cada vez mais.

À Universidade Estadual de Londrina e todos os seus componentes, os quais possibilitaram minha formação desde o ensino médio (por meio do Colégio Estadual José Aloísio de Aragão - Colégio de Aplicação), até minha Graduação, Mestrado em Ciência da Computação, e agora Doutorado em Engenharia Elétrica. A UEL sempre será minha casa.

À Universidade Federal do Paraná e todos os seus componentes, minha atual casa, por proporcionar um ambiente propício à evolução e aperfeiçoamento de seus integrantes, e por possibilitar minha dedicação exclusiva ao desenvolvimento desta tese no período de um ano e meio, por meio de afastamento para aperfeiçoamento.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pela concessão de bolsa PDSE 2019, possibilitando minha atuação como pesquisador visitante

na Universidade Politécnica de Valência, o que contribuiu significativamente para meu crescimento acadêmico, profissional e pessoal.

"Três coisas agradam a todo o mundo: gentileza, frugalidade e humildade. Pois os gentis podem ser corajosos, os frugais podem ser generosos e os humildes podem ser líderes."
(Lao Tsé - Tao Te King)

ASSIS, Marcos Vinicius Oliveira de. **Sistema de Detecção de Anomalias para Redes SDN utilizando Deep Learning e Teoria de Jogos.** 2020. 189 f. Tese (Doutorado em Engenharia Elétrica) - Universidade Estadual de Londrina, Londrina, 2020.

RESUMO

O tráfego de redes de computadores tem aumentado consideravelmente nos últimos anos devido às constantes inovações em tecnologias de comunicação. Além disso, dispositivos conectados que rodam soluções programadas, baseados em novos paradigmas, tal como Internet das Coisas (IoT, do inglês Internet of Things), incorporam características específicas a este tráfego devido à heterogeneidade dos requisitos de cada aplicação. Assim, o gerenciamento e segurança de toda essa complexa infraestrutura, principalmente das redes de computadores, é essencial. Um emergente modelo de arquitetura de redes, conhecido como Redes Definidas por Software (SDN, do inglês Software Defined Networking), objetiva facilitar este processo através da centralização de todos os dispositivos de rede em um único controlador central, o qual pode ser gerenciado inteiramente via software. Entretanto, essa centralização pode trazer problemas pois, se o controlador SDN for atacado, o funcionamento de toda a rede pode ser prejudicado. Dessa forma, são necessários mecanismos que garantam a segurança deste controlador contra ataques ou anomalias que possam acometer seu funcionamento, tais como ataques distribuídos de negação de serviço (DDoS, do inglês Distributed Denial of Service). Esta tese tem como objetivo apresentar um sistema de detecção e mitigação de anomalias aplicado a ambientes SDN. Para isso, diferentes técnicas de detecção de anomalias são testadas, como redes neurais do tipo perceptron com múltiplas camadas, um modelo de aprendizagem de máquina, além de Redes Neurais Convolucionais (CNN, do inglês Convolutional Neural Networks) e Gated Recurrent Units (GRU), abordagens de aprendizagem profunda. Além disso, é proposta uma abordagem baseada em Teoria de Jogos que tem por finalidade otimizar a quantidade de pacotes descartados em uma política de mitigação. Para mensurar a eficiência do sistema proposto, diferentes cenários de testes são aplicados. Os resultados apontam que o sistema obtém boas taxas de detecção tanto em ataques do tipo DDoS quanto em diferentes ataques de intrusão. Por fim, o sistema se demonstrou eficaz em trazer a rede de volta para seu estado normal por meio do processo de mitigação.

Palavras-Chave: Redes Definidas por Software. Teoria de Jogos. Aprendizagem de Máquina. Aprendizagem Profunda. Detecção de Anomalias

ASSIS, Marcos Vinicius Oliveira de. **Anomaly Detection System for SDN Networks using Deep Learning and Game Theory.** 2020. 189 p. Thesis. (Ph.D. in Electrical Engineering) - Londrina State University, Londrina, 2020.

ABSTRACT

The traffic of computer networks has been considerably increased in past years due to constant innovations in communication technologies. Furthermore, connected devices that run programmed solutions based on new paradigms, such as the Internet of Things (IoT), incorporate specific characteristics to this traffic due to the heterogeneity of each application's requirements. Thus, the management and security of this complex infrastructure, especially of computer networks, is essential. An emergent network architecture model, known as Software Defined Network (SDN), aims to ease this process by centralizing all network devices into a single central controller, which is entirely manageable by software. However, this centralization may imply problems because, if the SDN controller is attacked, the whole network operation may be impaired. Thus, mechanisms able to guarantee the security of this controller against attacks or anomalies that may prejudice its operation, such as Distributed Denial of Service (DDoS) attacks, are needed. This thesis objectives to present a system for the detection and mitigation of anomalies applied to SDN environments. To accomplish this objective, different anomaly detection techniques are tested, such as the Multi-Layered Perceptron neural network, a machine learning model, and Convolutional Neural Networks (CNN) and Gated Recurrent Units (GRU), deep learning approaches. Furthermore, a method based on Game Theory is proposed to optimize the amount of dropped packets in mitigation policy. Different test scenarios are applied to measure the efficiency of the proposed system. The preliminary results point out that the system achieves reasonable detection rates in DDoS and various types of intrusion attacks. Finally, the system proved to be effective in bringing the network back to its normal state through the mitigation process.

Key words: Software Defined Networking. Game Theory. Machine Learning. Deep Learning. Anomaly Detection.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

Lista de Siglas e Abreviaturas

1D-CNN	*One-dimensional Convolutional Neural Network*
ACO	*Ant Colony Optimization*
CIC	*Canadian Institute for Cybersecurity*
CNN	*Convolutional Neural Networks*
D-MLP	*Dense Multi-Layered Perceptron*
DDoS	*Distributed Denial of Service*
DL	*Deep Learning*
DNN	*Deep Neural Networks*
DNS	*Domain Name System*
DoS	*Denial of Service*
DWT	*Discrete Wavelet Transform*
GA	*Genetic Algorithms*
GADS	*Genetic Algorithms for Digital Signature*
GD	*Gradient Descent*
GRU	*Gated Recurrent Units*
GT	*Game Theory*
HWDS	*Holt-Winters for Digital Signature*
IDS	*Intrusion Detection System*
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	*Internet of Things*
IP	*Internet Protocol*
ISP	*Internet Service Provider*
kNN	*k-Nearest Neighbors*
LR	*Logistic Regression*
LSTM	*Long-Short Term Memory*
MD5	*Message Digest version 5*
ML	*Machine Learning*
MLP	*Multi-Layer Perceptron*
ONF	*Open Networking Foundation*
PCA	*Principal Component Analysis*
PS	*Port Scan*
PSO	*Particle Swarm Optimization*
RGB	*Red, Green and Blue*
RNN	*Recurrent Neural Networks*
RvNN	*Recursive Neural Networks*

SDN	*Software-Defined Networking*
SVDD	*Support Vector Domain Description*
SVM	*Support Vector Machine*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*

Lista de Símbolos

Capítulo 2.3

G	*Tupla de definição do jogo de defesa contra ataques DDoS.*
A_{att}	*Conjuntos de ações possíveis para o atacante.*
A_{def}	*Conjunto de ações possíveis para o sistema de defesa.*
P_{att}	*Função de utilidade para o atacante.*
P_{def}	*Função de utilidade para o sistema de defesa.*
u	*Intensidade do ataque.*
m	*Quantidade de nós atacantes.*
w_i^{att}	*Peso para a métrica i, utilizada no cálculo de P_{att}.*
w_i^{def}	*Peso para a métrica i, utilizada no cálculo de P_{def}.*
E	*Erro normalizado entre o comportamento esperado e o observado.*
BC	*Consumo de banda médio de usuários legítimos, comparados a maliciosos.*
AC	*Custo do ataque para o atacante.*
PL	*Estimativa de perda de pacotes de usuários legítimos em descartes.*
Pkt_{exp}	*Quantidade esperada de pacotes.*
Pkt_{end}	*Quantidade de pacotes observados após o processo de mitigação.*
Pkt_{leg}	*Quantidade de pacotes de usuários legítimos.*
Pkt_{new}	*Quantidade de pacotes de usuários novos.*
D	*Taxa de descarte de novos pacotes*
d_{bits}	*Diferença de largura de banda entre a quantidade esperada e observada.*
B_{exp}	*Volume em bits esperado.*
B_{end}	*Volume em bits observado após o processo de mitigação.*
B_{leg}	*Volume em bits de usuários legítimos.*
B_{new}	*Volume em bits de usuários novos.*
Pb_{leg}	*Proporção em bits de legítimos dentro do total de novos usuários.*
d_{pkt}	*Diferença entre pacotes observados e esperados.*
Pp_{leg}	*Proporção de pacotes de legítimos dentro do total de novos usuários.*

Capítulo 2.4.1

$I_j^{(k)}$	*Soma ponderada realizada pelo neurônio j na camada (k).*
Dim	*Quantidade de dimensões de fluxo avaliadas.*
$W_{ji}^{(k)}$	*Pesos sinápticos que conectam o neurônio j ao i na camada (k).*
x_i	*i-ésimo neurônio da camada de entrada.*
$Y_j^{(k)}$	*Saída calculada do neurônio j na camada (k).*

$\sigma(\cdot)$	*Função de ativação sigmoidal.*
q	*Entrada da função de ativação (resposta prévia do neurônio).*
N	*Total de neurônios da camada.*

Capítulo 2.5.2

h_{t-1}	*Estado oculto (hidden state) do intervalo de tempo* $t-1$.
x_t	*Dados de entrada.*
h_t	*Estado oculto (hidden state) de saída no intervalo* t.
σ	*Função de ativação sigmoidal.*
r_t	*Resultado do portão Reset no intervalo* t.
u_t	*Resultado do portão Update no intervalo* t.
W_r	*Pesos sinápticos da rede neural do portão Reset.*
W_u	*Pesos sinápticos da rede neural do portão Update.*
b_r	*Vetor de bias para a rede neural do portão Reset.*
b_u	*Vetor de bias para a rede neural do portão Update.*
$tanh$	*Função de ativação Tangente Hiperbólica.*
$\tilde{h_t}$	*Saída da rede neural com função de ativação tanh no intervalo* t.
W_o	*Pesos sinápticos da rede neural com função de ativação tanh.*
b_o	*Vetor de bias para a rede neural com função de ativação tanh.*

Lista de Operadores

$[x, y]$	*Concatenação entre* x *e* y
$x * y$	*Convolução entre* x *e* y
$max(x)$	*Máximo de* x
$\|x\|$	*Módulo ou valor absoluto* x
$x \cdot y$	*Produto elemento a elemento entre* x *e* y

SUMÁRIO

1 Introdução

Aplicações de rede e dispositivos conectados que utilizam a Internet como meio de transmissão de dados vêm crescendo rapidamente nos últimos anos, tanto em quantidade quanto em complexidade (MARWAT et al., 2018). A cada dia surgem novas aplicações, tais como redes sociais, comércios *online* e transações bancárias digitais, assim como novos recursos em dispositivos móveis trazidos com a popularização do paradigma Internet das Coisas (*IoT*, do inglês *Internet of Things*) (FRUSTACI et al., 2018). Neste contexto, garantir o bom desempenho da rede, bem como as demandas dessas novas soluções, têm aumentado a complexidade para administração e gerenciamento de segurança das redes tradicionais devido à sua infraestrutura heterogênea e estática.

Rede Definida por *Software* (*SDN*, do inglês *Software Defined Networking*) é um modelo emergente de arquitetura de redes que busca prover recursos para possibilitar o gerenciamento eficiente de ativos de rede e, consequentemente, de aplicações conectadas. Este paradigma tem como principal característica a divisão entre diferentes níveis, separando os planos de controle e de dados dos dispositivos de rede através de um plano de abstração (KREUTZ et al., 2015). Essa divisão permite que o comportamento da rede seja controlado, modificado e gerenciado de forma dinâmica por meio de uma interface programável (controlador central), diferentemente de modelos tradicionais de rede em que os dispositivos são "caixas pretas" proprietárias, o que limita sua flexibilidade com relação a seu controle interno (HAKIRI et al., 2014).

As redes *SDN* proporcionam novos recursos de monitoramento e gerenciamento capazes de melhorar a performance e reduzir gargalos existentes em redes tradicionais. Entretanto, embora haja diversas vantagens em se adotar uma arquitetura *SDN*, essas redes ainda possuem alguns problemas que precisam ser solucionados (LIU et al., 2019c) (MUBARAKALI; ALQAHTANI, 2019). Devido à natureza centralizada da inteligência da rede, o controlador *SDN* pode ser alvo de ataques de Negação de Serviço (*DoS*, do inglês *Denial of Service*) (KALKAN; GUR; ALAGOZ, 2017) (XIA et al., 2015). Ataques do tipo *DoS* buscam exaurir os recursos de rede por meio do envio de um número massivo de solicitações ou pacotes, e costumam ser mais poderosos e de difícil contenção quando executados de forma distribuída (*DDoS*, do inglês *Distributed Denial of Service*). Em servidores, esses ataques objetivam tornar o serviço provido indisponível, enquanto em ataques à infraestrutura, o objetivo é inundar a conexão de rede, consumindo toda a largura de banda disponível (JONKER et al., 2017) (DONG; ABBAS; JAIN, 2019).

Ataques do tipo *DDoS* vêm se intensificando ao longo dos anos, principalmente por conta da popularização do uso de dispositivos *IoT* (GADDAM; WILKIN; ANGELOVA, 2019) (PUNDIR et al., 2020). Segundo Kim, Kim e Jang (2018), estes dispositivos são suscetíveis a infecções de *malware*, os quais se propagam entre equipamentos não-seguros

formando *botnets* (redes de dispositivos infectados de terceiros que podem ser utilizadas em ataques *DDoS*) (MENDES; ALOI; PIMENTA, 2019). De acordo com CISCO (2018), cerca de 83% dos dispositivos de amostra coletada em redes parceiras estavam operando com vulnerabilidades conhecidas, e dispositivos *IoT* estavam implementados sem qualquer planejamento de segurança. Recentemente, dispositivos *IoT* foram utilizados em um ataque *DDoS* contra servidores da Dyn Inc., uma companhia que controla grande parte da infra-estrutura *DNS* da Internet (KOLIAS et al., 2017). Esse ataque é considerado um dos maiores deste tipo, com uma taxa transmissão de 1.2 Tb por segundo.

Além disso, ataques *DDoS* frequentemente são precedidos de ataques de sondagem, tal como técnicas de escaneamento de portas (*Port Scan*), as quais são caracterizadas pela varredura das portas de um servidor buscando encontrar uma abertura para a realização de ataques. Estes ataques são conhecidos como ataques de intrusão e, assim como os ataques *DDoS*, eles vêm se especializando ao longo dos anos (ROHRMANN; ERCOLANI; PATTON, 2017) (LUCHS; DOERR, 2019). Somado a isto, novas ameaças vêm sendo desenvolvidas rapidamente, as quais desafiam a operação de sistemas tradicionais de detecção de intrusão (IDS, do inglês *Intrusion Detection Systems*) em implementações de serviços baseados em nuvem (ALJAMAL et al., 2019) ou ambientes *IoT* (LI; ZHANG, 2019).

Dessa forma, garantir a segurança de sistemas de rede, sejam eles tradicionais ou redes *SDN*, é uma tarefa complexa, uma vez que é necessário garantir a disponibilidade, confiabilidade e integridade dos serviços de rede disponibilizados a usuários finais (WU et al., 2020) (RAFIQUE et al., 2020) (BUTUN; ÖSTERBERG; SONG, 2020). Com o já citado aumento no tráfego de rede (GUBBI et al., 2013) e a heterogeneidade de requisitos que diferentes serviços precisam atender (FRUSTACI et al., 2018), essa tarefa vem se mostrando cada vez mais impraticável por gerentes de rede utilizando soluções reativas. Dessa forma, a utilização de abordagens eficientes para auxiliar no gerenciamento e em processos de segurança autônomos, tais como detecção de ataques e mitigação, é necessária em ambientes *SDN*.

Este trabalho tem como objetivo propor um sistema para a detecção e mitigação de ataques *DDoS* e de intrusão em ambientes *SDN*, o qual é dividido em três grandes módulos: Detecção, Identificação e Mitigação.

O Módulo de Detecção implementa diferentes métodos responsáveis por detectar os ataques de rede. Dentre os diversos tipos de abordagens possíveis (FERNANDES JR. et al., 2019), se destacam atualmente na literatura modelos baseados em aprendizagem de máquina (*ML*, do inglês *Machine Learning*) (ELTANBOULY et al., 2020) (KUNAL; DUA, 2019) (MEYER; LABIT, 2020). De acordo com Wehbi et al. (2019), estes métodos são um tipo de inteligência artificial, sendo amplamente aplicados em tarefas de classificação, tais como detecção de anomalias. Neste contexto, foi proposta a utilização da rede neural Perceptron de Múltiplas Camadas (*MLP*, do inglês *Multi-Layer Perceptron*).

Além disso, métodos de aprendizagem profunda (*DL*, do inglês *Deep Learning*), uma sub-classe dos métodos de *ML*, vêm se popularizando atualmente (ROOPAK; TIAN; CHAMBERS, 2019) (ISHAQUE; HUDEC, 2019) devido à sua capacidade de extração automática de atributos e identificação de características não-óbvias nos dados avaliados (WANG et al., 2020). Com isso, essas abordagens são capazes de gerar detecções precisas, mesmo em ataques mais sutis. Foi proposta a utilização de dois métodos que se enquadram nesta classe, sendo eles a Rede Neural Convolucional (*CNN*, do inglês *Convolutional Neural Network*) e *Gated Recurrent Units* (*GRU*).

O Módulo de Identificação, por sua vez, é responsável por coletar e disponibilizar ao controlador *SDN* informações importantes acerca do tráfego, seja ele normal ou anômalo. A coleta de informações relacionadas ao tráfego anômalo, tais como endereços *IP*, portas e protocolos, auxilia na identificação da anomalia detectada em casos em que o método de detecção seja binário. Com isso, o Módulo de Mitigação pode adotar contra-medidas mais específicas, aumentando a eficiência da resposta do sistema. Por outro lado, a coleta e armazenamento de dados relativos a tráfego normal auxiliam na proteção de usuários legítimos contra políticas de descarte de pacotes, contribuindo para a manutenção do estado normal da rede *SDN*. Dessa forma, essa coleta de informações contribui para potencializar os resultados obtidos através dos processos de mitigação.

Por fim, no Módulo de Mitigação, responsável pela definição de contra-medidas contra os ataques detectados, foram propostas duas abordagens de mitigação, uma probabilística e outra direcionada. Com relação à primeira, foi proposta a utilização de Teoria de Jogos (*GT*, do inglês *Game Theory*) na geração de políticas de descarte de pacotes contra ataques *DDoS*. Segundo Ghimire (2019), *GT* é eficaz em sistemas em que uma decisão precisa ser tomada entre diferentes usuários que possuam conflito de interesses (SUNG; HSIAO, 2019). No contexto de segurança de redes de computadores, este jogo é disputado entre o sistema de defesa, o qual objetiva a manutenção do funcionamento normal da rede, e o atacante, que busca a interrupção dos serviços prestados (ataques *DDoS*). Além disso, uma abordagem de mitigação direcionada é proposta e avaliada em casos em que seja possível a identificação da origem do ataque (direcionada ao endereço *IP* do atacante).

Dentre as principais contribuições deste trabalho, pode-se citar:

- A proposta de um sistema de defesa aplicado a ambientes *SDN* contra ataques *DDoS* e de intrusão.

- Modelagem de uma abordagem baseada em Teoria de Jogos para a otimização da taxa de descarte de pacotes em políticas de mitigação de ataques *DDoS*, bem como implementação de mitigação direcionada contra ataques de intrusão. Embora a mitigação de ataques, principalmente *DDoS*, seja um assunto amplamente estudado, este campo ainda apresenta desafios, uma vez que a diferenciação entre usuários legítimos e atacantes não é trivial. Diferentes modelos na literatura apresentam

abordagens para a mitigação de *DDoS*, muitas vezes se baseando em limiares de ativação (YIN; ZHANG; YANG, 2018), os quais não são generalizáveis para ataques de diferentes intensidades, ou mesmo políticas de bloqueio geral de pacotes (XUANYUAN; RAMSURRUN; SEEAM, 2019), que prejudicam usuários legítimos. O modelo proposto possui o propósito de mitigar os efeitos do ataque *DDoS* independentemente de sua intensidade, enquanto reduz o impacto causado em usuários legítimos da *SDN*.

- Estudo e implementação de um modelo de aprendizagem de máquina, utilizando redes perceptron de múltiplas camadas (*MLP*), aplicado à detecção de ataques *DDoS* e Escaneamento de Portas (*PS*, do inglês *Port Scan*), bem como a realização de testes comparativos com outros métodos. Ainda que modelos de aprendizagem de máquina normalmente apresentem elevado custo de treinamento, sua aplicação é relativamente leve em comparação com outras abordagens de detecção, tais como as meta-heurísticas Otimização por Colônia de Formigas (*ACO*, do inglês *Ant Colony Optimization*)(ZANG; GONG; HU, 2019) e Algoritmos Genéticos (*GA*, do inglês *Genetic Algorithms*) (SAMPATH et al., 2020). Esse baixo custo computacional é uma característica essencial em sistemas de defesa que operam próximos do tempo real (*near real-time*), como o sistema proposto.

- A proposta de um modelo de mitigação inversa (aplicado às redes de origem do ataque) contra ataques *DDoS* em ambientes *SDN*. Abordagens convencionais de mitigação são comumente implementadas no servidor alvo (SMITH-PERRONE; SIMS, 2017) (XUANYUAN; RAMSURRUN; SEEAM, 2019). A mitigação inversa proposta introduz o paradigma de "dividir para conquistar"neste contexto, tratando os ataques *DDoS* ainda em sua rede de origem. Essa abordagem protege o controlador *SDN* contra efeitos colaterais desses ataques (ataque indireto), consequentemente protegendo o servidor alvo.

- Estudo de diferentes intervalos de coleta de dados (1 minuto, 5 segundos, 1 segundo e coleta de fluxos *IP* individuais), permitindo a diminuição do tempo de resposta do sistema proposto. Abordagens de detecção tradicionais costumam avaliar o tráfego por meio de diferentes intervalos de coleta. O tempo desse intervalo varia de acordo com o método aplicado e a capacidade de processamento do sistema, podendo ser de cinco minutos (ADANIYA et al., 2012; BEREZIńSKI; JASIUL; SZPYRKA, 2015), um minuto (SUN et al., 2016; HAMAMOTO et al., 2018), ou até na ordem de segundos (CARVALHO et al., 2018). Quanto menor este tempo, maior a velocidade de resposta do sistema frente a uma ameaça detectada e, consequentemente, menor é o impacto causado na rede. A detecção utilizada pelo sistema proposto objetiva análises próximas do tempo real (*near real-time*).

- Execução de diferentes cenários de testes, utilizando desde fluxos *IP* emulados a bases de dados públicas, na avaliação de performance do sistema proposto. Ainda que o uso de emuladores de rede seja importante para a realização de testes em ambientes *SDN*, a utilização de bases de dados públicas é essencial, uma vez que aumenta a confiabilidade dos resultados. Enquanto diversos trabalhos utilizam a emulação como único ambiente de testes (LAN; PAN, 2019) (BHUNIA; GURUSAMY, 2017), a presente tese também apresenta resultados derivados de bases públicas, o que garante a replicabilidade dos mesmos.

Ainda que se possa considerar o elevado custo de treinamento e a aprendizagem supervisionada como desvantagens, o sistema proposto é o primeiro (com base em revisão da literatura) que, utilizando uma mesma estrutura de defesa, é capaz de detectar e mitigar diferentes tipos de ataques *DDoS* e de intrusão por meio da análise individualizada (*near real-time*) de fluxos *IP*.

O restante deste trabalho está dividido da seguinte forma: no Capítulo 2, uma fundamentação teórica acerca dos principais temas abordados neste trabalho é apresentada; no Capítulo 3 o modelo proposto é apresentado com base nos artigos científicos desenvolvidos; no Capítulo 4, os resultados dos artigos desenvolvidos são detalhados e discutidos; no Capítulo 5 são apresentadas as considerações finais e trabalhos futuros; finalmente, os Apêndices A a D apresentam os artigos desenvolvidos.

2 Fundamentação teórica

Neste capítulo é apresentada uma fundamentação teórica acerca dos principais assuntos abordados no desenvolvimento da presente tese. A explicação desses assuntos é realizada com base em revisão da literatura, paralelamente abordando a forma com que foram aplicados no sistema de defesa proposto.

2.1 *Software Defined Networking*

Segundo Masoudi e Ghaffari (2016), a comunicação realizada em redes de computadores tradicionais é baseada principalmente na utilização de dispositivos como *switches* e roteadores. Ainda segundo os autores, esses dispositivos possuem circuitos integrados de aplicação específica, ou seja, são ambientes em que *hardware* e *software* são desenvolvidos com foco em potencializar os processos de comunicação através da transmissão e controle de dados.

Em redes tradicionais, são agrupados em dispositivos individuais os planos de controle, responsável pelo gerenciamento do tráfego de rede, e o plano de dados, que realiza o encaminhamento desses dados de acordo com as regras e políticas estabelecidas pelo plano de controle (PARASHAR; POONIA; SATISH, 2019).

Como redes de computadores de larga-escala vêm crescendo continuamente em tamanho e complexidade, mecanismos de gerenciamento que sejam eficientes e de rápida resposta são cada vez mais necessários. A popularização de tecnologias de rede proporciona o surgimento de um grande número de aplicações *online*, tais como dispositivos conectados (IoT) e ambientes de computação em nuvem. Como resultado, a quantidade de informações valiosas e relevantes que trafegam através de redes de computadores cresceu substancialmente nos últimos anos (CHAABOUNI et al., 2019). Com isso, a necessidade de maior largura de banda, bem como maior adaptação a ambientes com dispositivos heterogêneos e gerenciamento dinâmico, são características fundamentais para a manutenção da escalabilidade da rede, garantindo a qualidade dos serviços prestados.

Dessa forma, o paradigma de Redes Definidas por *Software* (SDN) surge como uma poderosa e flexível arquitetura de redes. Este paradigma foi desenvolvido para simplificar e aperfeiçoar o processo de gerenciamento por meio de abstrações nas funções de rede, bem como maior flexibilidade dos dispositivos de controle.

Redes SDN separam o plano de controle do plano de dados, ou seja, o processo de controle do roteamento de pacotes é implementado em nível de *software* por um controlador centralizado e programável (YAN et al., 2016) (LI; MENG; KWOK, 2016), ao invés de ser controlado individualmente por roteadores e *switches*. Segundo Scott-Hayward,

Natarajan e Sezer (2016), essa centralização do controle de funções lógicas (plano de controle) mantém o estado da rede e provê instruções de operação ao plano de dados. Os dispositivos de rede do plano de dados, por sua vez, encaminham os pacotes de acordo com as instruções de controle recebidas.

Redes *SDN* já se provaram eficazes em diversos cenários de aplicação, tais como no *backbone* da rede Google (JAIN et al., 2013) e a nuvem pública da Microsoft (PATEL et al., 2013).

São diversas as características que tornam ambientes *SDN* eficientes para redes do futuro (PARASHAR; POONIA; SATISH, 2019). Dentre as principais características que definem este paradigma, descritas em (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016), pode-se destacar: i) controle lógico centralizado; ii) interfaces abertas e programáveis; e iii) integração de dispositivos de terceiros.

O controle lógico centralizado se refere à característica essencial de ambientes *SDN*, uma vez que a utilização do controlador permite a centralização lógica, embora fisicamente distribuída, da rede (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016). O controlador é capaz de manter uma visão geral de toda a estrutura de roteamento da rede, sendo capaz de programar encaminhamentos de pacotes baseado em políticas definidas por *software*.

Com relação ao segundo tópico (interfaces abertas e programáveis), a separação dos planos de controle e dados proporcionada pela *SDN* permite que os dispositivos de encaminhamento sejam simplificados, uma vez que não necessitam realizar nenhuma rotina lógica de controle. Com isso, ambientes *SDN* permitem que os *software* de rede possam evoluir de forma independente, aumentando o potencial de implementação de tecnologias inovadoras (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016).

Por fim, ambientes *SDN* permitem a integração de dispositivos de diferentes desenvolvedores na arquitetura, de forma semelhante a sistemas operacionais, em que aplicativos de diferentes módulos, bibliotecas e desenvolvedores podem ser incorporados ao sistema, inclusive interagindo entre si (SCOTT-HAYWARD; NATARAJAN; SEZER, 2016). Essa característica permite a escalabilidade e flexibilidade na arquitetura de rede, possibilitando ainda a redução de custos de serviços proprietários.

Apesar das características discutidas, as redes *SDN* também são sujeitas a ameaças e vulnerabilidades de segurança. Segundo Li, Meng e Kwok (2016), eventos anômalos que impactam o funcionamento de redes *SDN* podem ser divididos em três grupos: i) ataques direcionados ao plano de controle; ii) o comprometimento da comunicação entre os planos de dados e controle, e; iii) ameaças aos equipamentos do plano de dados. O foco do sistema proposto neste trabalho é a defesa contra eventos do primeiro grupo.

Devido à natureza centralizada da inteligência da rede por meio do controlador, este dispositivo pode ser considerado um ponto crítico de falhas. O controlador pode ser alvo de ações maliciosas, tais como técnicas de intrusão (AJAEIYA et al., 2017) ou ataques de negação de serviço (*DoS*) (KALKAN; GUR; ALAGOZ, 2017) (XIA et al., 2015). Ataques

DoS, bem como sua versão distribuída (*DDoS*), são especialmente eficazes contra redes *SDN*, uma vez que são capazes de gerar uma enorme quantidade de solicitações que sobrecarregam o controlador, consequentemente, prejudicando o funcionamento da rede como um todo (SCARANTI et al., 2020).

Dessa forma, o sistema de defesa proposto objetiva defender o controlador *SDN* contra anomalias de ataque, sejam elas de origem externa ou interna à rede. Num primeiro momento, o sistema foi implementado em um dispositivo de borda, protegendo o controlador contra ataques externos. Posteriormente, o sistema de defesa foi implementado diretamente no controlador, utilizando suas características para interagir com os planos de dados (solicitações e coletas de dados) e controle (definição de políticas de mitigação).

2.2 Detecção de Ataques

Segundo Chandola, Banerjee e Kumar (2009), anomalias são padrões que não estão em conformidade com o comportamento esperado, comumente sendo referenciados também como *outliers* ou exceções. Em redes de computadores, uma anomalia pode ser considerada uma modificação do comportamento esperado do tráfego de rede (PROENÇA et al., 2004) (PROENÇA; ZARPELAO; MENDES, 2005). Como exemplo pode-se citar a realização de ataque distribuído de negação de serviço (*DDoS*), situação em que o volume de dados trafegando na rede aumenta significativamente.

Anomalias nem sempre estão relacionadas a ataques ou ações voltadas a danificar sistemas computacionais, podendo ser causadas, por exemplo, por falha de *hardware*, falha humana, ou mesmo por uma grande quantidade de usuários legítimos requisitando acesso a um serviço (FERNANDES JR. et al., 2019).

Por ser um tema amplamente difundido na área de segurança de redes de computadores, anomalias são definidas e classificadas na literatura de diferentes formas. A classificação proposta por FERNANDES JR. et al. (2019) divide anomalias em dois grupos, com base em suas características: i) de acordo com sua natureza (agrupados por como são caracterizadas, independentemente se são maliciosas ou não); e ii) de acordo com seu aspecto causal (agrupadas de acordo com sua causa, levando em conta seu aspecto, seja ele malicioso ou não).

Levando em consideração o segundo grupo, o qual permite a distinção entre anomalias maliciosas (ataques) e não-maliciosas, é possível identificar quais situações possuem maior potencial de impacto em redes *SDN*, de acordo com as características apresentadas na Seção 2.1. Dessa forma, o foco deste trabalho está em defender o controlador *SDN* contra ataques, sejam eles de negação de serviços (*DoS* ou *DDoS*) ou técnicas de intrusão (tais como *PS*).

Ataques *DoS* têm como objetivo esgotar os recursos de rede por meio do envio de solicitações ou pacotes de forma massiva. Não sendo possível processar a quantidade

requisitada de informações, o servidor ou controlador *SDN* passa a não atender usuários legítimos, ou seja, negar serviço (BHARDWAJ et al., 2016).

Esses ataques costumam ser mais impactantes e de difícil contenção quando realizados de forma distribuída (*DDoS*). Ataques *DDoS* normalmente utilizam *botnets* como *hosts* atacantes, que são uma coleção de diversas máquinas infectadas por *malware* e remotamente controladas por um usuário malicioso (HOQUE; BHATTACHARYYA; KALITA, 2015). Por serem compostas por muitos dispositivos, essas *botnets* aumentam significativamente o potencial do ataque, podendo sobrecarregar tanto links de comunicação quanto servidores ou controladores *SDN*.

Além disso, invasões representam um conjunto de diferentes tipos de ataques, com características específicas, que objetivam roubo de informações, acesso não-autorizado, construção de *botnets*, entre outros. Alguns autores inclusive consideram ataques *DDoS* como formas de invasão, pois muitas vezes são utilizados como forma de distração enquanto ataques mais complexos são realizados (BORKAR; DONODE; KUMARI, 2017).

Dentre as diferentes abordagens de intrusão, pode-se destacar a técnica de escaneamento de portas (*PS*), utilizada na identificação de portas abertas, serviços disponíveis ou desprotegidos (desatualizados) em um *host*. Uma de suas principais características é a dificuldade de detecção, uma vez que essas abordagens não geram alteração significativa de volume de tráfego. Além disso, as informações coletadas podem ser utilizadas por outras técnicas em ataques mais específicos.

Assim, a necessidade da implementação de medidas de segurança contra diferentes ataques em ambientes *SDN* se mostra evidente, uma vez que seu funcionamento é baseado na centralização lógica da rede. Proteger o controlador *SDN* implica em proteger o funcionamento da rede e, consequentemente, em garantir a qualidade dos serviços prestados aos usuários finais.

2.3 Teoria de Jogos

De acordo com Roy et al. (2010), Teoria de Jogos (*GT*) descreve cenários de decisões envolvendo múltiplas pessoas (jogadores) por meio de jogos, no qual o resultado é a melhor recompensa individual levando em consideração a antecipação de ações racionais dos outros jogadores. Em outras palavras, é um método para se traduzir um problema do mundo real em um jogo, no qual dois ou mais jogadores tentam vencer.

Na abordagem proposta neste trabalho (*Game theoretical approach* - abordagem *GT*), dois jogadores competem no jogo: o atacante (usuário malicioso) e o sistema de defesa. Os objetivos do jogo são diferentes para cada jogador, mas se complementam. Enquanto o atacante tenta maximizar o dano causado à rede e reduzir suas chances de ser detectado, o sistema de defesa objetiva reduzir o impacto causado pelo atacante e preservar o

funcionamento regular da rede. Deste modo, a abordagem GT pode ser definida como um vetor de 4 tuplas:

$$G = (A_{att}, A_{def}, P_{att}, P_{def}) \tag{2.1}$$

Os elementos do vetor G são detalhados a seguir.

Cada jogador possui um conjunto limitado de ações. O elemento A_{att} representa o conjunto de ações possíveis que o atacante pode executar, ou seja, todas as possíveis estratégias do atacante. Duas ações compõem esse conjunto:

- Mudar a intensidade (u) do ataque, ou seja, a quantidade de pacotes por segundo direcionados à rede por cada nó atacante;
- Modificar a quantidade de nós atacantes (m).

De forma similar, o elemento A_{def} representa o conjunto de ações que o sistema de defesa pode executar, ou seja, as estratégias possíveis de defesa do sistema. Três ações compõem este conjunto:

- Permitir que pacotes sejam transportados ao controlador *SDN*;
- Descartar pacotes no *firewall* para proteger o controlador *SDN* por meio de uma taxa de descarte específica;
- Redirecionar pacotes a um *Honeypot*, uma máquina isolada e vulnerável propositalmente com o objetivo de proporcionar análises aprofundadas sobre o comportamento do ataque, sua motivação e origem.

Essas ações são quantificadas por equações específicas, gerando um valor final denominado "Função Utilidade" ou *payoff*. De acordo com Roy et al. (2010), Função Utilidade é uma recompensa positiva ou negativa atribuída a um jogador com relação a uma ação realizada em um jogo.

Assim, os elementos P_{att} e P_{def} são as funções utilidade para o atacante e o sistema de defesa, respectivamente. Os *payoffs* do atacante e sistema de defesa são representados como:

$$P = Recompensa - Custo \tag{2.2}$$

Portanto, o *payoff* relativo ao atacante (P_{att}) e ao sistema de defesa (P_{def}) são representados pelas Equações (2.3) e (2.4).

$$P_{att} = w_1^{att} \cdot E - w_2^{att} \cdot BC - w_3^{att} \cdot AC + w_4^{att} \cdot PL \tag{2.3}$$

$$P_{def} = -w_1^{def} \cdot E + w_2^{def} \cdot BC + w_3^{def} \cdot AC - w_4^{def} \cdot PL \tag{2.4}$$

Nessas equações, w^{att} e w^{def} são parâmetros de peso para cada métrica, relativos ao atacante e ao sistema de defesa, respectivamente. Além disso, as variáveis restantes representam: E representa o erro normalizado entre o comportamento esperado e o comportamento atual da rede; BC representa o consumo de banda médio de usuários legítimos em comparação com usuários maliciosos; AC é o custo do ataque para o atacante, e; PL é a estimativa de perda de pacotes de usuários legítimos por meio da política de descarte durante o processo de mitigação.

A métrica de erro normalizado E é definida por (2.5), sendo calculada por meio da comparação da quantidade de pacotes que era esperada Pkt_{exp} no intervalo de tempo corrente (caracterização de tráfego da dimensão Pacotes/s, calculado por algum método de predição como, por exemplo, o *Holt Winters for Digital Signature - HWDS* (ASSIS et al., 2013)) com o número de pacotes observados depois do processo de mitigação. Além disso, o valor resultante deve ser normalizado pelo máximo entre os pacotes esperados e observados, visando o ajuste da métrica para possibilitar interação com outras funções de custo/recompensa. Finalmente, o valor absoluto do erro normalizado é considerado um problema de minimização de erro, uma vez que o descarte excessivo de pacotes pode impactar negativamente o funcionamento da rede (valor ótimo é atingido quando $E = 0$).

$$E = \left| \frac{Pkt_{end} - Pkt_{exp}}{max(Pkt_{end}, Pkt_{exp})} \right| \tag{2.5}$$

Os pacotes observados após o processo de mitigação (Pkt_{end}) são obtidos por meio da Equação 2.6.

$$Pkt_{end} = Pkt_{leg} + Pkt_{new} \cdot (1 - D) \tag{2.6}$$

Nessa equação, Pkt_{leg} representa a quantidade de pacotes de usuários legítimos (conhecido através de análise dos dados do Módulo de Identificação, descrito no Capítulo (3)). Pkt_{new} é a quantidade de pacotes de usuários novos (ainda desconhecidos para o Módulo de Identificação), que une novos usuários legítimos e maliciosos. Finalmente, D é a taxa de descarte de novos pacotes, variando de 0 a 100%.

Assim como a variável Pkt_{leg}, o consumo médio de largura de banda (BC) pode ser calculado utilizando informações providas pelo Módulo de Identificação (Capítulo (3)). Primeiramente, a diferença de largura de banda entre a quantidade esperada de bits (B_{exp}) e a quantidade observada de bits (B_{end}) pode ser expressada pela Equação (2.7).

$$\begin{aligned} d_{bits} &= B_{exp} - B_{end} \quad \text{, em que} \\ B_{end} &= B_{leg} + B_{new} \end{aligned} \tag{2.7}$$

Nessa equação, de forma similar à Equação 2.6, B_{leg} representa a quantidade de bits de usuários legítimos, enquanto B_{new} é a quantidade de bits de usuários novos, que une novos usuários legítimos e maliciosos.

Com a diferença d_{bits} calculada, a proporção de bits de usuários legítimos dentro do total de bits de novos usuários (Pb_{leg}) pode ser determinada através da Equação (2.8).

$$Pb_{leg} = \frac{(B_{new} - d_{bits})}{B_{new}} \tag{2.8}$$

Finalmente, o consumo médio de largura de banda (BC), medido em bits/s, pode ser calculado considerando a taxa de descarte D, o número de *hosts* atacantes m e a intensidade do ataque u, representado na Equação (2.9).

$$BC = \frac{B_{leg} + D\left[B_{new} \cdot Pb_{leg} - m \cdot u(1 - Pb_{leg})\right]}{B_{leg} + D(B_{new} + m \cdot u)} \tag{2.9}$$

O custo do ataque (AC) é considerado linear com relação ao número de *hosts* m controlados pelo atacante, assim como proposto em Song et al. (2017). Esse parâmetro é normalizado para facilitar a interação com outras métricas, sendo obtido através da Equação (2.10).

$$AC = \frac{m}{max(m)} \tag{2.10}$$

Por fim, a taxa estimada de perda de pacotes (PL) pode ser obtida considerando as quantidades de pacotes esperados e observados em conjunto com os parâmetros de ataque (número de *hosts* atacantes m e a intensidade do ataque u, aqui medidos em pacotes/s) e o parâmetro de defesa D (taxa de descarte de novos pacotes). Para isso, a distância ou diferença d_{pkt} deve ser calculada por meio da equação (2.11).

$$d_{pkt} = (Pkt_{leg} + Pkt_{new}) - Pkt_{exp} \tag{2.11}$$

Então, o resultado obtido através da Equação (2.11) pode ser aplicado no cálculo da proporção estimada de pacotes de usuários legítimos dentro dos novos pacotes, como ilustrado na Equação (2.12).

$$Pp_{leg} = \frac{Pkt_{leg} - d_{pkt}}{Pkt_{new}} \tag{2.12}$$

Assim, o resultado obtido através da Equação (2.12) pode ser aplicado na Equação (2.13) para o cálculo da taxa PL.

$$PL = 1 - \frac{Pkt_{leg} + (1 - D)Pkt_{new} \cdot Pp_{leg}}{Pkt_{exp}} \tag{2.13}$$

Ao final do cálculo dos *payoffs*, dado pelas Equações (2.3) e (2.4), o método GT gera uma matriz que cruza funções objetivo de cada diferente possibilidade de ataque

(combinação de diferentes valores de m e u) com cada estratégia de defesa possível (cada possível valor de D). As células dessa matriz são organizadas com o par de informação (P_{att}, P_{def}).

No jogo modelado neste trabalho, a recompensa recebida por um jogador é a perda do outro (w_i^{att} e w_i^{def} são iguais para todos os valores de i). Pode-se citar como exemplo a situação em que o atacante é recompensado positivamente. Neste caso, o sistema de defesa será penalizado na mesma intensidade. Este tipo de dinâmica configura um jogo do tipo "soma-zero" (WU et al., 2010).

Além disso, a solução ótima para um jogo é encontrada através do equilíbrio de Nash, situação estável na qual nenhum jogador racionalmente escolheria mudar sua estratégia, de modo que qualquer ação possível acarretaria na diminuição de seu *payoff* (NASH, 1950).

Segundo Osborne e Rubinstein (1994), o equilíbrio de Nash de jogos "soma-zero" sempre existe, e pode ser alcançado através da transformação do jogo em um problema de otimização linear que, por sua vez, pode ser resolvido através do teorema Minimax, aplicado sobre a matriz de *payoffs* calculada. Embora possam existir múltiplos pontos de equilíbrio de Nash em um problema, em jogos competitivos de dois jogadores classificados como "soma-zero" todos os pontos de equilíbrio possuem o mesmo valor de *payoff* (NASH, 1950). Com isso, é necessário encontrar apenas um ponto de equilíbrio de Nash para que se encontre a solução ótima do problema.

Teoria de jogos pode ser aplicada em diversos cenários (SONG et al., 2020) (MOURA; HUTCHISON, 2019). Em (SUNG; HSIAO, 2019), por exemplo, os autores abordam o problema da venda online de ingressos para eventos, os quais frequentemente se esgotam em instantes. Essa situação ocorre pois cambistas compram grande quantidade de ingressos, objetivando revenda futura, mas deixando usuários legítimos sem acesso. Os autores destacam que essa disputa entre cambistas e usuários legítimos gera comportamento semelhante a *DDoS* no servidor de venda. Para mitigar este problema, os autores propuseram um mecanismo baseado em teoria de jogos, o qual submete usuários a *puzzles* de diferentes dificuldades. Com isso, o acesso de cambistas aos recursos (ingressos) é dificultado, equilibrando o sistema e mitigando o congestionamento no servidor.

Em (SHRIVASTAVA; RAMAKRISHNA; HOTA, 2019), os autores propuseram um modelo baseado em *GT* que utiliza o equilíbrio de Nash para expor tentativas de ataque de um jogador adversário, verificando a melhor medida de defesa contra este ataque. Os autores utilizaram uma combinação entre *Naive-bayes* e o método de clusterização k-*means* para avaliar dados não rotulados e explorar ataques de *malware*. Por fim, a abordagem *GT* é utilizada como abordagem de mitigação contra ataques *IoT*, utilizando *firewall* e um *IDS*.

2.4 Aprendizagem de Máquina e Redes Neurais

De acordo com Qiu et al. (2016), aprendizagem de máquina (*ML*) é um campo de pesquisa focado em sistemas e algoritmos de aprendizado. É uma área altamente interdisciplinar, que se utiliza de conceitos de diversos campos de pesquisa, tais como teoria da informação, estatística, ciência cognitiva, entre outros.

Métodos de *ML* podem ser divididos em 3 categorias principais, as quais designam a forma com que o processo de aprendizado ocorre. São elas a aprendizagem supervisionada, aprendizagem não-supervisionada e a aprendizagem por reforço (BISHOP, 2006). Aprendizagem supervisionada é aquela em que o método ajusta seus pesos sinápticos com auxílio de dados rotulados no processo de treinamento, ou seja, o aprendizado ocorre com a supervisão de um agente externo. Na aprendizagem não-supervisionada, por sua vez, os métodos extraem características e classificam os dados por conta própria, sem o apoio de rótulos, utilizando técnicas de clusterização, por exemplo. Por fim, a aprendizagem por reforço é aquela em que o método realiza seus ajustes sinápticos com base em respostas (*feedback*) recebidas através de iterações com o ambiente externo.

Segundo Qiu et al. (2016), abordagens de aprendizagem não-supervisionada e supervisionada são preferíveis em aplicações de análise de dados, enquanto a aprendizagem por reforço tem melhores resultados em processos de tomada de decisão. A detecção de ataques, tema central deste trabalho, se enquadra no campo de análise de dados.

Dentre abordagens de aprendizagem supervisionada, pode-se citar o trabalho desenvolvido por (ZHANG et al., 2019), em que os autores propuseram um esquema de detecção de anomalias baseado em máquina de vetor de suporte (*SVM*, do inglês *Support Vector Machine*), por meio da extração e otimização de *features* de treinamento. O modelo proposto treina a *SVM* através da divergência de *Kullback-Leibler* e correlação cruzada, a qual é calculada pelos planos de dados e controle do tráfego. Os autores destacam que essa abordagem se mostrou capaz de identificar intrusões e ataques de curta duração no tráfego da rede.

Em (YANG et al., 2020), por sua vez, os autores propuseram um algoritmo de classificação distribuído, denominado *SEED-kNN*, baseado no modelo de aprendizagem de máquina k-*Nearest Neighbors* (*kNN*). Este modelo foi aplicado para prevenir exposição de fluxos de controle e informações em ambientes industriais *IoT*, simultaneamente suportando classificação de dados de larga-escala em servidores distribuídos. Os autores realizaram testes de performance, e evidenciaram que o modelo proposto atingiu satisfatória segurança semântica e alta taxa de acurácia em seus processos de classificação.

Além disso, uma subcategoria da aprendizagem de máquina, denominada Redes Neurais (*Neural Networks*), vem ganhando espaço entre pesquisas da área nos últimos anos (CHIROMA et al., 2019) (ABIODUN et al., 2019). Segundo Haykin (2011), uma rede neural é uma máquina desenhada para modelar a forma com que o cérebro executa tarefas,

empregando uma interconexão massiva de células computacionais simples, denominadas "neurônios", "unidades de processamento" ou "*perceptrons*."

Dentre as principais características de redes neurais, SILVA, SPATTI e FLAUZINO (2010) destacam: adaptação por experiência, capacidade de aprendizado, habilidade de generalização, organização de dados, tolerância a falhas, armazenamento distribuído e a facilidade de prototipagem.

2.4.1 *Multi-Layer Perceptron - MLP*

Neste trabalho, foi introduzida a aplicação de um método de redes neurais, denominado Redes Perceptron de Múltiplas Camadas (*MLP*), aplicado ao processo de detecção de ataques em ambientes *SDN*. A *MLP* é uma rede neural baseada no funcionamento de redes Perceptron que possuam pelo menos uma camada oculta (camada entre a entrada e a saída da rede neural) em sua topologia, uma das principais características de redes *MLP* em comparação com modelos mais simples. Segundo Haykin (2011), os neurônios ocultos agem como detectores de características, desempenhando um papel fundamental no funcionamento do modelo.

Redes *MLP* são redes neurais de aprendizagem supervisionada que operam sem retroalimentação (rede do tipo *feedforward*). Os dados são inseridos separadamente através de uma camada de entrada (*input layer*), passam pelas camadas ocultas (*hidden layers*) e, finalmente, saem através da camada de saída (*output layer*), que apresenta os resultados da classificação realizada pela rede. A topologia é representada como um grafo totalmente conectado, ou seja, cada sinal de entrada é conectado com cada um dos neurônios intermediários. Por sua vez, esses neurônios podem se conectar com cada um dos neurônios de uma segunda camada oculta (varia conforme a aplicação) ou com cada um dos neurônios da camada de saída. A Figura 1 apresenta uma topologia que exemplifica essa organização.

O processo de aprendizagem se dá pelo treinamento dos pesos sinápticos, ou seja, as linhas que conectam os neurônios da rede. Através de repetidas iterações, a rede organiza diferentes pesos para cada conexão de modo que a saída seja condizente com a classificação desejada, levando em consideração os dados de treinamento (entrada e saída desejada, ou seja, os rótulos ou *labels* dos dados). Este treinamento opera através do algoritmo de *Backpropagation*. Em suma, os dados de entrada são processados pela rede neural, gerando uma saída. Essas respostas são utilizadas em conjunto com as respostas esperadas (rótulos) para ajustar os pesos sinápticos da rede, camada a camada, fazendo o percurso de volta aos pesos da camada de entrada. Por fim, os dados são novamente submetidos à rede (com pesos já ajustados), e a uma métrica de erro é utilizada para se mensurar a precisão da classificação. Este processo se repete até que um limiar predefinido de erro seja atingido, ou um número específico de repetições ocorra.

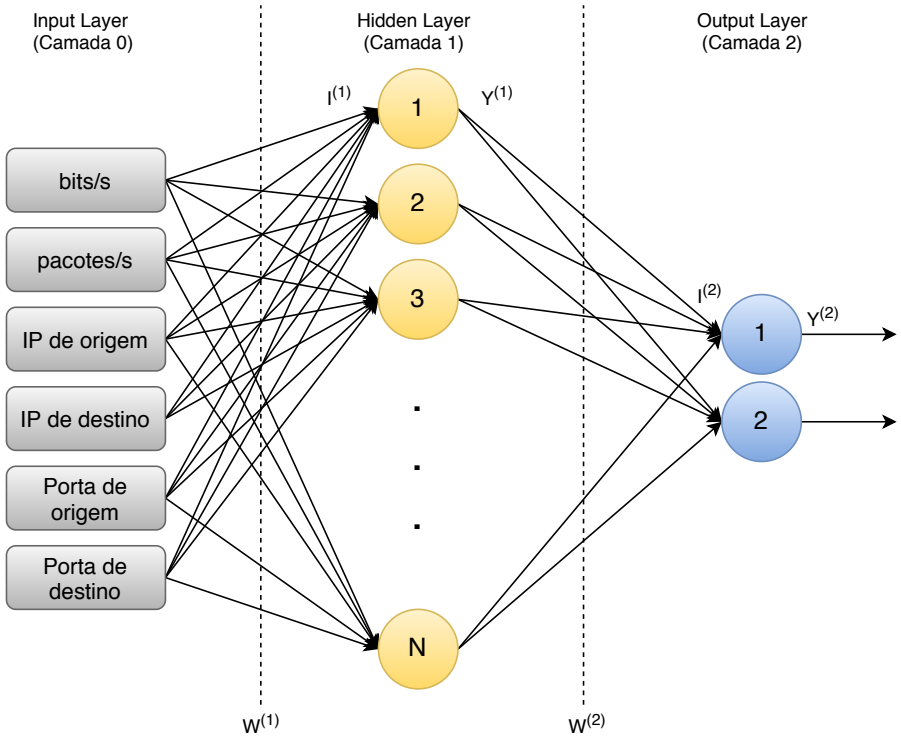


Figura 1 – Topologia de uma rede *MLP* com 1 camada oculta de N neurônios e 2 neurônios de saída.

Com a *MLP* treinada, levando em consideração a arquitetura da Figura 1, o processo de classificação se dá da seguinte forma. Inicialmente, são calculados os valores de $I_j^{(1)}$ e $Y_j^{(1)}$, dados pelas Equações (2.14) e (2.15), respectivamente.

$$I_j^{(1)} = \sum_{i=0}^{Dim} W_{ji}^{(1)} \cdot x_i \tag{2.14}$$

$$Y_j^{(1)} = \sigma(I_j^{(1)}) \tag{2.15}$$

Na Equação (2.14), $I_j^{(k)}$ representa a soma ponderada realizada pelo neurônio j na camada (k), Dim é a quantidade dimensões de fluxo avaliadas, $W_{ji}^{(1)}$ são os pesos sinápticos que conectam o neurônio j ao neurônio i na camada neural seguinte, e x_i é o i-ésimo neurônio da camada de entrada.

Na Equação (2.15), por sua vez, $Y_j^{(k)}$ representa a saída calculada do *Perceptron* j na camada k, e $\sigma(\cdot)$ é uma função de ativação. Neste trabalho, foi adotada a função de ativação logística (ou sigmoidal) com parâmetro $\beta = 1$ (função logística padrão), definida pela Equação (2.16), na qual q representa a entrada da função de ativação (resposta prévia do neurônio).

$$\sigma(q) = \frac{1}{1 + e^{-\beta \cdot q}} \tag{2.16}$$

De forma similar, após a geração dos resultados da camada oculta $Y_j^{(1)}$, são calculados os resultados da camada de saída $Y_j^{(2)}$ através das Equações (2.17) e (2.18). Estes resultados representam a classificação gerada pela *MLP*.

$$I_j^{(2)} = \sum_{i=0}^{N} W_{ji}^{(2)} \cdot Y_i^{(1)} \tag{2.17}$$

$$Y_j^{(2)} = \sigma(I_j^{(2)}) \tag{2.18}$$

Dentre os artigos da área que aplicam esse método em problemas de classificação e identificação, pode-se citar o trabalho desenvolvido em (SINGH; DE, 2017), em que os autores utilizam o *MLP* em conjunto com a otimização por algoritmos genéticos (*GA*, do inglês *Genetic Algorithms*) na detecção de ataques *DDoS* na camada de aplicação. O algoritmo foi desenvolvido com base na análise em campos de pacotes recebidos, tais como a quantidade de endereços *IP* durante um intervalo de tempo, mapeamento de número de portas e tamanho dos pacotes ingressantes. Utilizando *MLP* e *GA*, a base de dados é avaliada de forma binária (ataques ou usuários normais). Resultados experimentais mostraram que a abordagem proposta atingiu altas taxas de eficiência na detecção de ataques *DDoS*.

Já em (FLORENCIO et al., 2018), os autores apresentam uma implementação de *MLP* em um sistema embarcado Arduino, objetivando a detecção de intrusões. Os autores destacam que a detecção de intrusões em tempo real utilizando dispositivos de baixa potência é um grande desafio para a comunidade de pesquisa em *IoT*. O modelo *MLP* foi treinado através da base de dados *NLS-KDD for Weka*, uma versão modificada da base de dados *NLS-KDD* contendo menor quantidade de dimensões. Com base nos resultados obtidos, os autores afirmam a viabilidade de se utilizar dispositivos de baixa potência (tal como o Arduino) como *IDS*s, bem como o baixo custo computacional da aplicação de redes *MLP*.

2.5 *Deep Learning - DL*

Dentre as diferentes abordagens de classificação por meio de redes neurais disponíveis, técnicas de Aprendizagem Profunda (*DL*) vêm se tornando cada vez mais populares nos últimos anos (WANG et al., 2020) (TEDJOPURNOMO et al., 2020) (YIN et al., 2020).

Segundo Panteleev, Gao e Jia (2018), *Deep Learning* é um aprofundamento da técnica de aprendizagem de máquina, em que arquiteturas de redes neurais mais complexas são utilizadas para resolver problemas de classificação com ampla gama de dados de treino disponíveis.

Conforme definido por Chollet (2017), o termo "profundo" dessa classe de algoritmos se refere à ideia de camadas sucessivas de representação. Assim, a profundidade de uma rede

neural é definida pela quantidade de camadas utilizadas na composição de sua arquitetura. Métodos de *DL* normalmente utilizam três ou mais camadas de representação, enquanto métodos de aprendizagem "rasa" (*shallow learning*), como o *MLP*, utilizam apenas uma ou duas camadas no processo de aprendizado.

Em (LIU et al., 2019a), os autores destacam que a principal vantagem de métodos de aprendizagem profunda é a ausência da necessidade de extração/seleção manual de atributos (*feature extraction*). Em outras palavras, essas técnicas são capazes de encontrar padrões por conta própria dentro de conjuntos de dados massivos durante o processo de treinamento. Essa característica aumenta drasticamente a eficiência dos processos de classificação, uma vez que padrões complexos, muitas vezes ocultos do olhar humano, podem ser extraídos do conjunto de dados.

Diversos trabalhos têm se baseado em métodos *DL* em tarefas de classificação, detecção de anomalias e ataques. Em (KAO; JIANG, 2019), os autores propuseram um *framework* para detecção de anomalias em séries temporais uni-variadas. Buscando atingir este objetivo, eles primeiramente dividiram os dados em três classes, que são séries temporais estacionárias, periódicas, e não-estacionárias. Então, diferentes métodos estatísticos e de *DL* foram aplicados nesses dados separados para realizar detecção de anomalias. Os resultados apontaram que o *framework* proposto obteve bons resultados em comparação com métodos relacionados. De forma similar, em (QIN; CHEN; LIN, 2018), os autores propuseram o uso de redes *Long Short Term Memory* (*LSTM*) na detecção de anomalias em redes *IP*. Os resultados obtidos mostraram promissoras taxas de precisão e *recall*, demonstrando a eficiência do método em problemas de classificação.

Em (HATCHER; YU, 2018) os autores apresentam um *Survey* sobre o estado da arte da utilização de técnicas de *Deep Learning* em diversos campos de estudo. Segundo os autores, essas técnicas já possuem aplicações consolidadas em diversas áreas, tais como reconhecimento de imagens e vídeos, processamento de áudio, análise textual, robótica, diagnósticos médicos, biologia computacional, previsões econômicas e financeiras e *cyber* segurança. Entretanto, ainda segundo os autores, o potencial de aplicação de técnicas *Deep Learning* é grande, com aplicações emergentes em áreas como o gerenciamento de redes de computadores, estratégias de otimização e aprendizado distribuído de *IoT*.

Embora *Deep Learning* seja uma sub-área dos métodos de aprendizagem de máquina, diversas técnicas se enquadram nessa classificação. Em (POUYANFAR et al., 2018), os autores descrevem algumas dessas técnicas, destacando suas principais características. Segundo os autores, Redes Neurais Recursivas (*RvNN*, do inglês *Recursive Neural Networks*) utilizam uma estrutura em árvore eficiente para problemas de processamento de linguagem natural (SOCHER et al., 2011), Redes Neurais Recorrentes (*RNN*, do inglês *Recurrent Neural Networks*) são eficazes para informações sequenciais e comumente aplicados em problemas de processamento de linguagem natural e processamento de fala (LI; WU, 2015) e Redes Neurais Convolucionais (*CNN*), originalmente utilizadas no reconheci-

mento de imagens, tiveram sua aplicação estendida para as áreas de visão computacional, processamento de fala e processamento de linguagem natural (ABDEL-HAMID et al., 2014).

Dentre os campos de pesquisa emergentes na área, Hatcher e Yu (2018) destacam o gerenciamento e controle de redes, campo que ainda conta com poucos trabalhos divulgados. Segundo os autores, redes *SDN* estão dentre as principais soluções propostas para resolver o problema da crescente complexidade e volume de tráfego, provendo ferramentas importantes para resolver os problemas de redes do futuro. Entretanto, características como seu gerenciamento e segurança são altamente dependentes da otimização de serviços e dispositivos físicos, área na qual técnicas de *Deep Learning* possuem aplicação viável devido à sua eficiência em aprender o comportamento da rede e seus usuários.

2.5.1 *Convolutional Neural Network - CNN*

Durante o desenvolvimento deste trabalho, diferentes técnicas de aprendizagem profunda foram avaliadas como alternativas para a detecção de ataques. O primeiro destes métodos foram as Redes Neurais Convolucionais (*CNN*). Como descrito por Chollet (2017), uma diferença fundamental entre uma camada totalmente conectada (utilizada em redes *MLP*, por exemplo) e uma camada convolucional é que a primeira aprende padrões globais em seu espaço de características de entrada (*input feature space*), enquanto a segunda é capaz de aprender padrões locais. Como as redes *CNN* são comumente aplicadas em ambientes de processamento de imagem, elas podem aprender padrões locais da imagem, o que melhora significativamente a acurácia de problemas de classificação.

Essa precisão é possível por conta de operações de convolução que compõem as redes *CNN*. Uma convolução é uma operação entre duas funções que produzem uma terceira, a qual expressa como o formato da primeira é modificado pela segunda (BUDUMA; LOCASCIO, 2017). Como descrito por Buduma e Locascio (2017), convoluções operam sobre tensores 3D chamados mapas de características (*feature maps*). Em problemas de classificação de imagem, por exemplo, esses mapas descrevem duas dimensões espaciais (largura e altura) e um eixo de canal (para imagens *RGB*, a dimensão do canal é 3, enquanto para imagens em preto e branco, essa dimensão é 1). Para convolucionar essa entrada, um filtro (matriz aleatoriamente inicializada, também conhecida como extrator de características ou *kernel*) é aplicado através de produtos ponto a ponto, visando a extração de padrões locais. Este filtro funciona como uma janela deslizante, realizando produtos ponto a ponto em todas as posições únicas em que ele pode ser inserido na imagem, codificando características específicas da entrada.

Entretanto, o tráfego de fluxos *IP* não é representado como uma imagem, a qual é descrita por dados bidimensionais (largura e altura), mas sim como série temporal. Dessa forma, uma variação do modelo tradicional de convolução foi utilizada no desenvolvi-

mento deste trabalho, denominada *1D-CNN* (CHOLLET, 2017). Em uma comparação direta, esse modelo funciona com as mesmas estruturas e funções, mas através de dados unidimensionais (séries temporais), conforme ilustrado pela Figura 2.

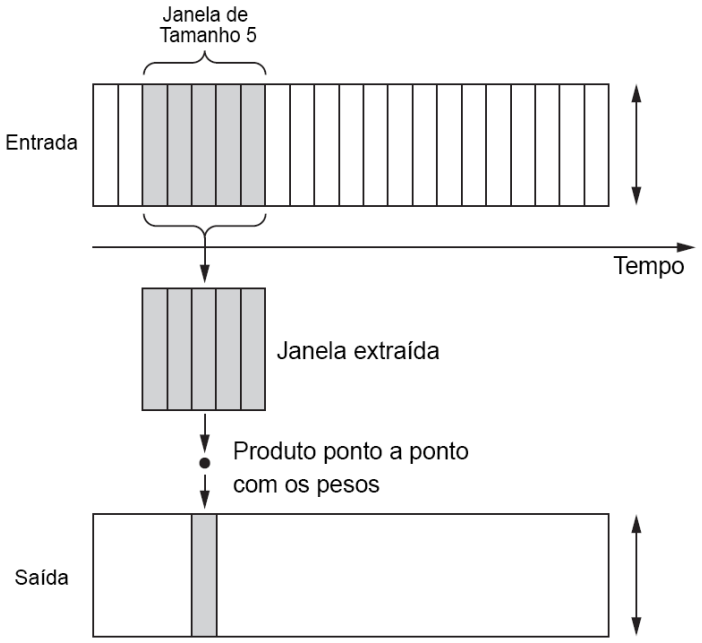


Figura 2 – Exemplo de convolução em 1 dimensão. (CHOLLET, 2017)

Como ilustrado, este filtro "desliza" sobre todas as posições possíveis do vetor de entrada, convolucionando (produto interno com o filtro) os dados e extraindo características locais. A Equação (2.19) ilustra um exemplo de uma convolução unidimensional, com *kernel* de tamanho 2.

$$[1 \quad 2 \quad 5] * [0 \quad 3] = [6 \quad 15] \tag{2.19}$$

Na referida equação, o vetor $[1 \quad 2 \quad 5]$ representa os dados de entrada, $[0 \quad 3]$ um filtro, e $[6 \quad 15]$ um vetor de convolução, gerado a partir do produto interno entre os dados de entrada e o filtro. Com o tamanho do *kernel* definido como 2, o produto interno com o filtro é realizado em todas as posições possíveis de janelas de 2 elementos com relação ao vetor de entrada. Assim, o primeiro elemento do vetor de convolução (6) é calculado pelo produto interno entre $[1 \quad 2]$ e $[0 \quad 3]$, enquanto o segundo (15) é resultante do produto interno entre $[2 \quad 5]$ e $[0 \quad 3]$.

Como uma rede neural profunda, são geradas dezenas de filtros diferentes para abstrair todas as informações relevantes da série temporal de entrada. A aprendizagem nessas redes se dá pelo treinamento supervisionado, ou seja, em que dados previamente rotulados são submetidos para ajuste da *CNN*. Com base nos dados de treinamento, a aprendizagem se dá pela distribuição de pesos sinápticos, de modo que filtros mais relevantes na

classificação do problema recebam pesos mais significativos, enquanto os menos relevantes são preteridos através da atribuição de pesos baixos (BUDUMA; LOCASCIO, 2017).

A Figura 3 ilustra a arquitetura da *CNN* implementada neste trabalho.

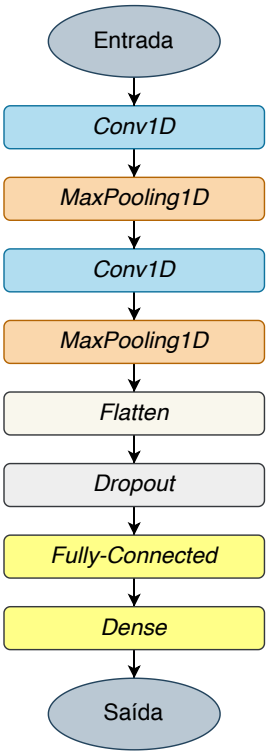


Figura 3 – Arquitetura da *CNN*.

Como observado, a arquitetura da *CNN* é composta pelo empilhamento de duas camadas *Conv1D*, intercaladas com duas camadas *MaxPooling1D*. As camadas *MaxPooling1D* são responsáveis por reduzir a dimensionalidade das convoluções, extraindo as características mais representativas de cada filtro e reduzindo o custo computacional do modelo. Essas camadas são precedidas de uma camada *Flatten*, responsável por converter a saída tridimensional das camadas anteriores em entrada bidimensional para as posteriores. Em seguida, é aplicada uma camada *Dropout* pretendendo evitar a ocorrência de *overfitting*, uma vez que redes *CNN* tendem a convergir rapidamente a uma solução. Por fim, uma camada totalmente conectada (*fully-connected layer*), responsável por classificar os filtros gerados, e uma camada densa, a qual realiza o processo de classificação geral, são aplicadas, gerando a resposta final do modelo.

Dentre os artigos que utilizam o *CNN* em soluções de classificação, pode-se destacar o trabalho de Wang, An e Huang (2018), em que foi proposta uma abordagem na qual dados de fluxos *IP* não processados (resultado de exportação direta do controlador, sem exclusão de *features*, por exemplo) são representados como uma imagem, a qual é aplicada a *CNN* para classificação e identificação de tráfego malicioso. Os autores atingiram bons resultados na detecção de diferentes eventos maliciosos. Embora essa representação seja uma abordagem promissora utilizando *CNN*, a utilização de fluxos não processados no

treinamento pode ocasionar enviesamento, uma vez que o método pode aprender que um endereço *IP* específico está relacionado a um comportamento malicioso.

Liu et al. (2019a), por sua vez, propuseram duas abordagens de classificação utilizando *payload*, as quais são baseadas em *CNN* e *RNN*. Esses métodos foram aplicados na detecção de ataques, e os autores destacaram sua capacidade em aprender representações de dados sem a realização de engenharia de características (*feature engineering*) nos dados originais. Os autores compararam os métodos apresentados com diferentes abordagens, atingindo resultados satisfatórios em testes realizados por meio da base de dados DARPA 1998.

2.5.2 *Gated Recurrent Units - GRU*

Dentre as diferentes abordagens de aprendizagem profunda, redes neurais recorrentes são particularmente promissoras no campo de detecção de anomalias e ataques. Diferentemente das redes sem retroalimentação (*feedforward*), tais como redes neurais densas (*DNN*, do inglês *Dense Neural Network*), ou redes *CNN*, as *RNN* possuem memória, isto é, consideram informações do passado no processo de predição/classificação (CANIZO et al., 2019). Essa memória representa uma característica importante na detecção de anomalias, uma vez que o estado da rede antes da ocorrência de um ataque (fluxos *IP* analisados anteriormente) podem ser usados em conjunto com fluxos atuais na geração de alarmes (CANIZO et al., 2019) (CHO et al., 2014).

Entretanto, redes *RNN* possuem uma limitação conhecida como problema da dissipação do gradiente (*vanishing gradient*). Embora essas redes teoricamente sejam capazes de reter informações acerca de entradas de dados analisadas muitos intervalos de tempo antes, na prática, redes *RNN* são incapazes de aprender dependências de longo prazo (HE; DROPPO, 2016). De forma resumida, isso ocorre porque operações sucessivas em dados de longo prazo gradualmente reduzem sua significância. Assim, quanto mais profunda a análise, menos significativos se tornam esses dados para influenciar no resultado da rede (BENGIO; SIMARD; FRASCONI, 1994).

Diferentes abordagens foram propostas para resolver este problema, e a mais comumente utilizada na literatura é conhecida como *LSTM* (HOCHREITER; SCHMIDHUBER, 1997) (NOVAES et al., 2020). Este método implementa uma série de mecanismos denominados "portões" (*gates*), os quais são capazes de regular as taxas de aprendizado e esquecimento da rede neural, garantindo que dados de longo prazo mantenham sua influência sobre classificações atuais. Cho et al. (2014) propuseram uma versão modificada do *LSTM*, denominada *Gated Recurrent Units* (*GRU*), a qual resume o funcionamento de *LSTM*s por meio da redução do número de portões utilizados, enquanto mantém a relevância de memórias de longo prazo (CHO et al., 2014).

Redes *GRU*, assim como *LSTM*, funcionam através da utilização de portões, os quais

são diferentes redes neurais que decidem quais informações devem ser esquecidas ou retidas. Redes *GRU* operam por meio de dois portões, o de Atualização (*Update*) e o de Redefinição (*Reset*). O primeiro é responsável por definir quais informações referentes a novas entradas serão esquecidas e quais novas informações serão acrescentadas na memória. Em contrapartida, o segundo portão descreve a quantidade de memória de longo prazo (dados do passado) que será esquecida. A Figura 4 ilustra o funcionamento de uma célula *GRU*, bem como seus portões.

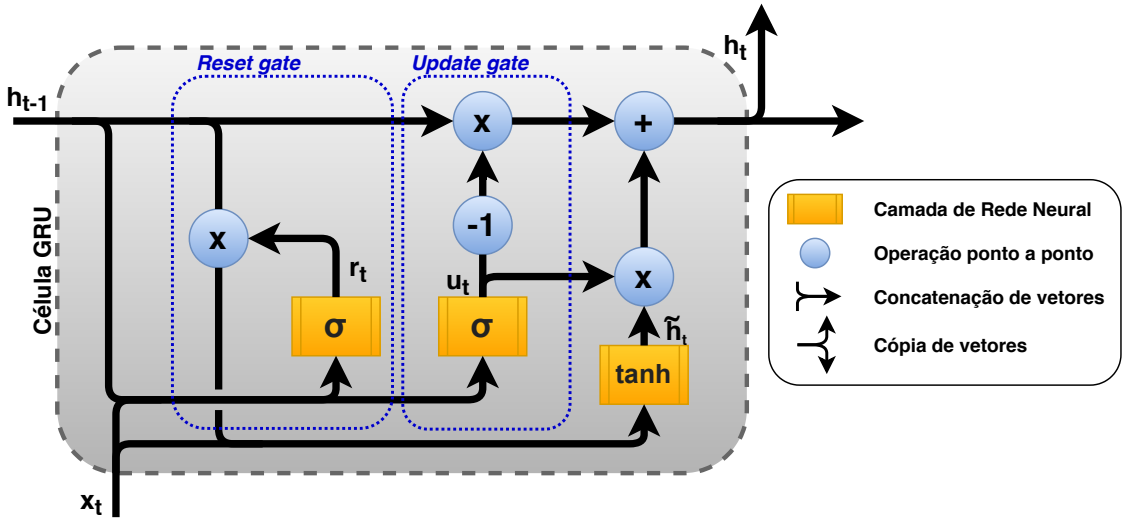


Figura 4 – Representação de uma célula *GRU*

Conforme apresentado pela Figura 4, h_{t-1} descreve o estado oculto (*hidden state* ou memória) do intervalo de tempo $t-1$, enquanto x_t e h_t representam dados de entrada e estado oculto de saída no intervalo t, respectivamente.

Assim como redes *CNN*, as redes *GRU* são mecanismos de aprendizagem supervisionada, as quais necessitam de dados históricos rotulados em seu processo de treinamento. Pode-se observar através da Figura 4 a presença de 2 *gates* que operam por meio de função de ativação sigmoidal (σ), que são neurais internas à célula *GRU*. Essas redes são utilizadas para definir quais informações serão esquecidas ou mantidas. A referida função de ativação é aplicada para simplificar este processo, uma vez que ela normaliza os valores de saída entre 0 e 1. Assim, qualquer valor multiplicado por 0 será esquecido, enquanto valores multiplicados por 1 serão mantidos.

Para se calcular o valor de h_t, a célula inicia concatenando h_{t-1} e x_t, e então submetendo o vetor resultante aos portões *Reset* e *Uptade*, obtendo seus resultados r_t e u_t por meio das Equações (2.20) e (2.21), respectivamente.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2.20}$$

$$u_t = \sigma(W_u \cdot [h_{t-1}, x_t] + b_u) \tag{2.21}$$

Nas referidas equações, W_r e W_u representam as matrizes de pesos das redes neurais, enquanto b_r e b_u são os vetores de *bias* das redes neurais. Assim, uma multiplicação ponto a ponto é realizada entre r_t e h_{t-1}, e o resultado é concatenado com x_t e submetido a uma terceira rede neural, com uma função de ativação tangente hiperbólica ($tanh$) dessa vez. O uso da $tanh$ normaliza os dados entre -1 e 1, regulando a saída da rede neural e prevenindo que os dados sejam super ou sub estimados entre iterações. A saída $\tilde{h_t}$ dessa rede neural é computada através da Equação (2.22).

$$\tilde{h_t} = tanh(W_o \cdot [r_t \cdot h_{t-1}, x_t] + b_o) \tag{2.22}$$

Na Equação (2.22), W_o e b_o representam a matriz de pesos e o vetor de *bias* na rede neural, respectivamente. A saída do portão *Update* é usada para duas situações. Na primeira, ela decide qual parte da nova informação será adicionada, multiplicando sua saída com $\tilde{h_t}$. Na segunda, ela determina quais informações descartar, realizando uma multiplicação ponto a ponto de h_{t-1} com $1 - u_t$. Finalmente, uma adição ponto a ponto é realizada entre essas duas saídas, gerando o resultado da célula *GRU*, o estado oculto (*hidden state*) h_t. A Equação (2.23) descreve essa situação.

$$h_t = (1 - u_t) \cdot h_{t-1} + u_t \cdot \tilde{h_t} \tag{2.23}$$

Esse processo descreve o funcionamento de uma única célula *GRU*, e a profundidade da rede neural se dá pela representação por meio de diversas células, utilizadas para ajustar a classificação ao problema proposto.

A Figura 5 apresenta a arquitetura *GRU* utilizada neste trabalho.

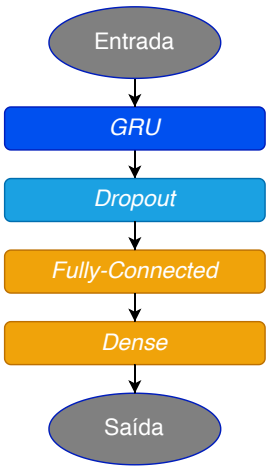


Figura 5 – Arquitetura da *GRU*.

Como observado a arquitetura da rede é composta de uma camada *GRU*, responsável por realizar o processo de classificação inicial. Em seguida, uma camada *Dropout* é adicionada, visando evitar a ocorrência de *overfitting*. Por fim, é adicionada uma camada totalmente conectada (*fully-connected layer*), capaz de realizar uma classificação

global dos resultados gerados pelas células *GRU*, e uma camada densa, responsável pela classificação final dos dados avaliados.

Dentre os trabalhos da área que utilizam o *GRU* como método de detecção, pode-se destacar (LAVROVA; ZEGZHDA; YARMAK, 2019). Nesse trabalho, os autores propuseram uma abordagem de detecção de falhas de segurança em sistemas de controle de processos automatizados (*APCS*, do inglês *automated process control systems*). Tal abordagem consiste na previsão de séries temporais de múltiplas variáveis formadas dos valores dos parâmetros de operação nos dispositivos *end system*. Os autores aplicaram *GRU* para a realização de tal predição, atingindo bons resultados em ataques simulados, os quais, em sua maioria, foram detectados em um estágio inicial.

Em (LIU et al., 2019b), os autores apresentam uma abordagem de detecção de anomalias para registros de rede (*logs*), utilizando *GRU* e *Support Vector Domain Description* (*SVDD*). Inicialmente, os autores aplicam o método *Principal Component Analysis* (*PCA*) para reduzir a dimensionalidade das bases de dados utilizadas e extrair atributos significativos. Posteriormente, as bases de dados processadas são treinadas por meio do modelo de classificação *GRU-SVDD*. Resultados de experimentos em bases de dados clássicas do *KDD Cup* 99 mostraram que o método proposto foi mais eficiente que os algoritmos clássicos *GRU-MLP* e *LSTM*.

Neste trabalho são propostos e avaliados diferentes métodos de detecção, baseados em aprendizagem de máquina e aprendizagem profunda. Além disso, é proposta uma abordagem baseada em Teoria de Jogos aplicada à mitigação de ataques *DDoS*. As abordagens utilizadas no desenvolvimento deste trabalho são amplamente referenciadas na literatura, o que demonstra a eficiência destes métodos em diferentes cenários de aplicação. Com isso, este trabalho busca avaliar a aplicação dessas técnicas em um sistema de defesa contra ataques *DDoS* e de intrusão em ambientes *SDN*. O próximo capítulo descreve o processo de funcionamento do sistema de defesa proposto, desde a extração de dados à detecção de ataques e geração de políticas de mitigação.

3 Sistema proposto

Neste capítulo é descrito o funcionamento geral do sistema de defesa proposto. Trata-se de um modelo baseado na análise de diferentes características (dimensões ou *features*) de fluxos *IP*, coletados através do protocolo OpenFlow (ONF, 2015). Seu objetivo é executar a detecção de ataques *DDoS* e de intrusão, bem como mitigar o impacto desses ataques em ambientes *SDN*.

Pretendendo o desenvolvimento de um sistema factível e eficaz, este foi sendo desenvolvido ao longo de quatro anos, e os resultados obtidos foram publicados em quatro artigos científicos. Durante o desenvolvimento desses trabalhos, o sistema foi sofrendo modificações, tais como a diminuição gradativa do intervalo de coleta de dados (objetivando agilizar a resposta do sistema frente a ataques detectados) e a avaliação de diferentes métodos de detecção (buscando classificações mais precisas e com baixo tempo de resposta).

As seções a seguir descrevem o funcionamento do sistema proposto, sua divisão modular e as abordagens propostas em cada um dos trabalhos desenvolvidos.

3.1 Organização do sistema

O sistema proposto divide-se em três módulos principais, que são i) Módulo de Detecção, ii) Módulo de Identificação (ou informação) e iii) Módulo de Mitigação, conforme ilustrado pela Figura 6.

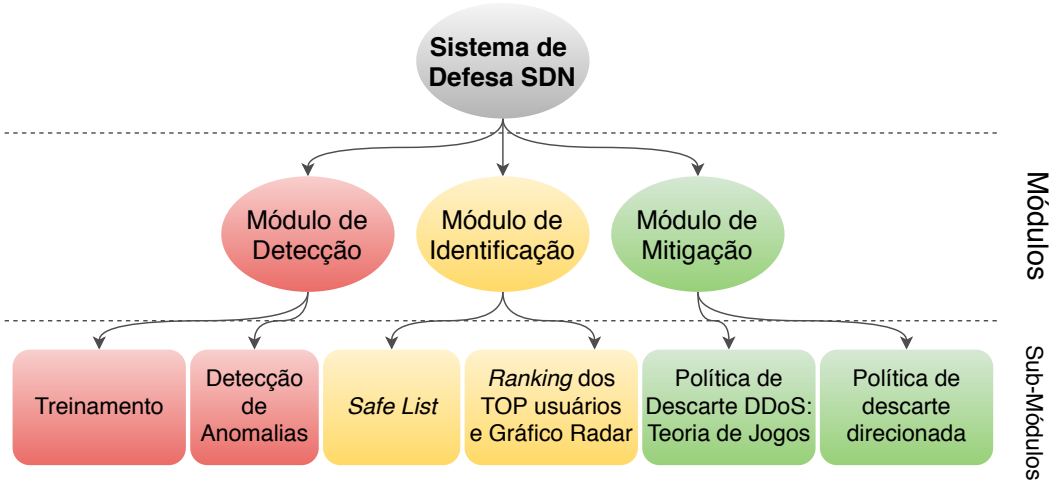


Figura 6 – Organização modular do sistema proposto.

No Módulo de Detecção, uma análise multi-dimensional de fluxos *IP* é realizada com base no gerenciamento de diferentes características, tais como: bits/s, pacotes/s, endereços *IP* de origem e destino e portas de origem e destino, entre outras. Diferentes méto-

dos de detecção, selecionados pelo gerente do sistema, podem executar as funções deste módulo, tais como: o *HWDS* (ASSIS; RODRIGUES; PROENÇA, 2013) (ASSIS; RODRIGUES; PROENçA JR., 2014), um modelo estatístico de previsão de séries temporais aplicado à detecção de anomalias; o *MLP* (HAYKIN, 2011), um modelo de aprendizagem de máquina (rede neural); ou Redes *CNN* (CHOLLET, 2017) e *GRU* (CHO et al., 2014), modelos de aprendizagem profunda (*DL*), aplicados em problemas de classificação, regressão, entre outros.

O Módulo de Identificação armazena dados relevantes, tais como endereços *IP* de origem e destino, acerca dos fluxos classificados como anômalos (endereços mais frequentes no intervalo de tempo avaliado, ou "TOP usuários") e não-anômalos (ASSIS; RODRIGUES; PROENçA JR., 2014). Com isso, uma lista de endereços *IP* legítimos (não-maliciosos), aqui denominada *safe list*, pode ser criada, o que melhora consideravelmente os resultados obtidos pelo processo de mitigação. Nos últimos trabalhos (Apêndices C e D), este módulo é incorporado ao Módulo de Mitigação, simplificando o modelo geral. Além disso, as informações dos TOP usuários pode ser utilizada por métodos de detecção incapazes de identificar o tipo do ataque por conta própria.

No Módulo de Mitigação, por sua vez, são geradas políticas de descarte de pacotes visando neutralizar a anomalia detectada, trazendo a rede de volta para seu estado normal. Essas políticas de descarte são passadas juntamente com a *safe list* gerada ao controlador *SDN* que, por sua vez, as aplica na rede. Para executar essa tarefa é proposto um modelo baseado em Teoria de Jogos objetivando a geração de uma taxa eficiente de descarte de pacotes para a mitigação de ataques *DDoS*. Com relação a outras formas de ataques, tais como *PS*, são utilizadas políticas de descarte direcionado.

3.2 Funcionamento do sistema

Conforme descrito nas seções anteriores, o modelo de operação do sistema proposto foi sofrendo modificações conforme novos trabalhos foram desenvolvidos, embora o fluxo de funcionamento geral tenha se mantido o mesmo. Dessa forma, a Figura 7 descreve esse processo por meio de um fluxograma, resumindo as diferenças e similaridades relativas aos quatro artigos desenvolvidos nesta tese em uma única imagem.

As seções a seguir descrevem separadamente cada etapa do processo de defesa do sistema proposto, traçando um paralelo entre os artigos desenvolvidos de modo a evidenciar a evolução ao longo do desenvolvimento desta tese.

3.2.1 Dispositivo de aplicação

A viabilidade do sistema proposto foi testada em dois diferentes dispositivos: em um dispositivo de borda, e no controlador central da rede.

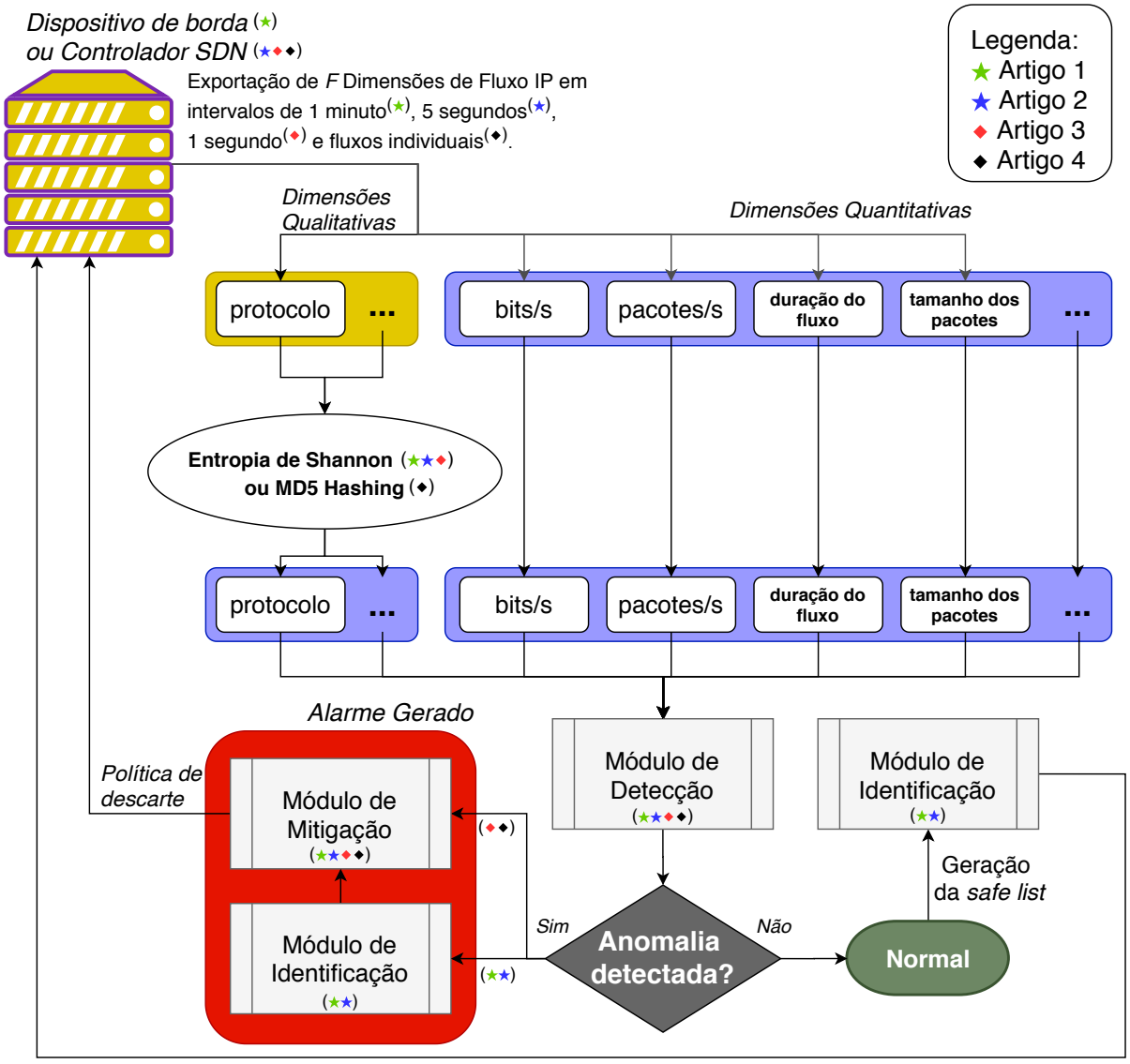


Figura 7 – Fluxograma de funcionamento do sistema, resumindo as diferenças e similaridades relativas aos quatro artigos desenvolvidos nesta tese.

No desenvolvimento do Artigo 1 (Apêndice A), o sistema de defesa foi inicialmente aplicado em um dispositivo de borda da rede *SDN*, conforme ilustrado pela Figura 8.

Como observado, o sistema de defesa proposto age como um intermediário, trabalhando em conjunto com um *firewall* para proteger o controlador contra ataques externos. O tráfego que entra na rede é avaliado e, caso um ataque seja detectado, contramedidas são geradas para minimizar seu impacto.

Entretanto, essa organização não leva em conta a possibilidade de ataques internos, ou seja, de dispositivos componentes do ambiente *SDN*. Pretendendo aperfeiçoar sua capacidade de defesa, tornando a rede mais robusta, o sistema foi implementado diretamente dentro do controlador central *SDN* a partir do Artigo 2 (Apêndice B).

Por fim, no Artigo 3 (Apêndice C), o sistema foi implantado novamente dentro do controlador central, porém visando a detecção e mitigação de ataques internos (surgindo

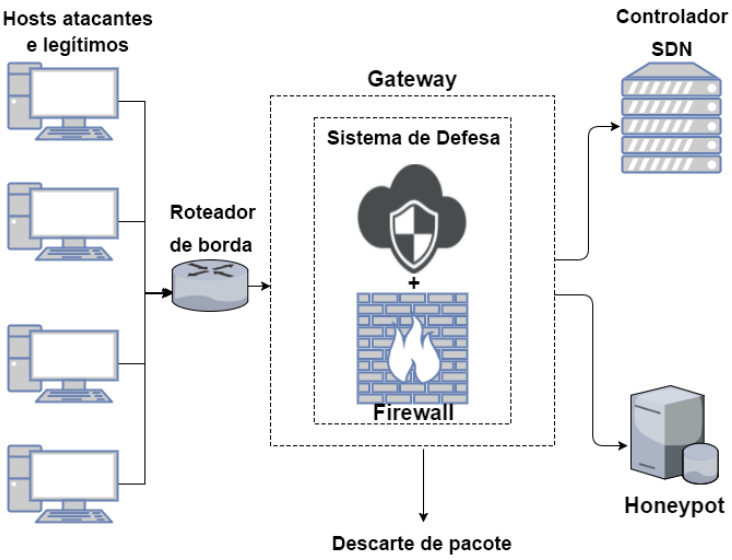


Figura 8 – Topologia da rede e organização do sistema de defesa no Artigo 1.

de *hosts* pertencentes ao ambiente *SDN*) contra alvos externos à rede. Embora esses ataques não tenham como alvo o controlador central, o alto volume de tráfego pode interferir em seu funcionamento, impactando no desempenho da rede *SDN* (ataque *DDoS* indireto). A Figura 9 ilustra esse cenário.

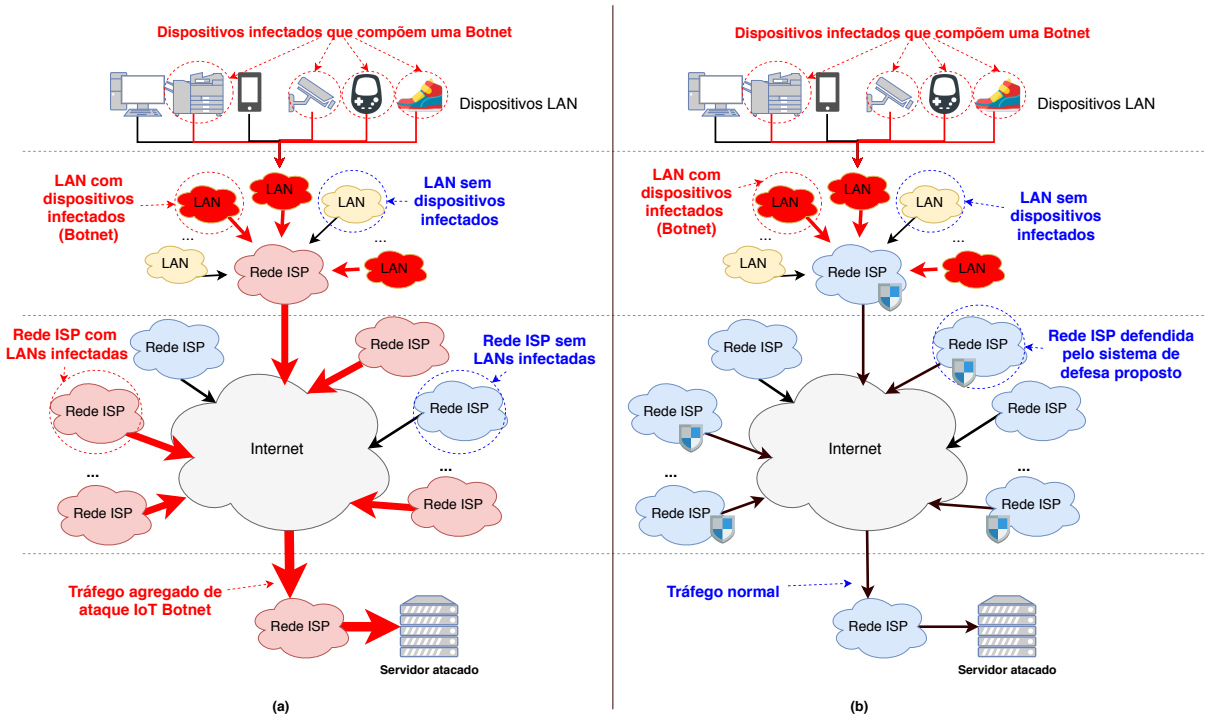


Figura 9 – Esquema de mitigação inversa utilizando Teoria de Jogos contra ataques *DDoS*, proposto no Artigo 3.

Além disso, conforme ilustrado pela Figura 9, a aplicação distribuída do sistema de defesa proposto em redes *SDN* de diferentes provedores de serviço de Internet (*ISP*, do

inglês *Internet Service Provider*) seria capaz de mitigar ataques *DDoS* em sua origem. Essa abordagem de mitigação inversa, fundamentada no paradigma "dividir e conquistar", é capaz de reduzir significativamente o impacto causado no servidor alvo.

3.2.2 Coleta de dados

Como ilustrado pela Figura 7, o controlador *SDN* realiza a exportação de fluxos *IP*, ou seja, a coleta dos dados utilizados na análise da rede. Por conta do volume de dados tratados, é comum que o processo de exportação ocorra por meio de coleta intervalada (CARVALHO et al., 2018) (BEREZIńSKI; JASIUL; SZPYRKA, 2015) (SUN et al., 2016), aqui denominada amostragem. Em outras palavras, a cada intervalo de tempo (minutos ou segundos) ocorre uma nova coleta de dados.

Por outro lado, quanto maior o tempo de amostragem, consequentemente maior o tempo que o sistema de defesa leva para detectar a presença de ataques, bem como para aplicar as políticas de mitigação. Seguindo essa linha de raciocínio, como a rede *SDN* depende do controlador central, ela se tornará inoperante por mais tempo, e a perda de dados também será maior.

Dessa forma, quanto menor o intervalo de análise, maiores são as chances de se detectar um ataque em seu início, reduzindo o impacto causado aos usuários finais da rede. Os intervalos de coleta de dados foram sendo reduzidos gradualmente nos trabalhos desenvolvidos, com coletas de um minuto, cinco segundos e um segundo nos Artigos 1, 2 e 3 (Apêndices A, B e C), respectivamente. Por fim, o Artigo 4 (Apêndice D) elimina a amostragem do processo, analisando fluxos individuais nos Módulos de Detecção e Mitigação de ataques.

3.2.3 Dimensões analisadas

A análise de fluxos *IP* fornece uma ampla gama de informações acerca das comunicações que ocorrem na rede. A utilização de múltiplas *features* (dimensões) significativas possibilita uma maior precisão para mecanismos de detecção de anomalias (MOLNAR; MOCZAR, 2011). Dessa forma, múltiplas dimensões de fluxos *IP* são utilizadas pelo sistema proposto, sendo exportadas pelo controlador central.

O método de detecção *HWDS*, utilizado no Artigo 1, utiliza sete diferentes dimensões para a detecção de ataques *DDoS*. São elas: bits/s, pacotes/s, fluxos/s, endereços *IP* de origem e destino e portas de origem e destino. Ainda no mesmo artigo, foram utilizadas 6 diferentes dimensões para a análise realizada por meio do método *Fuzzy-GADS*, retirando a *feature* "fluxos/s". Não havendo alteração significativa no desempenho dos métodos por meio da redução da dimensionalidade no Artigo 1, todos métodos utilizados na detecção do Artigo 2 passaram a utilizar 6 dimensões de fluxos *IP*.

Os Artigos 3 e 4 introduzem a utilização de métodos de aprendizagem profunda na detecção de ataques. Conforme descrito anteriormente (Seção 2.5), esses métodos utilizam uma grande quantidade de dados, deixando de lado a seleção manual de dimensões dos artigos anteriores, e passando a realizar essa seleção de forma automática. No Artigo 3, o primeiro cenário de testes utilizou 6 dimensões, enquanto o segundo utilizou 87 diferentes dimensões para a detecção de ataques *DDoS*. No Artigo 4, o primeiro cenário de testes (*DDoS*) utilizou 83 dimensões, enquanto o segundo (ataques de intrusão) utilizou 79 diferentes dimensões.

3.2.4 Conversão de dados qualitativos

As dimensões de fluxo *IP* exportadas podem ser classificadas em dois grupos: dados quantitativos e qualitativos. O primeiro grupo contém *features* com valores numéricos, tais como bits/s, pacotes/s, tamanho dos pacotes, entre outros. O segundo, por sua vez, possui *features* com informações não contáveis, como protocolo, endereço *IP* de origem e destino. Dimensões quantitativas podem ser diretamente aplicadas ao Módulo de Detecção, enquanto as qualitativas precisam antes passar por uma conversão.

Nos Artigos 1, 2 e 3, foi aplicada a entropia de Shannon (SHANNON, 2001), que permite a extração de informação relativa à concentração/dispersão dos dados nessas dimensões de fluxo. Assim, a *feature* "protocolo", por exemplo, tem como resultado um valor numérico que avalia se há uma concentração de algum protocolo específico no intervalo de tempo analisado.

Como o Artigo 4 introduz a análise de fluxos *IP* individualizados (sem amostragem), a utilização da entropia de Shannon se mostra inviável, uma vez que *features* contêm valores individuais. Dessa forma, a quantificação dessas informações se deu por meio da aplicação da técnica de *Hashing MD5*. Assim, seguindo o mesmo exemplo anterior, a dimensão "protocolo" terá seu valor (*TCP*, *UDP*, etc.) convertido em um código *hash*, possibilitando sua utilização pelos métodos do Módulo de Detecção. A motivação por trás da utilização do *hashing* ao invés de rótulos estáticos (que ocupariam menos memória) foi a generalização do processo, que pode ser aplicado a quaisquer dimensões qualitativas presentes na base de dados avaliada.

3.2.5 Módulo de Detecção

Os dados devidamente tratados no passo anterior chegam ao Módulo de Detecção, responsável por avaliar as informações recebidas visando a detecção de ataques, sejam eles de *DDoS* ou técnicas de intrusão.

Diferentes métodos de detecção podem executar as funções deste módulo. No Artigo 1, o *HWDS* (ASSIS; RODRIGUES; PROENçA JR., 2014), um modelo estatístico de previsão de séries temporais aplicado à detecção de anomalias, é utilizado. O desempenho

deste modelo é comparado ao do método de Lógica *Fuzzy* e algoritmos genéticos (*GA*, do inglês *Genetic Algorithms*), denominado *Fuzzy-GADS*, ambos sendo avaliados na detecção de ataques *DDoS*.

No Artigo 2, são propostos três diferentes métodos para a detecção de ataques *DDoS* e *PS*. O primeiro deles é uma rede *MLP*, modelo de rede neural de aprendizagem supervisionada (Seção 2.4.1). O segundo utiliza o método de Otimização por Enxame de Partículas (*PSO*, do inglês *Particle Swarm Optimization*) em uma abordagem de aprendizagem não-supervisionada, baseada em clusterização de dados e na desigualdade de Chebyshev para detecção de ataques. O terceiro, denominado *WaveDetect*, utiliza a Transformada Discreta de *Wavelet* (*DWT*, do inglês *Discrete Wavelet Transform*). Essa é uma técnica de aprendizagem não-supervisionada baseada em processamento de sinais que decompõe os dados de entrada em suas partes constituintes, utilizando suas frequências para caracterizar o tráfego de rede e detectar a presença de anomalias. Por fim, os métodos de detecção propostos são avaliados em comparação com outros três métodos da literatura, sendo eles: k-*means* (MüNZ; LI; CARLE, 2007), *kNN* (HAUTAMAKI; KARKKAINEN; FRANTI, 2004) e *SVM* (RAMASWAMY; RASTOGI; SHIM, 2000).

O Artigo 3, por sua vez, introduz a utilização de uma técnica de aprendizagem profunda, denominada Rede *CNN* (Seção 2.5.1). Esse é um modelo de aprendizagem supervisionada, amplamente aplicado a problemas de reconhecimento de imagens. O método foi implementado e comparado a outros três modelos: o *MLP*, o *dense MLP* (*D-MLP* ou *DNN*, do inglês *Dense Neural Network*) (ABDULHAMMED et al., 2019), uma versão de aprendizagem profunda do modelo *MLP* tradicional, e a Regressão Logística (*LR*, do inglês *Logistic Regression*) (CARVALHO et al., 2018).

Por fim, o Artigo 4 propõe a utilização do método *GRU* (Seção 2.5.2), modelo de aprendizagem profunda, supervisionado e recorrente, ou seja, que utiliza informações passadas no processo de classificação (modelo com memória). São comparados outros sete modelos de detecção neste trabalho, sendo eles: *D-MLP*, *CNN*, *SVM* (LEI, 2017), *LSTM* (QIN; CHEN; LIN, 2018), *LR*, *kNN* e *Gradient Descent* (*GD*) (WIJNHOVEN; WITH, 2010).

Após análise realizada pelo método de detecção, se um ataque/anomalia é detectado, então um alarme é gerado e os dados são submetidos ao Módulo de Identificação (Artigos 1 e 2) ou diretamente para o Módulo de Mitigação (Artigos 3 e 4).

3.2.6 Módulo de Identificação

O Módulo de Identificação é responsável por capturar informações relevantes acerca dos fluxos *IP* analisados de modo a contribuir e aprimorar os resultados obtidos pelo Módulo de Mitigação.

As informações capturadas, conforme descrito pela Figura 6, são organizadas em es-

truturas que auxiliam na identificação do tipo do ataque (em abordagens binárias, em que o tráfego é classificado como anômalo ou normal), sua origem, destino, entre outras informações. Conforme apresentado em (ASSIS; RODRIGUES; PROENçA JR., 2014), os dados de endereço *IP* de origem e destino, portas de origem e destino e protocolo são elencados por frequência de aparição (TOP usuários), o que possibilita identificar diferentes tipos de ataques. Como exemplo, pode-se citar uma situação com alta frequência de envio de pacotes de um mesmo endereço *IP* de origem para um único endereço *IP* e porta de destino, o que pode identificar um ataque do tipo *DoS*. Além disso, também pode ser gerado um gráfico de radar, que auxilia o administrador de rede a visualizar o comportamento de diferentes dimensões de fluxo *IP* de forma global.

Outra tarefa desempenhada por este módulo é a manutenção de uma *safe list*, uma lista que contém os endereços *IP* de origem de fluxos considerados legítimos (não-maliciosos) relativos aos últimos cinco minutos. Essa lista é utilizada para prevenir que pacotes de usuários legítimos sejam descartados, melhorando consideravelmente os resultados obtidos pelo processo de mitigação. A *safe list* é atualizada sempre que um intervalo avaliado for classificado como normal, sendo direcionada para o controlador *SDN*. O intervalo de tempo de 5 minutos foi selecionado por meio de testes empíricos, e a otimização deste intervalo será tema de trabalhos futuros.

Este módulo está presente nos Artigos 1 e 2, enquanto no Artigo 3 suas funções foram incorporadas ao Módulo de Mitigação de modo a simplificar a organização do sistema de defesa. Por utilizar uma abordagem de análise de fluxos *IP* isolados, o Artigo 4 não necessita dessas informações, uma vez que fluxos detectados como anômalos podem ser diretamente descartados.

3.2.7 Módulo de Mitigação

Caso uma anomalia ou ataque seja detectado, o Módulo de Mitigação irá atuar de modo a gerar políticas de descarte de pacotes, com o objetivo de minimizar o impacto causado na rede.

Este módulo recebe informações do Módulo de Identificação (Artigos 1 e 2) ou diretamente do Módulo de Detecção (Artigos 3 e 4). Conforme descrito no início deste capítulo, o sistema proposto visa a proteção do controlador *SDN* contra ataques *DDoS* e de intrusão. Dessa forma, são duas as abordagens de mitigação possíveis.

A primeira delas pretende mitigar ataques *DDoS* em que, por sua característica distribuída, não é possível identificar com clareza os dispositivos de origem do ataque. Para executar essa tarefa é proposto um modelo baseado em Teoria de Jogos (Seção 2.3) objetivando a geração de uma taxa de descarte de pacotes para a mitigação de ataques *DDoS*. Esse modelo foi desenvolvido no Artigo 1, mas também está presente em todos os demais.

A segunda abordagem é utilizada em casos em que o ataque é gerado por um único

host (como o ataque *PS*, por exemplo). Dessa forma, o Módulo de Identificação é capaz de isolar o endereço *IP* de origem do atacante, possibilitando que uma política de descarte direcionado seja implementada pelo controlador *SDN*. Essa abordagem está presente nos Artigos 1, 2 e 4 (trabalhos que não focam apenas em ataques *DDoS*).

Após a definição da política de descarte, essas informações são enviadas para o controlador *SDN* que, por sua vez, realizará a implementação da mitigação do ataque.

Caso as políticas de descarte geradas não sejam eficientes, ou caso um novo ataque ocorra, o sistema proposto irá avaliar o estado da rede novamente no próximo intervalo de tempo (Artigos 1 a 3) ou próximo fluxo *IP* analisado (Artigo 4), gerando novas políticas de mitigação até que a rede volte ao seu estado normal.

4 Artigos Desenvolvidos

Este capítulo aborda individualmente cada um dos quatro artigos desenvolvidos que compõem a presente tese, três publicados e um em processo de avaliação. Embora a organização e operação gerais do sistema proposto se mantenham as mesmas através de todos os trabalhos desenvolvidos (conforme descrito na Seção 3), cada um desses artigos aborda aspectos específicos. Como resultado, o sistema de defesa pôde ser avaliado em diferentes perspectivas de ataques (*DDoS* e intrusão), tais como: ataques externos contra o controlador *SDN* (Seção 4.1); ataques internos e externos contra o controlador *SDN* (Seção 4.2); e ataques internos contra um servidor externo à rede *SDN* (Seção 4.3). Além disso, diferentes mecanismos de detecção foram avaliados, utilizando desde abordagens estatísticas, como o *HWDS* (seção 4.1), até modelos de aprendizagem de máquina e aprendizagem profunda (seções 4.2 a 4.4).

A Tabela 1 traça um paralelo comparativo entre os trabalhos desenvolvidos, evidenciando a evolução do modelo ao longo do desenvolvimento desta tese.

Nas seções a seguir os artigos são discutidos separadamente, sendo abordados seus objetivos e os modelos avaliados, bem como os resultados obtidos e as contribuições individuais de cada autor para seu desenvolvimento.

Tabela 1 – Comparativo entre os trabalhos desenvolvidos

	Artigo 1	**Artigo 2**	**Artigo 3**	**Artigo 4**
Intervalo de coleta e análise de dados	1 minuto	5 segundos	1 segundo	Fluxos individuais
Ataques considerados	*DoS* e *DDoS* (*UDP flood*)	*DDoS* (*UDP flood*) e *PS*	*DDoS* (12 tipos)	*DDoS* (12 tipos) e Ataques de Intrusão (14 tipos)
Métodos de detecção	*HWDS* e *Fuzzy-GADS*	*MLP*, *PSO*, *DWT*, k-*means*, *kNN* e *SVM*	*CNN*, *MLP*, *DNN*, *LR*	*GRU*, *DNN*, *CNN*, *LSTM*, *SVM*, *LR*, *kNN*, *GD*
Dimensões de fluxos *IP* analisadas	6 (*Fuzzy-GADS*) e 7 (*HWDS*)	6	6 (primeiro cenário de testes) e 87 (segundo cenário de testes)	83 (primeiro cenário de testes) e 79 (segundo cenário de testes)

Tabela 1 (continuação) - Comparativo entre os trabalhos desenvolvidos

	Artigo 1	**Artigo 2**	**Artigo 3**	**Artigo 4**
Mitigação	Teoria de Jogos (*DDoS*) e descarte direcionado (*DoS*)	Teoria de Jogos (*DDoS*) e descarte direcionado (*PS*)	Teoria de Jogos	Teoria de Jogos (*DDoS*) e descarte direcionado
Local de aplicação do sistema	Dispositivo de rede de borda da rede *SDN*	Controlador *SDN*	Controlador *SDN*	Controlador *SDN*
Escopo de defesa	Contra ataques externos direcionados ao controlador *SDN*	Contra ataques internos e externos direcionados ao controlador *SDN*	Contra ataques internos contra um servidor localizado fora da rede *SDN* (ataque *DDoS* indireto)	Contra ataques internos e externos direcionados ao controlador *SDN*
Journal	*IEEE Access* (ISSN: 2169-3536)	*IEEE Access* (ISSN: 2169-3536)	*Computers and Electrical Engineering Journal* (ISSN: 0045-7906)	Em avaliação
Classificação *Qualis*	A1	A1	A2	=========

4.1 Artigo 1 - *A Game Theoretical Based System Using Holt-Winters and Genetic Algorithm With Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks*

O primeiro trabalho desenvolvido, intitulado "*A Game Theoretical Based System Using Holt-Winters and Genetic Algorithm With Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks*", é um artigo de revista publicado pelo *IEEE Access* em 2017. Nesse artigo foram aplicados e comparados os métodos *HWDS* e *Fuzzy-GADS* na detecção de ataques Dos e *DDoS*. Nesse contexto, o sistema de proteção foi aplicado em um dispositivo de borda da rede *SDN*, o qual analisa fluxos *IP* em intervalos de um minuto, protegendo o controlador contra o tráfego externo que entra na rede.

Além disso, nesse trabalho o jogo utilizado no processo de mitigação de ataques *DDoS*

(*GT*) foi modelado e detalhado (Seção 2.3).

As contribuições de cada autor no desenvolvimento deste artigo foram: i) Marcos V. O. de Assis - Concepção do sistema de defesa (organização, funcionamento e localização) contra ataques externos, a proposição do intervalo de análise de fluxos *IP* de 1 minuto, a concepção e implementação do método *HWDS*, a modelagem e implementação da mitigação com Teoria de Jogos (*GT*), a geração de ataques *DoS* e *DDoS* para realização de testes, a execução de testes e a escrita; ii) Anderson H. Hamamoto - Descrição e implementação do modelo *Fuzzy-GADS* e revisão textual; iii) Taufik Abrão - Revisão conceitual e formal da modelagem do jogo de mitigação (*GT*); e iv) Mario L. Proença Jr. - Orientação e discussão do planejamento, organização bem como revisão dos resultados encontrados.

Para avaliar a performance do sistema proposto, foram gerados e aplicados cinco diferentes dias de testes, um contendo ataque *DoS* e outros quatro contendo ataques *DDoS* de diferentes intensidades. Os dados foram gerados utilizando o emulador de rede Mininet em conjunto com o controlador *SDN* FloodLight. As implementações foram realizadas utilizando *software* Matlab.

Como observado na Fig. 10, os resultados obtidos corroboram a eficiência dos dois métodos de detecção avaliados, os quais atingiram valores similares em métricas de detecção, tanto para *DoS* (experimento 5) quanto *DDoS* (experimentos 1 a 4). Ambos os métodos obtiveram taxas de acurácia superior a 98% na média, e o método *HWDS* apresentou maior taxa de detecção geral que o *Fuzzy-GADS*.

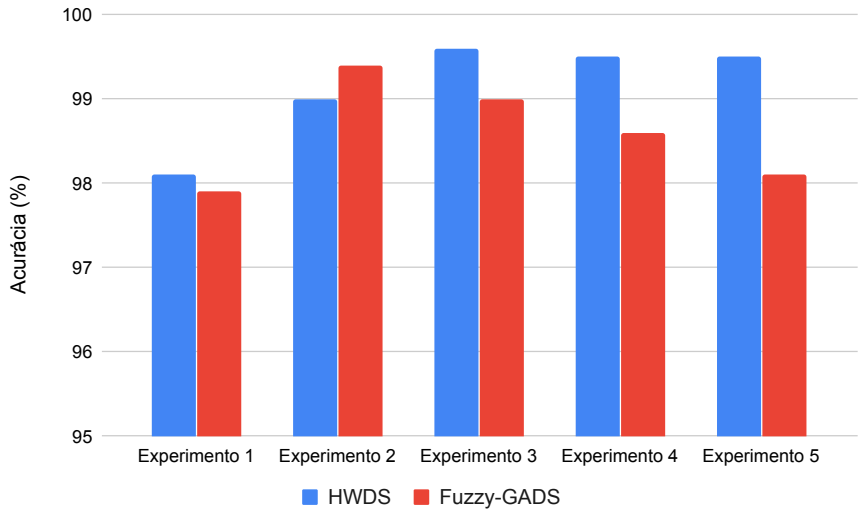


Figura 10 – Resultados de acurácia dos métodos *HWDS* e *Fuzzy-GADS* na detecção de ataques *DDoS* (Experimentos 1 a 4) e *DoS* (Experimento 5).

Entretanto, a diferença mais significativa entre ambos os métodos foi evidenciada na mitigação. O sistema atingiu melhores resultados utilizando *HWDS* em testes de performance levando em conta a mitigação de ataques *DDoS* (*GT*), como ilustrado pela Fig. 11.

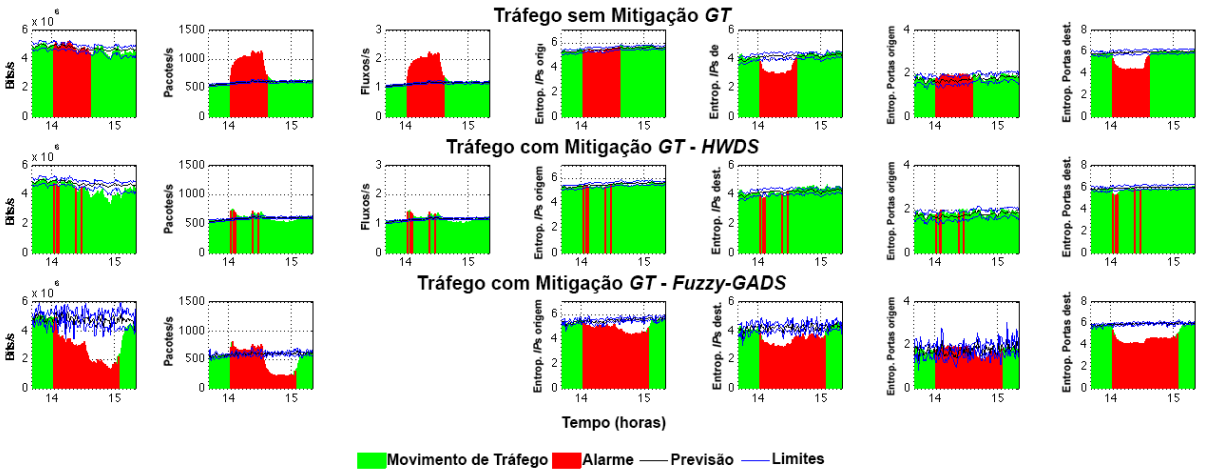


Figura 11 – Comparação entre o movimento de tráfego com e sem a abordagem de mitigação *GT* utilizando os métodos *HWDS* e *Fuzzy-GADS* com relação ao Experimento 3 (ataque *DDoS* com 2560 *hosts*).

Isso ocorre devido à disponibilidade do Módulo de Identificação como parte do método *HWDS* (ASSIS; RODRIGUES; PROENçA JR., 2014), o qual provê importantes informações sobre o ataque e a rede *SDN* para o Módulo de Mitigação. Por meio do uso dessas informações, foi possível a criação de uma *safe list*, contendo informações de *hosts* legítimos da rede. Com isso, esses *hosts* foram protegidos da política de descarte *GT*, conforme ilustrado pelas Fig. 12 e 13.

Como ilustrado pelas Fig. 12 e 13, a taxa de descarte de fluxos de usuários legítimos foi

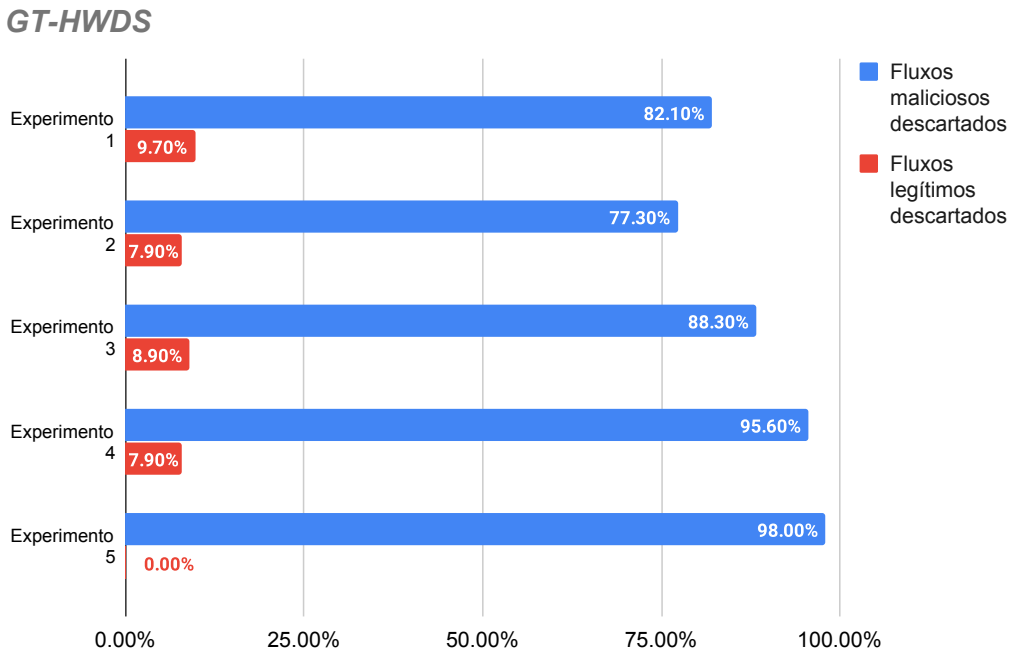


Figura 12 – Porcentagem de descarte de fluxos legítimos e maliciosos pelo método *HWDS*.

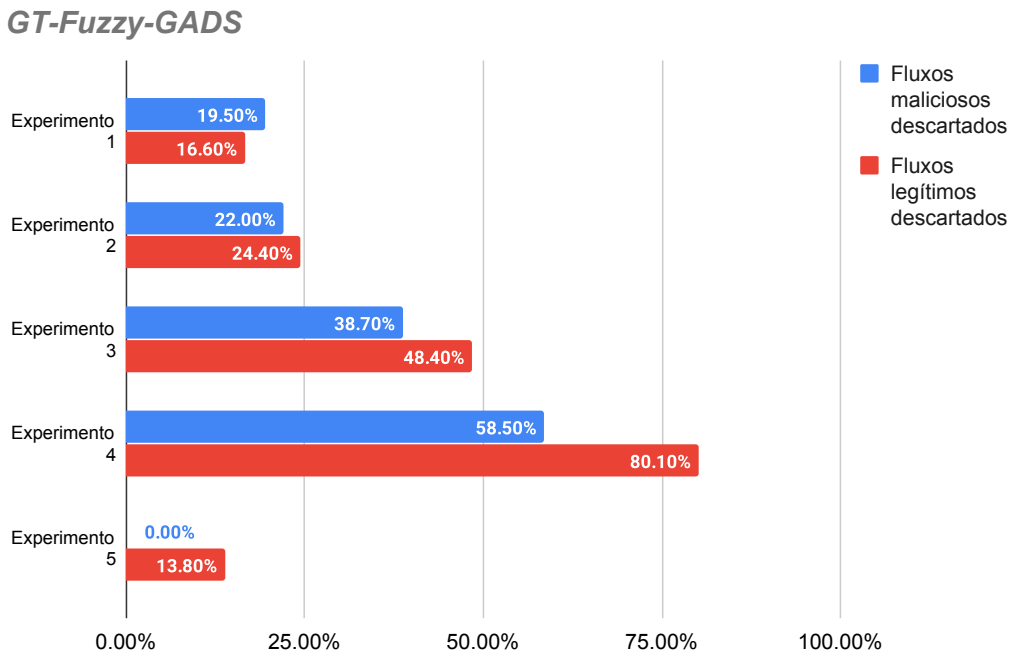


Figura 13 – Porcentagem de descarte de fluxos legítimos e maliciosos pelo método *Fuzzy-GADS*.

menor utilizando o método *HWDS*, o que implica diretamente na eficácia do processo de mitigação dos ataques. Além disso, o método *HWDS* também se saiu melhor na mitigação de ataques *DoS* utilizando política de descarte direcionado, uma vez que foi possível para ele a identificação do *IP* de origem do atacante.

Assim, pode-se concluir que o sistema proposto é um eficiente mecanismo de defesa para ambientes *SDN* contra ataques externos, impedindo o congestionamento do tráfego no controlador central por meio do uso de eficazes ações de mitigação. Além disso, é possível concluir que a abordagem de tomada de decisões *GT* pode ser utilizada como um Módulo de Mitigação independente. *GT* é capaz de garantir o funcionamento de redes *SDN* contra ataques *DDoS* independentemente do modelo de detecção aplicado. Entretanto, a *safe list* (lista de endereços *IP* de usuários legítimos) gerada pelo Módulo de Identificação do *HWDS* melhora os resultados da mitigação e, por este motivo, foi incorporada em versões posteriores do modelo de mitigação.

4.2 Artigo 2 - *Fast Defense System Against Attacks in Software Defined Networks*

O segundo artigo desenvolvido, intitulado "*Fast Defense System Against Attacks in Software Defined Networks*", um artigo de revista publicado no *IEEE Access* em 2018, expande os resultados obtidos no primeiro de diferentes maneiras. Primeiramente, foram

propostos e avaliados três outros modelos de detecção de anomalias aplicados contra ataques *DDoS* e *PS*, que são: *MLP*, *PSO* e *DWT*. Neste contexto, o sistema de proteção foi aplicado diretamente dentro do controlador *SDN*, o qual analisa as dimensões de fluxo *IP* em intervalos de cinco segundos, protegendo o controlador contra ataques tanto externos quanto internos.

O sistema de defesa *SDN* proposto é avaliado neste trabalho contra ataques *DDoS* e de escaneamento de portas (*PS*). Diferentemente de ataques *DDoS*, caracterizados como ataques de volume ou *flooding*, ataques *PS* são mais sutis e, consequentemente, de detecção menos óbvia. Para avaliar a eficiência do sistema proposto nessas condições, foram gerados fluxos *IP* relativos a dois dias através do emulador de redes Mininet, em conjunto com o controlador *SDN* FloodLight. Cada um destes dias contém 3 ataques de diferentes intensidades, sendo que o primeiro deles possui ataques do tipo *DDoS*, enquanto o segundo, ataques *PS*. As implementações foram realizadas utilizando a linguagem Python em conjunto com Keras e Sklearn.

Os métodos de detecção propostos (*MLP*, *PSO* e *WaveDetect*) são avaliados em comparação com outros três métodos da literatura, sendo eles: k-*means*, *kNN* e *SVM*.

No Módulo de Mitigação, duas abordagens são utilizadas. Para mitigação de ataques *DDoS*, a abordagem *GT*, discutida anteriormente na seção 2.3, é aplicada, enquanto uma abordagem de descarte direcionado é realizada contra ataques *PS*. Isso se dá pois ataques *PS* geralmente são originados por um mesmo dispositivo. Assim, caso este dispositivo seja identificado, uma política de descarte direcionado especificamente a este *host* pode ser aplicada.

As contribuições de cada autor no desenvolvimento deste artigo foram: i) Marcos V. O. de Assis - Concepção do sistema de defesa (organização, funcionamento e localização) contra ataques internos e externos, a redução do intervalo de análise para 5 segundos, a descrição e implementação do modelo *MLP*, a implementação dos modelos k-*means*, *kNN* e *SVM*; Implementação das mitigações *GT* e direcionada, a execução de testes e a escrita; ii) Matheus P. Novaes - Descrição e implementação do método *PSO* e a revisão textual; iii) Cinara B. Zerbini - Descrição e implementação do modelo *WaveDetect*, bem como a revisão textual; iv) Taufik Abrão - Revisão conceitual do sistema proposto, e; v) Mario L. Proença Jr. - Orientação e discussão do planejamento, organização bem como revisão dos resultados encontrados.

Com relação à detecção de ambos os ataques, conforme ilustrado na Fig. 14, os métodos k-*means* e *kNN* se destacaram negativamente, com resultados de 61.71% e 86.87% na métrica "área sob a curva". Os métodos *PSO*, *MLP*, *WaveDetect* e *SVM*, por sua vez, alcançaram valores de 98.75%, 99.73%, 99.65% e 99.67%, respectivamente, para a mesma métrica, identificando os ataques com precisão.

Dentre os métodos com melhores resultados, tanto *SVM* quanto *MLP* são abordagens de aprendizagem supervisionada. Entretanto, o *SVM* tradicional não é capaz de identi-

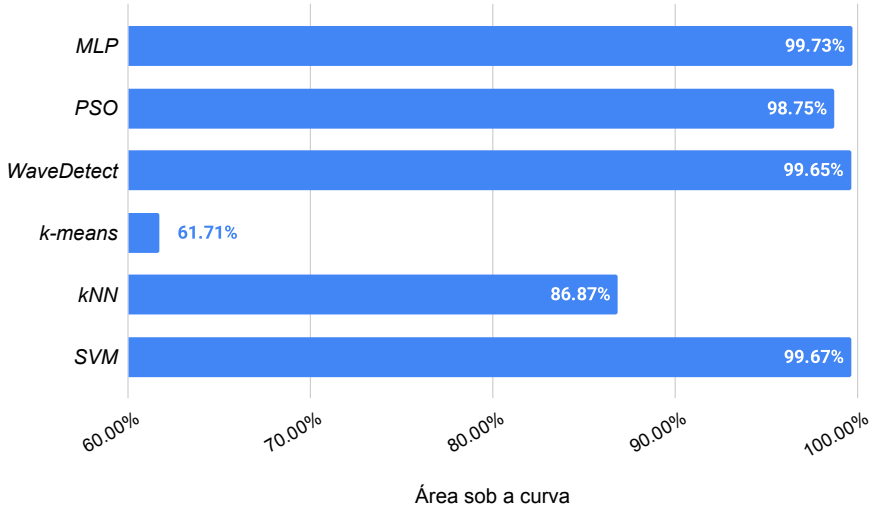


Figura 14 – Resultados da métrica de "área sob a curva" com relação à detecção de ataques pelos métodos avaliados.

ficar a anomalia detectada, uma vez que sua classificação é binária. Essa característica poderia ser contornada por meio das estratégias "um-contra-todos" ou "um-contra-um", permitindo o processamento de múltiplas classes simultaneamente com a desvantagem de adicionar complexidade computacional ao processo (GAMA et al., 2011). Dessa forma, o método *MLP* se torna a abordagem mais eficiente para detecção quando houver dados disponíveis para treinamento. Por outro lado, quando estes dados não forem disponíveis, os métodos *PSO* e *WaveDetect* podem ser aplicados. Dentre estes dois métodos, o *WaveDetect* atingiu melhores resultados na média.

Além disso, foram avaliados os resultados relativos ao Módulo de Mitigação para os ataques *DDoS* e *PS*. A abordagem *GT* foi utilizada diretamente no controlador *SDN* para protegê-lo contra ataques *DDoS*, gerando eficientes taxas de descartes e minimizando o impacto gerado em usuários finais da rede, conforme ilustrado pela Fig 15.

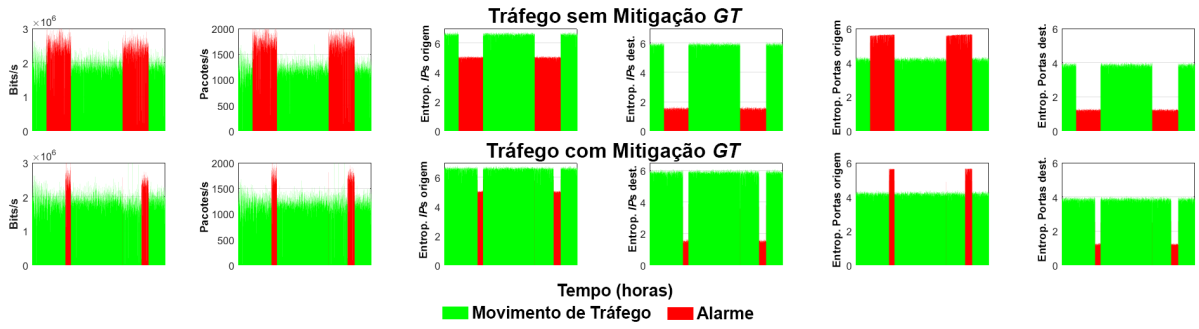


Figura 15 – Movimento de tráfego da rede avaliada antes e depois da mitigação *GT*, com detecção realizada pelo método *MLP*.

Para ataques *PS*, uma política de descarte direcionado foi aplicada, uma vez que o Módulo de Identificação do sistema de defesa foi capaz de identificar o endereço *IP*

de origem do atacante. Os resultados mostram que as abordagens de mitigação foram eficientes em trazer a rede de volta para o estado normal de funcionamento.

4.3 Artigo 3 - *Near Real-time Security System Applied to SDN Environments in IoT Networks using Convolutional Neural Network*

No terceiro artigo desenvolvido, intitulado "*Near Real-time Security System Applied on SDN Environments in IoT Networks*", um artigo publicado na revista *Computers and Electrical Engineering* em 2020, é proposto um modelo de mitigação inverso para ataques *DDoS*. Nesse esquema a mitigação ocorre na rede do *ISP* de onde o ataque está sendo gerado, diferentemente da abordagem tradicional em que o servidor atacado é quem toma as contra-medidas. Essa mudança visa não apenas proteger o servidor alvo, mas também o controlador *SDN* do *ISP* de origem contra efeitos do ataque gerado internamente contra alvos externos, uma vez que estes dispositivos são suscetíveis a ataques de negação de serviço. Protegendo o controlador *SDN* em diferentes redes de *ISP*s, o problema pode ser mitigado antes mesmo de atingir o servidor alvo, ou seja, uma abordagem de dividir para conquistar.

Além disso, o intervalo de coleta e análise dos dados é reduzido de cinco para um segundo, possibilitando respostas ainda mais rápidas do sistema, diminuindo o impacto causado em usuários finais e tornando a aplicação das políticas de mitigação mais precisa.

Por fim, é introduzida uma abordagem de detecção baseada em aprendizagem profunda (*DL*), denominada *CNN*. Essa abordagem foi comparada com três outros métodos de detecção de anomalias em testes de eficiência, sendo eles: *MLP*, *D-MLP* (ou *DNN*) e *LR*. As implementações foram realizadas utilizando a linguagem Python em conjunto com Keras e Sklearn.

As contribuições de cada autor no desenvolvimento deste artigo foram: i) Marcos V. O. de Assis - Concepção do sistema de defesa (organização e funcionamento) contra ataques indiretos, a proposição da abordagem de mitigação inversa contra ataques *DDoS*, a redução do intervalo de análise para 1 segundo, a descrição e implementação do modelo *CNN*, a implementação dos modelos *MLP* e *D-MLP*, a implementação da mitigação *GT*, o processamento da base de dados *CICDDoS* 2019 para testes no segundo cenário, a execução de testes e a escrita; ii) Luiz F. Carvalho - Implementação do modelo *LR*, a geração dos dados para testes no primeiro cenário e a revisão textual; iii) Joel J. P. C. Rodrigues - Revisão conceitual do modelo de rede neural; iv) Jaime Lloret - Orientação com relação às abordagens de aprendizagem profunda e organização textual, e; v) Mario L. Proença Jr. - Orientação e discussão do planejamento, organização bem como revisão dos resultados encontrados.

Para se mensurar a eficiência do mecanismo de detecção proposto, o método *CNN*, bem como os demais métodos avaliados, foram submetidos a dois cenários de testes. No primeiro deles, foram utilizados dados *SDN* simulados por meio do emulador de redes Mininet, OpenFlow e controlados pelo FloodLight (controlador *SDN*), em diferentes organizações. Ao todo, foram gerados sete dias de dados, contendo ataques *DDoS* de diferentes intensidades e durações. Os resultados na média, com relação às métricas de acurácia, precisão, *recall* e *f-measure*, são ilustrados na Fig. 16.

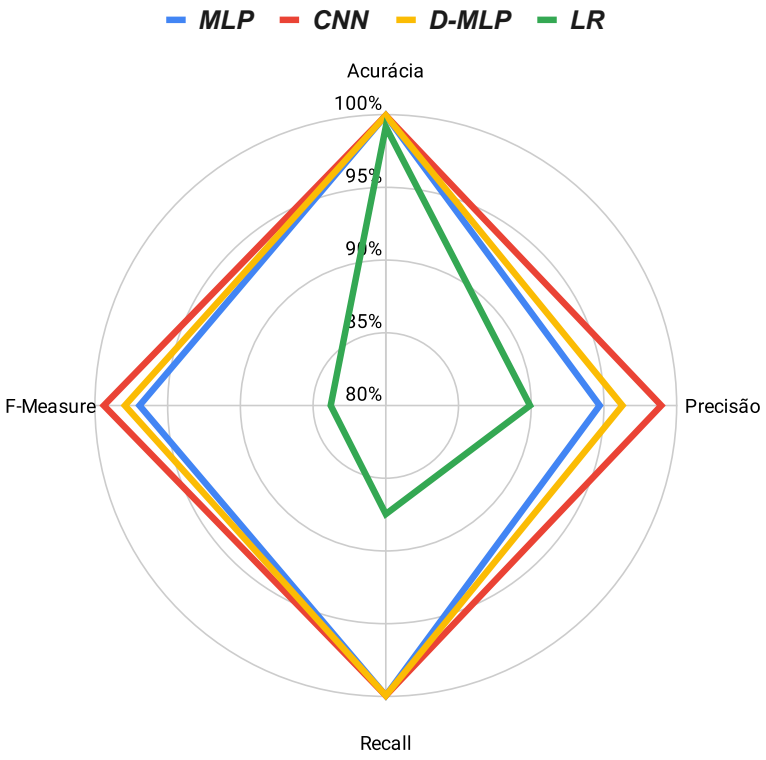


Figura 16 – Gráfico de radar apresentando os resultados na média dos métodos avaliados com relação às métricas de acurácia, precisão, *recall* e *f-measure* - Cenário 1 de testes.

No segundo cenário, os métodos são avaliados em uma base de dados *DDoS* pública, denominada *CICDDoS* 2019 (SHARAFALDIN et al., 2019), uma vez que o uso de bases de dados de referência garante maior confiabilidade no processo de comparação de métodos de detecção. Essa base de dados conta com 12 tipos de ataques *DDoS*[1], bem como 87 dimensões de fluxos *IP* disponíveis em dados rotulados para testes. Os resultados na média, com relação às métricas de acurácia, precisão, *recall* e *f-measure*, são ilustrados na Fig. 17.

Como mostrado nas Fig. 16 e 17, em ambos os cenários o *LR* atingiu resultados inferiores aos demais métodos avaliados. Os métodos *CNN*, *MLP* e *D-MLP* obtiveram

[1]Ataques descritos em: https://www.unb.ca/cic/datasets/ddos-2019.html

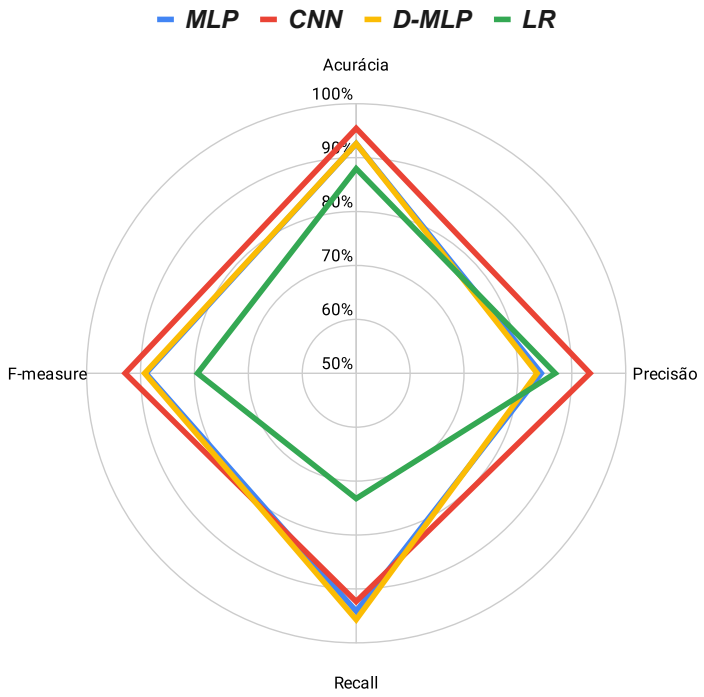


Figura 17 – Gráfico de radar apresentando os resultados na média dos métodos avaliados com relação às métricas de acurácia, precisão, *recall* e *f-measure* - Cenário 2 de testes.

resultados satisfatórios de detecção na média, apresentando valores maiores que 95% e 85% para o primeiro e segundo cenários de teste, respectivamente. O *CNN* obteve os melhores resultados na média, alcançando baixa taxa de falsos-positivos, maior acurácia, precisão e *f-measure* em comparação com as demais técnicas. Os métodos *MLP* e *D-MLP* obtiveram melhores valores de *recall* que os demais, ambos com resultados similares.

Além disso, foi aplicada a abordagem *GT* de mitigação contra ataques *DDoS* indiretos, ou seja, de *hosts* internos tendo como alvo um servidor externo à rede *SDN*. O *GT* foi aplicado a um dia de ataque *DDoS* relativo ao primeiro cenário de testes, contendo dois ataques, como pode ser observado na Fig. 18.

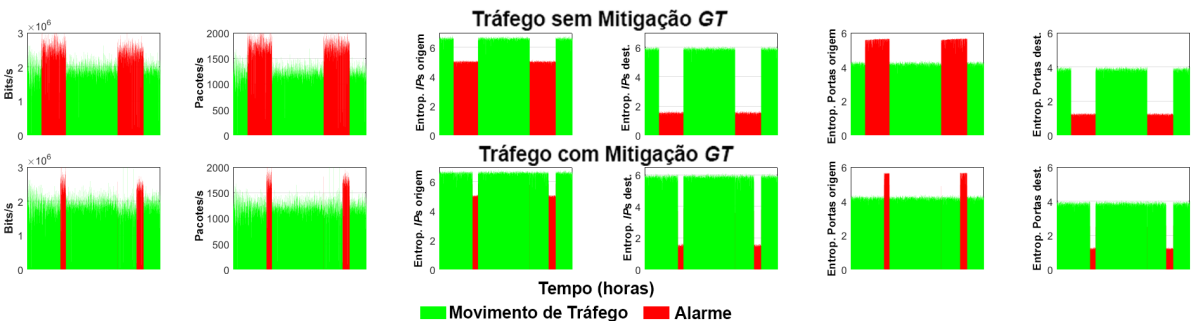


Figura 18 – Movimento de tráfego da rede avaliada antes e depois da mitigação *GT*, com detecção realizada pelo método *CNN*.

Os resultados apresentados pela Fig. 18 revelam que a abordagem de mitigação se mostrou eficiente em restaurar a rede *SDN* a seu funcionamento normal, mesmo contra ataques indiretos. Deste modo, a implantação do sistema de defesa em diferentes redes de *ISP*s pode viabilizar o mecanismo de mitigação inversa proposto e descrito na Seção 3.2.1, indiretamente protegendo o servidor alvo contra ataques distribuídos.

4.4 Artigo 4 - *A GRU Deep Learning System against Attacks in Software Defined Networks*

O quarto trabalho desenvolvido, intitulado "*A GRU Deep Learning System against Attacks in Software Defined Networks*", é um artigo atualmente em avaliação. O objetivo desse trabalho é estender a área de atuação do sistema de defesa proposto, ampliando o espectro de anomalias detectadas pelo sistema. Assim, são avaliados não somente diferentes tipos de ataques *DDoS*, mas também ataques de intrusão.

Além disso, uma importante evolução do sistema é introduzida: a coleta e análise de fluxos *IP* deixa de ser intervalada (1 segundo) e passa a ser individualizada. Essa mudança, além de tornar a resposta do sistema mais ágil, também permite a identificação direta de fluxos anômalos.

Adicionalmente, neste artigo é proposta a utilização do *GRU* (Seção 2.5.2), um mo-

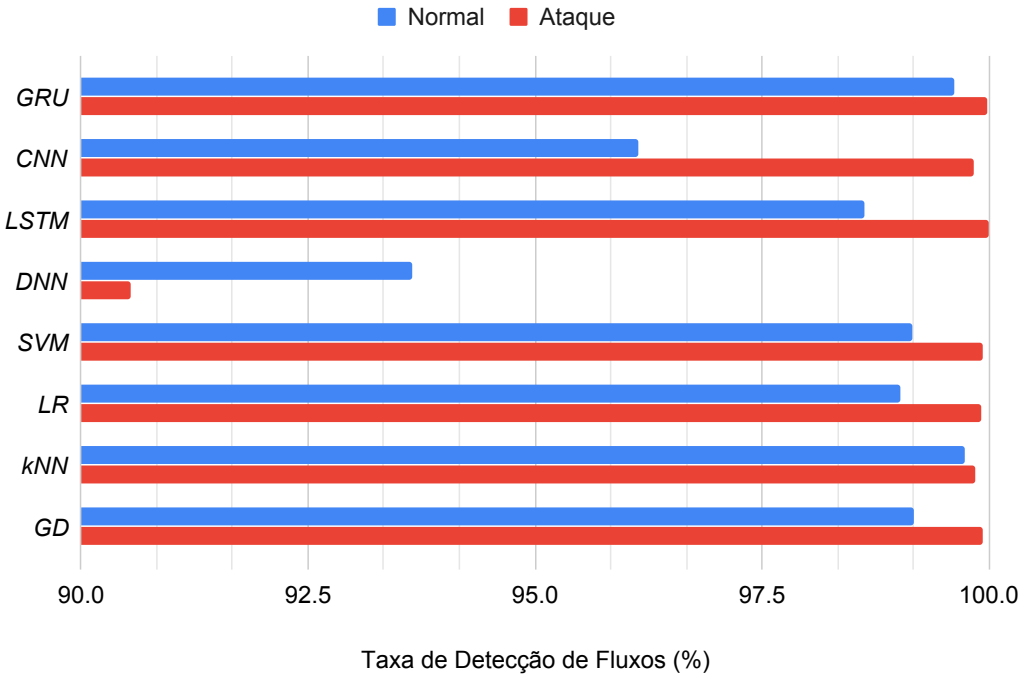


Figura 19 – Percentual de detecção de fluxos normais (usuários legítimos) e anômalos (ataques) para cada método avaliado no primeiro cenário de testes.

delo de aprendizagem profunda, supervisionado e recorrente, no Módulo de Detecção do sistema. Além disso, a eficiência deste método é avaliada em comparação com outros sete modelos, sendo eles: *D-MLP*, *CNN*, *SVM*, *LSTM*, *LR*, *kNN* e *GD*. As implementações foram realizadas utilizando a linguagem Python em conjunto com Keras e Sklearn.

Por fim, tendo como base a análise de fluxos *IP* individualizada, um modelo de descarte direcionado é proposto e avaliado como método de mitigação dos ataques detectados.

As contribuições de cada autor no desenvolvimento deste artigo foram: i) Marcos V. O. de Assis - Concepção do sistema de defesa (organização e funcionamento), a realização de coleta e análise de fluxos *IP* individuais, a descrição e implementação do modelo *GRU*, a implementação dos modelos *DNN*, *CNN*, *LSTM*, *SVM*, *kNN*, *LR* e *GD*, o processamento das bases de dados *CICDDoS* 2019 e *CICIDS* 2018 para testes, a execução de testes de detecção e viabilidade dos métodos propostos e a escrita; ii) Luiz F. Carvalho - Processamento da base de dados utilizada nos testes de viabilidade, bem como revisão textual; iii) Jaime Lloret - Orientação com relação às abordagens de aprendizagem profunda e organização textual, e; iv) Mario L. Proença Jr. - Orientação e discussão do planejamento, organização bem como revisão dos resultados encontrados.

Para mensurar a eficiência dos diferentes métodos citados, o sistema foi submetido a dois diferentes cenários de testes. O primeiro deles, assim como utilizado no Artigo 3, é a base de dados pública *CICDDoS* 2019, composta de 12 tipos de ataques *DDoS*[2]. Para esse cenário, foram utilizadas 83 diferentes dimensões de fluxo. Algumas dimensões foram excluídas da análise, como endereços *IP*, buscando evitar o enviesamento das classificações

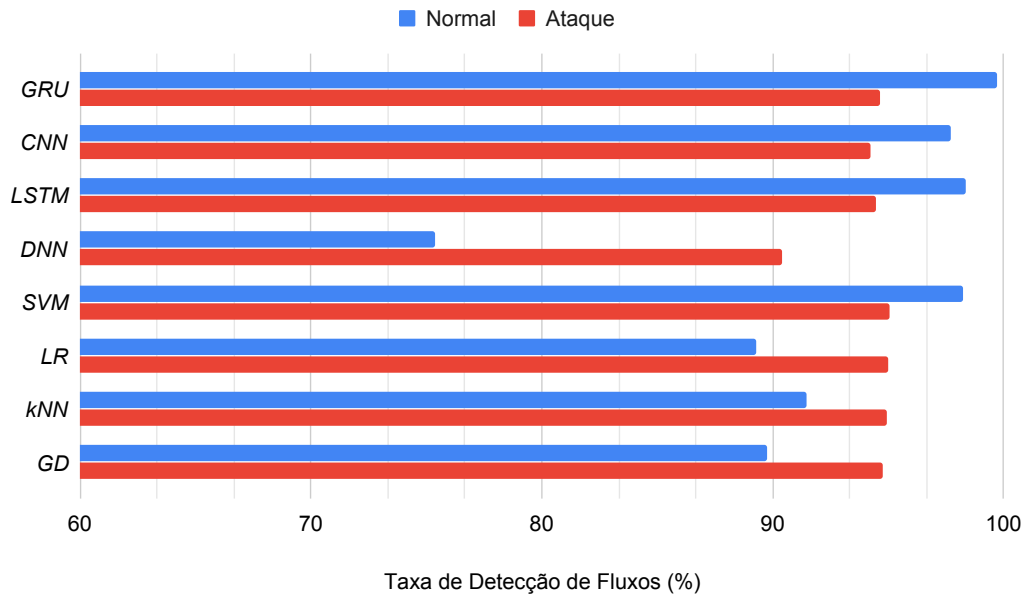


Figura 20 – Percentual de detecção de fluxos normais (usuários legítimos) e anômalos (ataques) para cada método avaliado no segundo cenário de testes.

[2]Ataques descritos em: https://www.unb.ca/cic/datasets/ddos-2019.html

do sistema (evitando relacionar, por exemplo, um ataque a um endereço *IP* específico). A Fig. 19 apresenta o percentual de detecção dos métodos com relação à fluxos normais (usuários legítimos) e anômalos (ataques).

Como observado na Fig. 19, com exceção do *DNN*, todos os métodos avaliados atingiram resultados semelhantes considerando as taxas de detecção de ataques. Entretanto, os métodos *GRU* e *kNN* apresentaram maior taxa de detecção de fluxos legítimos, com o método *GRU* obtendo o melhor resultado na média por uma pequena margem. A utilização de um grande conjunto de dimensões de fluxos *IP* influencia diretamente na detecção de ataques *DDoS* pelos métodos avaliados.

No segundo cenário de testes foi utilizada a base de dados pública *CICIDS* 2018 (SHARAFALDIN; LASHKARI; GHORBANI, 2018), composta de 14 diferentes técnicas de intrusão[3]. Da mesma forma realizada no cenário anterior, foram retiradas dimensões de fluxo *IP* que poderiam enviesar (ou "viciar") os resultados. Assim, 79 dimensões foram submetidas aos modelos avaliados. A Fig. 20 ilustra o percentual de detecção de fluxos normais e anômalos para este cenário de testes.

No segundo cenário os resultados se mostram mais heterogêneos, uma vez que a rede emulada é composta por mais dispositivos e esses ataques (intrusões) tendem a ser mais sutis que ataques *DDoS*. Como observado na Fig. 20, o método *GRU* novamente obteve a melhor performance dentre os modelos avaliados, atingindo os melhores resultados

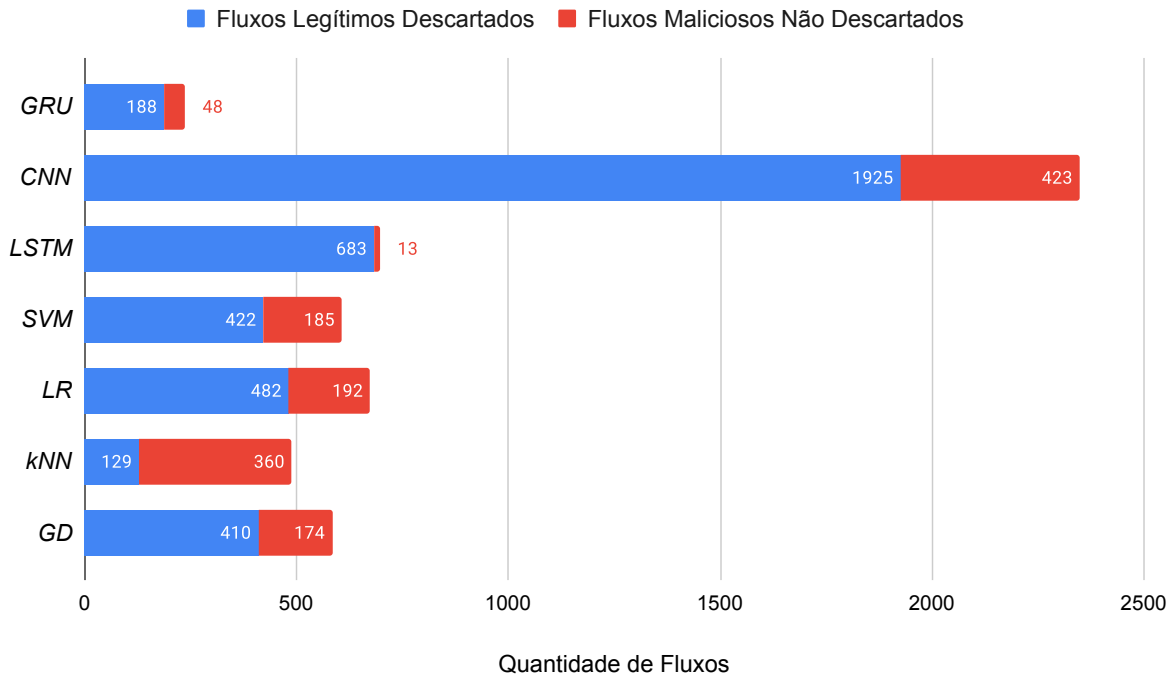


Figura 21 – Quantidade de fluxos normais descartados e anômalos não descartados para cada método avaliado no primeiro cenário de testes.

[3]Ataques descritos em: https://www.unb.ca/cic/datasets/ids-2018.html

gerais (mais balanceados) com relação à detecção de fluxos normais e anômalos. Como observado, novamente os métodos avaliados apresentaram taxas de detecção de ataques semelhantes, enquanto o método *GRU* atingiu melhores resultados de detecção de fluxos legítimos.

Além disso, uma abordagem de descarte direcionado de fluxos foi implementada e avaliada com relação aos ataques considerados em ambos os cenários de testes. As Fig. 21 e 22 apresentam a quantidade de fluxos legítimos descartados (usuários prejudicados) em conjunto com a quantidade de fluxos maliciosos não descartados (não detectados) para o primeiro e segundo cenários de teste, respectivamente.

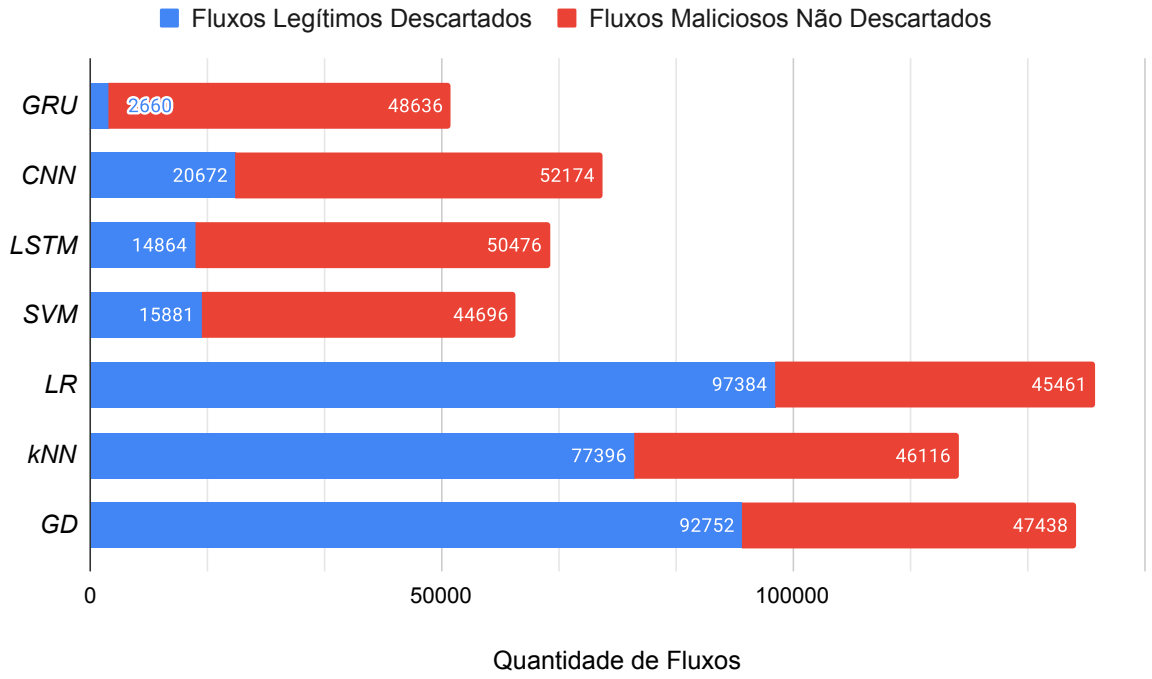


Figura 22 – Quantidade de fluxos normais descartados e anômalos não descartados para cada método avaliado no segundo cenário de testes.

Quanto menor os resultados nas Fig. 21 e 22, melhor o desempenho da abordagem de mitigação direcionada. Nesse tipo de abordagem, a eficiência da detecção influencia diretamente na qualidade da mitigação. Como observado, o método *GRU* atingiu os menores valores gerais em ambos os testes de mitigação, o que condiz com os resultados obtidos nos cenários de teste anteriormente avaliados. Além disso, pode-se concluir que sua utilização otimiza a operação do Módulo de Mitigação.

Por fim, foi mensurada a quantidade de fluxos por segundo que cada método é capaz de analisar e classificar, uma vez que velocidade é uma característica fundamental para o sistema (análise individual de fluxos). Os resultados obtidos foram comparados com dados reais coletados na rede da Universidade Estadual de Londrina, a qual é considerada de larga escala. Os resultados deste teste podem ser observados na Fig. 23.

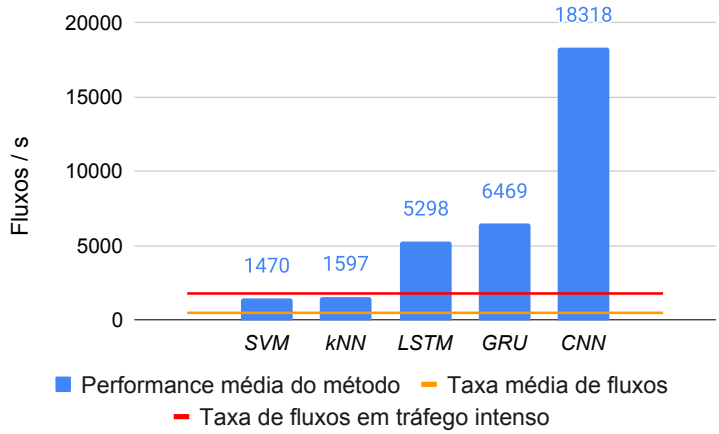


Figura 23 – Taxa de vazão em Fluxos/s dos métodos *GRU*, *CNN*, *LSTM*, *SVM* e *kNN* em comparação com dados reais coletados de uma rede de larga escala.

Como apresentado na Fig. 23, dos métodos avaliados que obtiveram melhores resultados de classificação nos cenários de testes, a utilização de *SVM*, *kNN*, *LSTM*, *GRU* e *CNN* se mostrou factível, considerando o valor médio de fluxos/s exportados pela rede avaliada. Entretanto, quando a taxa de captura de fluxos se mostra elevada (pior caso), a vazão dos métodos *SVM* e *kNN* não se mostra suficiente. Dessa forma, dos métodos avaliados, apenas *LSTM*, *GRU* e *CNN* possuem aplicação factível no sistema proposto. Esses resultados, somados aos obtidos por meio dos cenários de testes, levam à conclusão de que o *GRU* é uma abordagem promissora e factível na detecção de anomalias de ambientes *SDN* reais.

5 Conclusões

A presente tese tem como objetivo propor um sistema de defesa contra ataques *DDoS* e de intrusão em ambientes *SDN*. Embora a utilização desse paradigma traga diversos benefícios relacionados ao gerenciamento centralizado de recursos, roteamento dinâmico, entre outros, a utilização de um controlador central cria um ponto crítico de falhas, o qual precisa ser protegido. Com essa motivação o sistema proposto foi desenvolvido e aprimorado ao longo de quatro artigos científicos, três deles publicados e um em processo de análise. Dentre as principais contribuições deste trabalho, pode-se destacar:

- **Ambiente de aplicação do sistema:** O sistema proposto foi aplicado e avaliado em três diferentes ambientes. Primeiramente localizado em um dispositivo de borda, visando defender o controlador contra ataques externos. Em um segundo momento, o sistema foi aplicado diretamente no controlador *SDN*, possibilitando a defesa contra ataques tanto internos quanto externos. Por fim, ainda localizado no controlador central, o sistema foi aplicado na defesa de ataques com origem interna tendo como alvo dispositivos externos à rede (ataque indireto). Nesse último ambiente, foi proposto um esquema de mitigação inversa, buscando tanto a proteção do controlador de ambientes *SDN* quanto do alvo externo.

- **Redução do tempo de intervalo de coleta e análise de dados:** Durante o desenvolvimento do presente trabalho, o tempo de coleta e análise de fluxos *IP* foi sendo gradativamente reduzido. Inicialmente realizado em intervalos de um minuto, este intervalo foi reduzido para intervalos de cinco segundos, um segundo e, finalmente, para uma análise de fluxos individualizada. Essa redução consequentemente contribui para a diminuição do tempo de resposta do sistema, o que, por sua vez, reduz o impacto causado por ataques à rede.

- **Aplicação de novas técnicas para a solução da Detecção de Ataques:** Foram propostas e avaliadas diferentes técnicas e abordagens voltadas à detecção de ataques *DDoS* e de intrusão em ambientes *SDN*. Os métodos propostos são de diversas classes, sendo avaliados desde modelos estatísticos a técnicas de aprendizagem de máquina e aprendizagem profunda.

- **Mitigação de ataques:** Uma abordagem de mitigação de ataques *DDoS* foi modelada utilizando Teoria de Jogos. Além disso, políticas de descarte direcionado foram implementadas em casos em que o sistema proposto é capaz de identificar o dispositivo atacante.

O presente sistema de defesa foi desenvolvido através de uma organização modular, dividindo seu funcionamento em três módulos principais: Detecção, Identificação e Mitigação. Essa modularização permite que métodos específicos sejam aprimorados, ou mesmo substituídos, de forma isolada, sem prejudicar o funcionamento geral do sistema.

O Módulo de Detecção é responsável por analisar diferentes dimensões de fluxos *IP* e detectar a presença de ataques, sejam eles *DDoS* ou de intrusão. Neste módulo, foram propostos e avaliados diferentes métodos de detecção, tais como *HWDS* (método estatístico de previsão), *MLP* (aprendizagem de máquina e rede neural), *CNN* e *GRU* (aprendizagem profunda). Além disso, uma comparação foi realizada entre os métodos propostos e dez outras abordagens de detecção ao longo dos quatro artigos desenvolvidos. Tal comparação levou em consideração primeiramente a eficiência de detecção. Posteriormente, uma análise de capacidade de processamento de fluxos por segundo (vazão) foi realizada, pretendendo avaliar a factibilidade da aplicação dos modelos em ambientes não-emulados. Os testes realizados apontam o método *GRU* como factível, bem como a mais eficiente abordagem de detecção de ataques no sistema. Em ambientes com capacidade reduzida de processamento pelo controlador *SDN*, uma alternativa é a utilização de *IP*. Este método apresentou resultados consistentes de detecção, além de ser mais rápido que o *GRU* devido à sua organização sem retroalimentação (modelo *feedforward*).

O Módulo de Identificação desempenha o papel de auxiliar as ações de mitigação. As informações coletadas dos ataques auxiliam na decisão de qual abordagem de mitigação deve ser aplicada. Além disso, a análise de fluxos *IP* de usuários legítimos permite a criação da chamada *safe list*. Essa lista aprimora os resultados obtidos no Módulo de Mitigação, diminuindo a taxa de descarte de pacotes relativos a usuários legítimos.

Por fim, o Módulo de Mitigação é responsável pela geração de políticas de descarte, visando a proteção da rede contra ataques *DDoS* ou de intrusão. Uma abordagem baseada em Teoria de Jogos foi desenvolvida para gerar contra-medidas eficientes contra ataques *DDoS*. Essa abordagem foi avaliada em diferentes cenários e ambientes, e se mostrou uma solução eficaz, capaz de retornar a rede ao seu estado normal de operação. É importante ressaltar que a *safe list* gerada pelo Módulo de Identificação desempenha importante papel nessa técnica de mitigação. Além disso, políticas de descarte direcionado foram utilizadas com eficiência em situações em que o atacante pode ser identificado, como ataques *DoS* e *PS*.

Os resultados obtidos demonstram que o sistema proposto obteve resultados satisfatórios no que diz respeito ao cumprimento de seus objetivos. Utilizando eficientes técnicas de detecção e mitigação, aliadas à redução do tempo de resposta e aplicação em diferentes ambientes (contra ataques internos, externos e indiretos), o sistema foi capaz de regularizar o funcionamento da rede *SDN* de forma automática, sem intervenção humana. Com isso, o controlador central é protegido, consequentemente evitando que usuários finais da rede sejam prejudicados.

Devido à modularidade do sistema, outras técnicas de detecção ou mitigação podem ser aplicadas de modo a aprimorar seu funcionamento em trabalhos futuros. Podem ser apontadas como possíveis continuidades deste trabalho:

- a detecção de ataques inovadores (*zero day*) - A utilização de mecanismos de aprendizagem profunda provê ao sistema de defesa um certo grau de generalização, ou seja, a capacidade de generalizar a classificação de ataques para entradas que não estavam presentes no processo de treinamento. Assim, a avaliação da eficácia do sistema para esses tipos de entrada representa um interessante campo de pesquisa;

- a generalização dos modelos *GRU* e *CNN* como classificadores de múltiplos rótulos - Em sua versão atual, o sistema é capaz de distinguir entre fluxos normais e anômalos de forma binária. Um possível trabalho futuro seria a implantação de uma camada de saída que utilize uma codificação para representar múltiplos rótulos. Tal mudança possibilitaria não somente a detecção, mas também a identificação de tipos específicos de ataques (diferentes tipos de *DDoS*, por exemplo), o que pode contribuir para o aumento da eficiência do Módulo de Mitigação;

- a incorporação de diferentes tipos de ataques no jogo modelado - Atualmente, o jogo modelado trata de forma única qualquer tipo de ataque *DDoS*. A avaliação de particularidades desses diferentes ataques pode auxiliar no aprimoramento do jogo implementado, representando um promissor campo de pesquisa.

- a avaliação de diferentes abordagens de detecção no sistema - Como observado nos cenários de testes avaliados, cada método possui características únicas que podem contribuir para o processo de detecção de ataques. Como exemplos, é possível citar a capacidade de identificação de padrões locais do *CNN* e a caracterização de dependências temporais dos métodos *GRU* e *LSTM*. Deste modo, a avaliação de métodos *ensemble*, ou seja, a utilização de múltiplos métodos simultaneamente no processo de detecção, pode ser considerada um promissor campo para pesquisas futuras. Além disso, a adaptabilidade provida por métodos de aprendizagem por reforço pode ser uma interessante possibilidade para a continuidade do desenvolvimento deste trabalho.

Referências

ABDEL-HAMID, O.; MOHAMED, A.; JIANG, H.; DENG, L.; PENN, G.; YU, D. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, v. 22, n. 10, p. 1533–1545, Oct 2014. ISSN 2329-9290. Disponível em: <http://dx.doi.org/doi:10.1109/TASLP.2014.2339736>. 48

ABDULHAMMED, R.; FAEZIPOUR, M.; ABUZNEID, A.; ABUMALLOUH, A. Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic. *IEEE Sensors Letters*, v. 3, n. 1, p. 1–4, Jan 2019. Disponível em: <http://dx.doi.org/doi:10.1109/LSENS.2018.2879990>. 61

ABIODUN, O. I.; JANTAN, A.; OMOLARA, A. E.; DADA, K. V.; UMAR, A. M.; LINUS, O. U.; ARSHAD, H.; KAZAURE, A. A.; GANA, U.; KIRU, M. U. Comprehensive review of artificial neural network applications to pattern recognition. *IEEE Access*, v. 7, p. 158820–158846, 2019. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2019.2945545>. 43

ADANIYA, M. H. A. C.; LIMA, M. F.; RODRIGUES, J. J. P. C.; ABRAO, T.; PROENÇA, M. L. Anomaly detection using dsns and firefly harmonic clustering algorithm. In: *2012 IEEE International Conference on Communications (ICC)*. [s.n.], 2012. p. 1183–1187. Disponível em: <http://dx.doi.org/doi:10.1109/ICC.2012.6364088>. 32

AJAEIYA, G. A.; ADALIAN, N.; ELHAJJ, I. H.; KAYSSI, A.; CHEHAB, A. Flow-based intrusion detection system for sdn. In: *2017 IEEE Symposium on Computers and Communications (ISCC)*. [s.n.], 2017. p. 787–793. Disponível em: <http://dx.doi.org/doi:10.1109/ISCC.2017.8024623>. 36

ALJAMAL, I.; TEKEOğLU, A.; BEKIROGLU, K.; SENGUPTA, S. Hybrid intrusion detection system using machine learning techniques in cloud computing environments. In: *2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA)*. [s.n.], 2019. p. 84–89. Disponível em: <http://dx.doi.org/doi:10.1109/SERA.2019.8886794>. 30

ASSIS, M. V.; RODRIGUES, J. J.; PROENçA JR., M. L. A seven-dimensional flow analysis to help autonomous network management. *Information Sciences*, v. 278, p. 900 – 913, 2014. ISSN 0020-0255. Disponível em: <http://dx.doi.org/doi:10.1016/j.ins.2014.03.102>. 56, 60, 62, 68

ASSIS, M. V. O.; CARVALHO, L. F.; RODRIGUES, J. J. P. C.; PROENÇA, M. L. Holt-winters statistical forecasting and aco metaheuristic for traffic characterization. In: *2013 IEEE International Conference on Communications (ICC)*. [s.n.], 2013. p. 2524–2528. Disponível em: <http://dx.doi.org/doi:10.1109/ICC.2013.6654913>. 40

ASSIS, M. V. O.; 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.n.], 2013. p. 735–740. Disponível em: <http://dx.doi.org/doi:10.1109/GLOCOM.2013.6831160>. 56

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, v. 5, n. 2, p. 157–166, March 1994. ISSN 1941-0093. Disponível em: <http://dx.doi.org/doi:10.1109/72.279181>. 51

BEREZIńSKI, P.; JASIUL, B.; SZPYRKA, M. An entropy-based network anomaly detection method. *Entropy*, v. 17, n. 4, p. 2367–2408, 2015. ISSN 1099-4300. Disponível em: <http://dx.doi.org/doi:10.3390/e17042367>. 32, 59

BHARDWAJ, A.; SUBRAHMANYAM, G. V. B.; AVASTHI, V.; SASTRY, H.; GOUNDAR, S. Ddos attacks, new ddos taxonomy and mitigation solutions — a survey. In: *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*. [s.n.], 2016. p. 793–798. Disponível em: <http://dx.doi.org/doi:10.1109/SCOPES.2016.7955549>. 38

BHUNIA, S. S.; GURUSAMY, M. Dynamic attack detection and mitigation in iot using sdn. In: *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*. [s.n.], 2017. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/ATNAC.2017.8215418>. 33

BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738. 43

BORKAR, A.; DONODE, A.; KUMARI, A. A survey on intrusion detection system (ids) and internal intrusion detection and protection system (iidps). In: *2017 International Conference on Inventive Computing and Informatics (ICICI)*. [s.n.], 2017. p. 949–953. Disponível em: <http://dx.doi.org/doi:10.1109/ICICI.2017.8365277>. 38

BUDUMA, N.; LOCASCIO, N. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. [S.l.]: O'Reilly Media, 2017. ISBN 9781491925560. 48, 50

BUTUN, I.; ÖSTERBERG, P.; SONG, H. Security of the internet of things: Vulnerabilities, attacks, and countermeasures. *IEEE Communications Surveys Tutorials*, v. 22, n. 1, p. 616–644, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2019.2953364>. 30

CANIZO, M.; TRIGUERO, I.; CONDE, A.; ONIEVA, E. Multi-head cnn–rnn for multi-time series anomaly detection: An industrial case study. *Neurocomputing*, v. 363, p. 246 – 260, 2019. ISSN 0925-2312. Disponível em: <http://dx.doi.org/doi:10.1016/j.neucom.2019.07.034>. 51

CARVALHO, L. F.; ABRãO, T.; MENDES, L. de S.; PROENÇA, M. L. An ecosystem for anomaly detection and mitigation in software-defined networking. *Expert Systems with Applications*, v. 104, p. 121 – 133, 2018. ISSN 0957-4174. Disponível em: <http://dx.doi.org/doi:10.1016/j.eswa.2018.03.027>. 32, 59, 61

CHAABOUNI, N.; MOSBAH, M.; ZEMMARI, A.; SAUVIGNAC, C.; FARUKI, P. Network intrusion detection for iot security based on learning techniques. *IEEE Communications Surveys Tutorials*, v. 21, n. 3, p. 2671–2701, thirdquarter 2019. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2019.2896380>. 35

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 41, n. 3, jul. 2009. ISSN 0360-0300. Disponível em: <http://dx.doi.org/doi:10.1145/1541880.1541882>. 37

CHIROMA, H.; ABDULLAHI, U. A.; ABDULHAMID, S. M.; ALAROOD, A. A.; GABRALLA, L. A.; RANA, N.; SHUIB, L.; HASHEM, I. A. T.; GBENGA, D. E.; ABUBAKAR, A. I.; ZEKI, A. M.; HERAWAN, T. Progress on artificial neural networks for big data analytics: A survey. *IEEE Access*, v. 7, p. 70535–70551, 2019. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2018.2880694>. 43

CHO, K.; MERRIËNBOER, B. van; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1724–1734. Disponível em: <http://dx.doi.org/doi:10.3115/v1/D14-1179>. 51, 56

CHOLLET, F. *Deep Learning with Python*. 1st. ed. Greenwich, CT, USA: Manning Publications Co., 2017. ISBN 1617294438, 9781617294433. 17, 46, 48, 49, 56

CISCO. *Anual Cybersecurity Report*. CISCO, 2018. Disponível em: <https://www.cisco.com/c/dam/m/digital/elq-cmcglobal/witb/acr2018/acr2018final.pdf?dtid=odicdc000016&ccid=cc000160&oid=anrsc005679&ecid=8196&elqTrackId=686210143d34494fa27ff73da9690a5b&elqaid=9452&elqat=2>. 30

DONG, S.; ABBAS, K.; JAIN, R. A survey on distributed denial of service (ddos) attacks in sdn and cloud computing environments. *IEEE Access*, v. 7, p. 80813–80828, 2019. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2019.2922196>. 29

ELTANBOULY, S.; BASHENDY, M.; ALNAIMI, N.; CHKIRBENE, Z.; ERBAD, A. Machine learning techniques for network anomaly detection: A survey. In: *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*. [s.n.], 2020. p. 156–162. Disponível em: <http://dx.doi.org/doi:10.1109/ICIoT48696.2020.9089465>. 30

FERNANDES JR., G.; RODRIGUES, J. J.; CARVALHO, L. F.; AL-MUHTADI, J. F.; PROENÇA, M. L. A comprehensive survey on network anomaly detection. *Telecommun. Syst.*, Kluwer Academic Publishers, Norwell, MA, USA, v. 70, n. 3, p. 447–489, mar. 2019. ISSN 1018-4864. Disponível em: <http://dx.doi.org/doi:10.1007/s11235-018-0475-8>. 30, 37

FLORENCIO, F. de A.; MORENO, E. D.; MACEDO, H. T.; SALGUEIRO, R. J. P. de B.; NASCIMENTO, F. B. do; SANTOS, F. A. O. Intrusion detection via mlp neural network using an arduino embedded system. In: *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC)*. [s.n.], 2018. p. 190–195. Disponível em: <http://dx.doi.org/doi:10.1109/SBESC.2018.00036>. 46

FRUSTACI, M.; PACE, P.; ALOI, G.; FORTINO, G. Evaluating critical security issues of the iot world: Present and future challenges. *IEEE Internet of Things Journal*, v. 5, n. 4, p. 2483–2495, Aug 2018. ISSN 2327-4662. Disponível em: <http://dx.doi.org/doi:10.1109/JIOT.2017.2767291>. 29, 30

GADDAM, A.; WILKIN, T.; ANGELOVA, M. Anomaly detection models for detecting sensor faults and outliers in the iot - a survey. In: *2019 13th International Conference on Sensing Technology (ICST)*. [s.n.], 2019. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/ICST46873.2019.9047684>. 29

GAMA, J.; FACELI, K.; LORENA, A.; CARVALHO, A. D. *Inteligência artificial: uma abordagem de aprendizado de máquina*. [S.l.]: Grupo Gen - LTC, 2011. ISBN 9788521618805. 71

GHIMIRE, B. A game theory based reputation system for security. In: *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. [s.n.], 2019. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/ICECCT.2019.8869002>. 31

GUBBI, J.; BUYYA, R.; MARUSIC, S.; PALANISWAMI, M. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, v. 29, n. 7, p. 1645 – 1660, 2013. ISSN 0167-739X. Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond. Disponível em: <http://dx.doi.org/doi:10.1016/j.future.2013.01.010>. 30

HAKIRI, A.; GOKHALE, A.; BERTHOU, P.; SCHMIDT, D. C.; GAYRAUD, T. Software-defined networking: Challenges and research opportunities for future internet. *Computer Networks*, Elsevier B.V., v. 75, n. PartA, p. 453–471, 2014. ISSN 13891286. Disponível em: <http://dx.doi.org/doi:10.1016/j.comnet.2014.10.015>. 29

HAMAMOTO, A. H.; CARVALHO, L. F.; SAMPAIO, L. D. H.; ABRãO, T.; PROENÇA, M. L. Network anomaly detection system using genetic algorithm and fuzzy logic. *Expert Systems with Applications*, v. 92, p. 390 – 402, 2018. ISSN 0957-4174. Disponível em: <http://dx.doi.org/doi:10.1016/j.eswa.2017.09.013>. 32

HATCHER, W. G.; YU, W. A survey of deep learning: Platforms, applications and emerging research trends. *IEEE Access*, p. 24411–24432, 2018. ISSN 2169-3536. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2018.2830661>. 47, 48

HAUTAMAKI, V.; KARKKAINEN, I.; FRANTI, P. Outlier detection using k-nearest neighbour graph. In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* [s.n.], 2004. v. 3, p. 430–433 Vol.3. ISSN 1051-4651. Disponível em: <http://dx.doi.org/doi:10.1109/ICPR.2004.1334558>. 61

HAYKIN, S. *Neural Networks and Learning Machines*. [S.l.]: Pearson Education, 2011. ISBN 9780133002553. 43, 44, 56

HE, T.; DROPPO, J. Exploiting lstm structure in deep neural networks for speech recognition. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [s.n.], 2016. p. 5445–5449. ISSN 2379-190X. Disponível em: <http://dx.doi.org/doi:10.1109/ICASSP.2016.7472718>. 51

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 9, n. 8, p. 1735–1780, nov. 1997. ISSN 0899-7667. Disponível em: <http://dx.doi.org/doi:10.1162/neco.1997.9.8.1735>. 51

HOQUE, N.; BHATTACHARYYA, D. K.; KALITA, J. K. Botnet in ddos attacks: Trends and challenges. *IEEE Communications Surveys Tutorials*, v. 17, n. 4, p. 2242–2270, 2015. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2015.2457491>. 38

ISHAQUE, M.; HUDEC, L. Feature extraction using deep learning for intrusion detection system. In: *2019 2nd International Conference on Computer Applications Information Security (ICCAIS)*. [s.n.], 2019. p. 1–5. Disponível em: <http://dx.doi.org/doi:10.1109/CAIS.2019.8769473>. 31

JAIN, S.; KUMAR, A.; MANDAL, S.; ONG, J.; POUTIEVSKI, L.; SINGH, A.; VENKATA, S.; WANDERER, J.; ZHOU, J.; ZHU, M.; ZOLLA, J.; HöLZLE, U.; STUART, S.; VAHDAT, A. B4: Experience with a globally-deployed software defined wan. In: *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. New York, NY, USA: Association for Computing Machinery, 2013. (SIGCOMM '13), p. 3–14. ISBN 9781450320566. Disponível em: <http://dx.doi.org/doi:10.1145/2486001.2486019>. 36

JONKER, M.; KING, A.; KRUPP, J.; ROSSOW, C.; SPEROTTO, A.; DAINOTTI, A. Millions of targets under attack: A macroscopic characterization of the dos ecosystem. In: *Proceedings of the 2017 Internet Measurement Conference*. New York, NY, USA: ACM, 2017. (IMC '17), p. 100–113. ISBN 978-1-4503-5118-8. Disponível em: <http://dx.doi.org/doi:10.1145/3131365.3131383>. 29

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. Disponível em: <http://dx.doi.org/doi:10.1109/MCOM.2017.1600970>. 29, 36

KAO, J.; JIANG, J. Anomaly detection for univariate time series with statistics and deep learning. In: *2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*. [s.n.], 2019. p. 404–407. ISSN null. Disponível em: <http://dx.doi.org/doi:10.1109/ECICE47484.2019.8942727>. 47

KIM, H.; KIM, T.; JANG, D. An intelligent improvement of internet-wide scan engine for fast discovery of vulnerable iot devices. *Symmetry*, v. 10, n. 5, 2018. ISSN 2073-8994. Disponível em: <http://dx.doi.org/doi:10.3390/sym10050151>. 29

KOLIAS, C.; KAMBOURAKIS, G.; STAVROU, A.; VOAS, J. Ddos in the iot: Mirai and other botnets. *Computer*, v. 50, n. 7, p. 80–84, 2017. ISSN 0018-9162. Disponível em: <http://dx.doi.org/doi:10.1109/MC.2017.201>. 30

KREUTZ, D.; RAMOS, F. M. V.; VERíSSIMO, P. E.; ROTHENBERG, C. E.; AZODOLMOLKY, S.; UHLIG, S. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, v. 103, n. 1, p. 14–76, Jan 2015. ISSN 0018-9219. Disponível em: <http://dx.doi.org/doi:10.1109/JPROC.2014.2371999>. 29

KUNAL; DUA, M. Machine learning approach to ids: A comprehensive review. In: *2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*. [s.n.], 2019. p. 117–121. Disponível em: <http://dx.doi.org/doi:10.1109/ICECA.2019.8822120>. 30

LAN, H.; PAN, Y. Sdn abnormal traffic detection algorithm based on rescaled range analysis. In: *2019 Computing, Communications and IoT Applications*

(ComComAp). [s.n.], 2019. p. 231–236. Disponível em: <http://dx.doi.org/doi:10.1109/ComComAp46287.2019.9018832>. 33

LAVROVA, D.; ZEGZHDA, D.; YARMAK, A. Using gru neural network for cyber-attack detection in automated process control systems. In: *2019 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. [s.n.], 2019. p. 1–3. Disponível em: <http://dx.doi.org/doi:10.1109/BlackSeaCom.2019.8812818>. 54

LEI, Y. Network anomaly traffic detection algorithm based on svm. In: *2017 International Conference on Robots Intelligent System (ICRIS)*. [s.n.], 2017. p. 217–220. ISSN null. Disponível em: <http://dx.doi.org/doi:10.1109/ICRIS.2017.61>. 61

LI, P.; ZHANG, Y. A novel intrusion detection method for internet of things. In: *2019 Chinese Control And Decision Conference (CCDC)*. [s.n.], 2019. p. 4761–4765. Disponível em: <http://dx.doi.org/doi:10.1109/CCDC.2019.8832753>. 30

LI, W.; MENG, W.; KWOK, L. F. A survey on openflow-based software defined networks: Security challenges and countermeasures. *Journal of Network and Computer Applications*, v. 68, p. 126 – 139, 2016. ISSN 1084-8045. Disponível em: <http://dx.doi.org/doi:10.1016/j.jnca.2016.04.011>. 35, 36

LI, X.; WU, X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [s.n.], 2015. p. 4520–4524. ISSN 1520-6149. Disponível em: <http://dx.doi.org/doi:10.1109/ICASSP.2015.7178826>. 47

LIU, H.; LANG, B.; LIU, M.; YAN, H. Cnn and rnn based payload classification methods for attack detection. *Knowledge-Based Systems*, v. 163, p. 332 – 341, 2019. ISSN 0950-7051. Disponível em: <http://dx.doi.org/doi:10.1016/j.knosys.2018.08.036>. 47, 51

LIU, S.; CHEN, X.; PENG, X.; XIAO, R. Network log anomaly detection based on gru and svdd. In: *2019 IEEE Intl Conf on Parallel Distributed Processing with Applications, Big Data Cloud Computing, Sustainable Computing Communications, Social Computing Networking (ISPA/BDCloud/SocialCom/-SustainCom)*. [s.n.], 2019. p. 1244–1249. Disponível em: <http://dx.doi.org/doi:10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00177>. 54

LIU, Y.; ZHAO, B.; ZHAO, P.; FAN, P.; LIU, H. A survey: Typical security issues of software-defined networking. *China Communications*, v. 16, n. 7, p. 13–31, 2019. Disponível em: <http://dx.doi.org/doi:10.23919/JCC.2019.07.002>. 29

LUCHS, M.; DOERR, C. The curious case of port 0. In: *2019 IFIP Networking Conference (IFIP Networking)*. [s.n.], 2019. p. 1–9. Disponível em: <http://dx.doi.org/doi:10.23919/IFIPNetworking.2019.8816853>. 30

MARWAT, S. N. K.; MEHMOOD, Y.; KHAN, A.; AHMED, S.; HAFEEZ, A.; KAMAL, T.; KHAN, A. Method for handling massive iot traffic in 5g networks. *Sensors*, v. 18, n. 11, 2018. ISSN 1424-8220. Disponível em: <http://dx.doi.org/doi:10.3390/s18113966>. 29

MASOUDI, R.; GHAFFARI, A. Software defined networks: A survey. *Journal of Network and Computer Applications*, v. 67, p. 1 – 25, 2016. ISSN 1084-8045. Disponível em: <http://dx.doi.org/doi:10.1016/j.jnca.2016.03.016>. 35

MENDES, L. D. P.; ALOI, J.; PIMENTA, T. C. Analysis of iot botnet architectures and recent defense proposals. In: *2019 31st International Conference on Microelectronics (ICM)*. [s.n.], 2019. p. 186–189. Disponível em: <http://dx.doi.org/doi:10.1109/ICM48031.2019.9021715>. 30

MEYER, M. L. B.; LABIT, Y. Combining machine learning and behavior analysis techniques for network security. In: *2020 International Conference on Information Networking (ICOIN)*. [s.n.], 2020. p. 580–583. Disponível em: <http://dx.doi.org/doi:10.1109/ICOIN48656.2020.9016500>. 30

MOLNAR, S.; MOCZAR, Z. Three-dimensional characterization of internet flows. In: *2011 IEEE International Conference on Communications (ICC)*. [s.n.], 2011. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/icc.2011.5963476>. 59

MOURA, J.; HUTCHISON, D. Game theory for multi-access edge computing: Survey, use cases, and future trends. *IEEE Communications Surveys Tutorials*, v. 21, n. 1, p. 260–288, 2019. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2018.2863030>. 42

MUBARAKALI, A.; ALQAHTANI, A. S. A survey: Security threats and countermeasures in software defined networking. In: *2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*. [s.n.], 2019. p. 180–185. Disponível em: <http://dx.doi.org/doi:10.1109/INFOCT.2019.8711319>. 29

MüNZ, G.; LI, S.; CARLE, G. Traffic anomaly detection using kmeans clustering. In: *In GI/ITG Workshop MMBnet*. [S.l.: s.n.], 2007. 61

NASH, J. F. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, National Academy of Sciences, v. 36, n. 1, p. 48–49, 1950. ISSN 0027-8424. Disponível em: <http://dx.doi.org/doi:10.1073/pnas.36.1.48>. 42

NOVAES, M. P.; CARVALHO, L. F.; LLORET, J.; PROENÇA, M. L. Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment. *IEEE Access*, v. 8, p. 83765–83781, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2020.2992044>. 51

ONF. Openflow switch specification 1.5.1. *tech. rep., Open Network Foundation*, 2015. Disponível em: <https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf>. 55

OSBORNE, M. J.; RUBINSTEIN, A. *A Course in Game Theory*. [S.l.]: The MIT Press, 1994. v. 1. (MIT Press Books, 0262650401). ISBN ARRAY(0x49f52ed8). 42

PANTELEEV, J.; GAO, H.; JIA, L. Recent applications of machine learning in medicinal chemistry. *Bioorganic & Medicinal Chemistry Letters*, v. 28, n. 17, p. 2807 – 2815, 2018. ISSN 0960-894X. Disponível em: <http://dx.doi.org/doi:10.1016/j.bmcl.2018.06.046>. 46

PARASHAR, M.; POONIA, A.; SATISH, K. A survey of attacks and their mitigations in software defined networks. In: *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. [s.n.], 2019. p. 1–8. Disponível em: <http://dx.doi.org/doi:10.1109/ICCCNT45670.2019.8944621>. 35, 36

PATEL, P.; BANSAL, D.; YUAN, L.; MURTHY, A.; GREENBERG, A.; MALTZ, D. A.; KERN, R.; KUMAR, H.; ZIKOS, M.; WU, H.; KIM, C.; KARRI, N. Ananta: Cloud scale load balancing. In: *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*. New York, NY, USA: Association for Computing Machinery, 2013. (SIGCOMM '13), p. 207–218. ISBN 9781450320566. Disponível em: <http://dx.doi.org/doi:10.1145/2486001.2486026>. 36

POUYANFAR, S.; SADIQ, S.; YAN, Y.; TIAN, H.; TAO, Y.; REYES, M. P.; SHYU, M.-L.; CHEN, S.-C.; IYENGAR, S. S. A survey on deep learning: Algorithms, techniques, and applications. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 51, n. 5, p. 92:1–92:36, set. 2018. ISSN 0360-0300. Disponível em: <http://dx.doi.org/doi:10.1145/3234150>. 47

PROENÇA, M. L.; COPPELMANS, C.; BOTTOLI, M.; ALBERTI, A.; MENDES, L. S. The hurst parameter for digital signature of network segment. In: SOUZA, J. N. de; DINI, P.; LORENZ, P. (Ed.). *Telecommunications and Networking - ICT 2004*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 772–781. ISBN 978-3-540-27824-5. Disponível em: <http://dx.doi.org/doi:10.1007/978-3-540-27824-5_103>. 37

PROENÇA, M. L.; ZARPELAO, B. B.; MENDES, L. S. Anomaly detection for network servers using digital signature of network segment. In: *Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop (AICT/SAPIR/ELETE'05)*. [s.n.], 2005. p. 290–295. Disponível em: <http://dx.doi.org/doi:10.1109/AICT.2005.26>. 37

PUNDIR, S.; WAZID, M.; SINGH, D. P.; DAS, A. K.; RODRIGUES, J. J. P. C.; PARK, Y. Intrusion detection protocols in wireless sensor networks integrated to internet of things deployment: Survey and future challenges. *IEEE Access*, v. 8, p. 3343–3363, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2019.2962829>. 29

QIN, G.; CHEN, Y.; LIN, Y. Anomaly detection using lstm in ip networks. In: *2018 Sixth International Conference on Advanced Cloud and Big Data (CBD)*. [s.n.], 2018. p. 334–337. ISSN null. Disponível em: <http://dx.doi.org/doi:10.1109/CBD.2018.00066>. 47, 61

QIU, J.; WU, Q.; DING, G.; XU, Y.; FENG, S. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, v. 2016, 12 2016. Disponível em: <http://dx.doi.org/doi:10.1186/s13634-016-0355-x>. 43

RAFIQUE, W.; QI, L.; YAQOOB, I.; IMRAN, M.; RASOOL, R. u.; DOU, W. Complementing iot services through software defined networking and edge computing: A comprehensive survey. *IEEE Communications Surveys Tutorials*, p. 1–44, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2020.2997475>. 30

RAMASWAMY, S.; RASTOGI, R.; SHIM, K. Efficient algorithms for mining outliers from large data sets. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2000. (SIGMOD '00), p. 427–438. ISBN 1-58113-217-4. Disponível em: <http://dx.doi.org/doi:10.1145/342009.335437>. 61

ROHRMANN, R. R.; ERCOLANI, V. J.; PATTON, M. W. Large scale port scanning through tor using parallel nmap scans to scan large portions of the ipv4 range. In: *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. [s.n.], 2017. p. 185–187. Disponível em: <http://dx.doi.org/doi:10.1109/ISI.2017.8004906>. 30

ROOPAK, M.; TIAN, G. Y.; CHAMBERS, J. Deep learning models for cyber security in iot networks. In: *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. [s.n.], 2019. p. 0452–0457. Disponível em: <http://dx.doi.org/doi:10.1109/CCWC.2019.8666588>. 31

ROY, S.; ELLIS, C.; SHIVA, S.; DASGUPTA, D.; SHANDILYA, V.; WU, Q. A survey of game theory as applied to network security. In: *2010 43rd Hawaii International Conference on System Sciences*. [s.n.], 2010. p. 1–10. Disponível em: <http://dx.doi.org/doi:10.1109/HICSS.2010.35>. 38, 39

SAMPATH, N.; JERLIN, M. A.; KRITHIKA, L. B.; ANITHA, A. Intrusion detection in software defined networking using genetic algorithm. In: *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. [s.n.], 2020. p. 1–5. Disponível em: <http://dx.doi.org/doi:10.1109/ic-ETITE47903.2020.464>. 32

SCARANTI, G. F.; CARVALHO, L. F.; BARBON, S.; PROENÇA, M. L. Artificial immune systems and fuzzy logic to detect flooding attacks in software-defined networks. *IEEE Access*, v. 8, p. 100172–100184, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2020.2997939>. 37

SCOTT-HAYWARD, S.; NATARAJAN, S.; SEZER, S. A survey of security in software defined networks. *IEEE Communications Surveys Tutorials*, v. 18, n. 1, p. 623–654, 2016. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2015.2453114>. 36

SHANNON, C. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 5, n. 1, p. 3–55, jan. 2001. ISSN 1559-1662. Disponível em: <http://dx.doi.org/doi:10.1145/584091.584093>. 60

SHARAFALDIN, I.; LASHKARI, A. H.; GHORBANI, A. A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: INSTICC. *Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,*. SciTePress, 2018. p. 108–116. ISBN 978-989-758-282-0. Disponível em: <http://dx.doi.org/doi:10.5220/0006639801080116>. 77

SHARAFALDIN, I.; LASHKARI, A. H.; HAKAK, S.; GHORBANI, A. A. Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In: *2019 International Carnahan Conference on Security Technology (ICCST)*. [s.n.], 2019. p. 1–8. ISSN 1071-6572. Disponível em: <http://dx.doi.org/doi:10.1109/CCST.2019.8888419>. 73

SHRIVASTAVA, R. K.; RAMAKRISHNA, S.; HOTA, C. Game theory based modified naïve-bayes algorithm to detect dos attacks using honeypot. In: *2019 IEEE 16th India Council International Conference (INDICON)*. [s.n.], 2019. p. 1–4. Disponível em: <http://dx.doi.org/doi:10.1109/INDICON47234.2019.9030355>. 42

SILVA, I. D.; SPATTI, D.; FLAUZINO, R. *REDES NEURAIS ARTIFICIAIS PARA ENGENHARIA E: CIENCIAS APLICADAS - CURSO PRATICO*. [S.l.]: ARTLIBER, 2010. ISBN 9788588098534. 44

SINGH, K.; DE, T. MLP-GA based algorithm to detect application layer DDoS attack. *Journal of Information Security and Applications*, Elsevier Ltd, v. 36, p. 145–153, 2017. ISSN 2214-2126. Disponível em: <http://dx.doi.org/doi:10.1016/j.jisa.2017.09.004>. 46

SMITH-PERRONE, J.; SIMS, J. Securing cloud, sdn and large data network environments from emerging ddos attacks. In: *2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence*. [s.n.], 2017. p. 466–469. Disponível em: <http://dx.doi.org/doi:10.1109/CONFLUENCE.2017.7943196>. 32

SOCHER, R.; LIN, C. C.-Y.; NG, A. Y.; MANNING, C. D. Parsing natural scenes and natural language with recursive neural networks. In: *Proceedings of the 28th International Conference on International Conference on Machine Learning*. USA: Omnipress, 2011. (ICML'11), p. 129–136. ISBN 978-1-4503-0619-5. Disponível em: <http://dx.doi.org/doi:10.5555/3104482.3104499>. 47

SONG, S.; PARK, H.; CHOI, B.; CHOI, T.; ZHU, H. Control path management framework for enhancing software-defined network (sdn) reliability. *IEEE Transactions on Network and Service Management*, v. 14, n. 2, p. 302–316, 2017. Disponível em: <http://dx.doi.org/doi:10.1109/TNSM.2017.2669082>. 41

SONG, X.; JIANG, W.; LIU, X.; LU, H.; TIAN, Z.; DU, X. A survey of game theory as applied to social networks. *Tsinghua Science and Technology*, v. 25, n. 6, p. 734–742, 2020. Disponível em: <http://dx.doi.org/doi:10.26599/TST.2020.9010005>. 42

SUN, D.; FU, M.; ZHU, L.; LI, G.; LU, Q. Non-intrusive anomaly detection with streaming performance metrics and logs for devops in public clouds: A case study in aws. *IEEE Transactions on Emerging Topics in Computing*, v. 4, n. 2, p. 278–289, April 2016. ISSN 2376-4562. Disponível em: <http://dx.doi.org/doi:10.1109/TETC.2016.2520883>. 32, 59

SUNG, K.; HSIAO, S. Mitigating ddos with pow and game theory. In: *2019 IEEE International Conference on Big Data (Big Data)*. [s.n.], 2019. p. 6223–6225. Disponível em: <http://dx.doi.org/doi:10.1109/BigData47090.2019.9006081>. 31, 42

TEDJOPURNOMO, D. A.; BAO, Z.; ZHENG, B.; CHOUDHURY, F.; QIN, A. K. A survey on modern deep neural network for traffic prediction: Trends, methods and challenges. *IEEE Transactions on Knowledge and Data Engineering*, p. 1–1, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/TKDE.2020.3001195>. 46

WANG, F.; ZHANG, M.; WANG, X.; MA, X.; LIU, J. Deep learning for edge computing applications: A state-of-the-art survey. *IEEE Access*, v. 8, p. 58322–58336, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2020.2982411>. 31, 46

WANG, Y.; AN, J.; HUANG, W. Using cnn-based representation learning method for malicious traffic identification. In: *2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*. [s.n.], 2018. p. 400–404. Disponível em: <http://dx.doi.org/doi:10.1109/ICIS.2018.8466404>. 50

WEHBI, K.; HONG, L.; AL-SALAH, T.; BHUTTA, A. A. A survey on machine learning based detection on ddos attacks for iot systems. In: *2019 SoutheastCon*. [s.n.], 2019. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/SoutheastCon42311.2019.9020468>. 30

WIJNHOVEN, R. G. J.; WITH, P. H. N. de. Fast training of object detection using stochastic gradient descent. In: *2010 20th International Conference on Pattern Recognition*. [s.n.], 2010. p. 424–427. ISSN 1051-4651. Disponível em: <http://dx.doi.org/doi:10.1109/ICPR.2010.112>. 61

WU, Q.; SHIVA, S.; ROY, S.; ELLIS, C.; DATLA, V. On modeling and simulation of game theory-based defense mechanisms against dos and ddos attacks. In: *Proceedings of the 2010 Spring Simulation Multiconference*. San Diego, CA, USA: Society for Computer Simulation International, 2010. (SpringSim '10). ISBN 9781450300698. Disponível em: <http://dx.doi.org/doi:10.1145/1878537.1878703>. 42

WU, W.; LI, R.; XIE, G.; AN, J.; BAI, Y.; ZHOU, J.; LI, K. A survey of intrusion detection for in-vehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, v. 21, n. 3, p. 919–933, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/TITS.2019.2908074>. 30

XIA, W.; WEN, Y.; FOH, C. H.; NIYATO, D.; XIE, H. A survey on software-defined networking. *IEEE Communications Surveys & Tutorials*, v. 17, n. 1, p. 27–51, 2015. ISSN 1553-877X. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2014.2330903>. 29, 36

XUANYUAN, M.; RAMSURRUN, V.; SEEAM, A. Detection and mitigation of ddos attacks using conditional entropy in software-defined networking. In: *2019 11th International Conference on Advanced Computing (ICoAC)*. [s.n.], 2019. p. 66–71. Disponível em: <http://dx.doi.org/doi:10.1109/ICoAC48765.2019.246818>. 32

YAN, Q.; YU, F. R.; GONG, Q.; LI, J. Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE Communications Surveys Tutorials*, v. 18, n. 1, p. 602–622, 2016. Disponível em: <http://dx.doi.org/doi:10.1109/COMST.2015.2487361>. 35

YANG, H.; LIANG, S.; NI, J.; LI, H.; SHEN, X. Secure and efficient knn classification for industrial internet of things. *IEEE Internet of Things Journal*, 2020. Disponível em: <http://dx.doi.org/doi:10.1109/JIOT.2020.2992349>. 43

YIN, D.; ZHANG, L.; YANG, K. A ddos attack detection and mitigation with software-defined internet of things framework. *IEEE Access*, v. 6, p. 24694–24705, 2018. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2018.2831284>. 32

YIN, X.; LI, X.; ZHANG, Y.; ZHANG, T.; LU, C.; AI, Q.; LI, Z.; SUN, Z. A survey of deep learning and its application in distribution network. In: *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. [s.n.], 2020. p. 643–646. Disponível em: <http://dx.doi.org/doi:10.1109/ICAIIC48513.2020.9065235>. 46

ZANG, X.; GONG, J.; HU, X. An adaptive profile-based approach for detecting anomalous traffic in backbone. *IEEE Access*, v. 7, p. 56920–56934, 2019. Disponível em: <http://dx.doi.org/doi:10.1109/ACCESS.2019.2914303>. 32

ZHANG, Y.; YANG, Q.; LAMBOTHARAN, S.; KYRIAKOPOULOS, K.; GHAFIR, I.; ASSADHAN, B. Anomaly-based network intrusion detection using svm. In: *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*. [s.n.], 2019. p. 1–6. Disponível em: <http://dx.doi.org/doi:10.1109/WCSP.2019.8927907>. 43

APÊNDICE A - *A Game Theoretical Based System Using Holt-Winters and Genetic Algorithm With Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks*

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 1, 2017, accepted April 21, 2017, date of publication May 9, 2017, date of current version June 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2702341

A Game Theoretical Based System Using Holt-Winters and Genetic Algorithm With Fuzzy Logic for DoS/DDoS Mitigation on SDN Networks

MARCOS V. O. DE ASSIS[1], ANDERSON H. HAMAMOTO[2], TAUFIK ABRÃO[3], (Senior Member, IEEE), AND MARIO LEMES PROENÇA JR.[2]

[1]Engineering and Exact Department, Federal University of Paraná, Palotina 85.950-000, Brazil
[2]Computer Science Department, State University of Londrina, Londrina 86.057-970, Brazil
[3]Department of Electrical Engineering, State University of Londrina, Londrina 86.057-970, Brazil

Corresponding author: Marcos V. O. de Assis (marcos.assis@ufpr.br)

This work was supported in part by the National Council for Scientific and Technological Development of Brazil under Grant 308348/2016-8 and Grant 304066/2015-0, in part by the Federal University of Paraná under Project Banpesq/2014016797 and in part by the SETI/Fundção Araucária for project 35987 under call 20/1012.

ABSTRACT The ever expanding the usage of cloud computing environments, connected applications and Internet of Things-based devices have progressively increased the amount of data that travels through our networks. Software-defined network (SDN) is an emergent paradigm that aims to support next-generation networks through its flexible and powerful management mechanisms. One of the biggest threats faced by these services nowadays is security management. Attacks based on the denial of service (DoS) are particularly efficient against this paradigm due to its centralized control characteristic. Once this controlling system receives a massive amount of malicious requests, the overall performance of the network operation is impaired. Although several researches propose to address this problem, most of them are reactive approaches, detecting the attacks and warning the network administrators, i.e., after the network is already compromised. This paper presents an autonomic DoS/DDoS defensive approach for SDNs called Game Theory (GT)-Holt-Winters for Digital Signature (HWDS), which unites the anomaly detection and identification provided by an HWDS system with an autonomous decision-making model based on GT. Real collected data and simulated attacks are used by the system to measure its effectiveness and efficiency. Furthermore, we also use a heuristic Fuzzy-GADS method for anomaly detection instead of HWDS, aiming to compare the achieved performance and evaluate the behavior of the presented game theoretical approaches a standalone mitigation module.

INDEX TERMS Game theory, HWDS, fuzzy logic, GADS, denial of service.

I. INTRODUCTION

As large-scale computer networks continuously grow in size and complexity, efficient and fast-responding management mechanisms are required now more than ever. The popularization of network technology is giving birth to a large number of useful online applications, such as Internet of Things (IoT) devices and Cloud Computing environments. As a result, the amount of relevant and valuable information traveling among large-scale networks has increased substantially in the last decade [1].

Therefore, people are increasingly dependent on online services to perform daily tasks. The advantage is that, as long as one has access to the service network, it is possible to generate and acquire information in a simple, ubiquitous and agile way. On the other hand, if the network services are unavailable, anenormous amount of end-users are negatively impacted. Thus, an efficient network management approach is needed to guarantee the availability and quality of the services provided by the network.

In that manner, Software-Defined Network (SDN) emerges as a powerful and flexible networking architecture. It was developed to simplify and improve the management process through better abstractions for network functions and more flexibility for controlling network devices. Ontraditional networks, both packet-forwarding (data plane) and routing tasks (control plane) are performed by routers and switches. The SDN basically separates the control plane from the data plane, *i.e.*, the network control process of the packet-forwarding plane is implemented in a software level by a centralized and

programmable controller [2], [3], instead of being controlled by network's routers.

There are three main groups of anomalous events that may impair a SDN operation [3]: attacks directed to the control plane; compromising of the communication between data and control planes, and; threats to data plane's equipment.

In this paper, we address the first cited anomalous event, in which Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks'mitigation represent a major challenge for this network architecture. DDoS attacks usually use botnets as attacking hosts, which are a collection of several malware-infected machines remotely controlled by a malicious user [1]. This characteristic significantly improves the impact of these attacks, as the number of infected hosts inside botnets is usually massive. In these attacks, malicious requests overwhelm the central controlling system, consequently hindering the SDN operations. Recently, two large-scale DDoS attacks targeted the governments of the United States and South Korea, demonstrating that even though these attacks are widely addressed in the literature we still lack efficient mechanisms to detect, identify and mitigate them [4].

Computer network security is a well-addressed research area, with several different proposed approaches to detect [5], [6] and identify attacks [7]. For instance, the Holt-Winters for Digital Signature (HWDS) system [8], [9] is capable of detecting and identifying several different kinds of anomalies, such as DoS, DDoS and Port Scan attacks, through a seven-dimensional flow analysis and traffic characterization process. However, most of these works only detect and provide relevant information to the network administrator, who has to supervise the required counter-measures manually.

To mitigate the effects of these attacks, simple rules, such as blocking traffic from suspicious IP addresses [10] or the entire suspension of communication with the attacked service, may be defined at the network entry switch [11]. However, routing rules set by human operators usually impair end-users. It occurs due to their inflexibility and lack of adaptation regarding normal traffic behavior, as well as the low human response-time and error-prone actions. Thus, a new management approach known as Autonomic Management [12] is necessary as a result of demands for new automatic control mechanisms able to not only detect and identify network anomalies and attacks but also take countermeasures to mitigate their effect efficiently.

Thus, to develop an autonomic SDN defense system, an efficient and fast decision-making method must be used. One of the most promising methods in the area is the Game Theory (GT). Widely used in economics and resource allocation, it has been increasingly addressed in network management applications aiming to optimize responses and actions [13]–[15]. Briefly, a GT-based method consists of converting a problem with conflicting interests into a game, where different players can take actions trying to optimize the results of acquiring their objectives.

In this paper, we propose a set of procedure called Game Theoretical Based System for DoS/DDoS Mitigation using Holt-Winters (GT-HWDS) and Genetic Algorithm with Fuzzy Logic (GT-Fuzzy-GADS) directly applicable to a SDN network.Such a supervision system is capable of autonomously detect, identify and mitigate the occurrence of DoS and DDoS attacks to SDNs through the use of a game theorymodel. To measure the efficiency of the presented system, IP flow records are collected from a real environment similar to a SDN. This data is used alongside a Network Anomaly Simulator called Scorpius [16], which can inject anomalous flows into real ones to simulate attacks such as DoS and DDoS. Furthermore, we measure the results and evaluate the behavior of the proposed game theoretical approach as a standalone mitigation module using another base method instead of HWDS. For this purpose, we used the Fuzzy-GADS method, an approach based on Genetic Algorithm (GA) and Fuzzy Logic for network anomaly detection [17].

The remainder of this paper is composed of the following sections: Section II presents the related works; Section III introduces the proposed GT-HWDS system, as well as important Game Theory concepts; Section IV describes Fuzzy-GADS method, which is used instead of HWDS for comparison purpose and behavior analysis of the proposed game theoretical approach; Section Vanalyses the performed tests and numerical results; Finally, Section VIoffers the main conclusions of the paper and future work projects.

II. RELATED WORK

The evolution of network-based applications and data exchange is massively increasing the amount of traffic that computer networks transport. The conventional communication system, although it provides an easy-to-manage environment, it is becoming impractical due to its robust architecture. Thus, Software Defined Networks (SDNs) have been developed to provide a flexible and powerful architecture for next-generation networks, where it is possible to dynamically allocate the available bandwidth according to the current network necessity. Several works in the area have been developed, such as [18], where the authors propose a formal network model used to detect anomalies caused by the interference between two or more functions or policies of the network, which is called inter-function anomalies. Li *et al.* [19] propose a new control plane management method called CPMan, aiming to reduce the overhead caused by the management messages in large scale SDNs. Song *et al.* [20] propose and develop a control path management framework to enhance the reliability of SDN through the issues identified by extensive analysis. Poulios *et al.* [21] present a study of the relationship between Autonomic Network Management and SDN through the prism of Long Term Evolution (LTE) Self-Organizing Networks (SONs), highlighting how these different paradigms interact with and complement each other.

One of the most important aspects of SDN is security. Even though there are currently several different kinds of network attacks and exploits, the DoS and DDoS attacks are the most common due to their simplicity and power [22]. Furthermore, these attacks are particularly effective against SDNs due to its centralized architecture controller. Long *et al.* [23] analyze the impact caused by DoS attacks in remote controlled systems, also discussed mitigation methods. Hoque *et al.* [1], highlight that Botnet DDoS attacks are catastrophic to the victim network, and present a survey on the matter.

To secure network systems from these attacks, several approaches have been proposed. Tan *et al.* [22] propose a system to detect DoS attacks through Multivariate Correlation Analysis. Carvalho *et al.* [24] propose an anomaly detection and identification system based on IP flow analysis using a modification of the Ant Colony Optimization metaheuristic to improve the characterization process.

As SDN is a new network architecture, conventional security mechanisms mustadapt their operation to this new paradigm. This adaptation is not a simple task since most of the traditional defense systems use the measurement of the network behavior based on its static architecture. Röpke and Holz [25] propose a sandbox system that allows the restriction of SDN applications and internal Network Operating System (NOS) components to only access a configurable set of critical operations. According to the authors, this approach is necessary due to the danger of significant impairments and failures due to the power that NOS have over SDNs. Furthermore, Lara and Ramamurthy [26] propose OpenSec, a security framework based on OpenFlow that allows the network administrator to create and implement security policies in a human-readable language. The authors highlight that 95% of the tested attacks were detected through the usage of this framework and its policies.

One most desirable characteristic of network defense mechanisms is the capability to autonomously take decisions to mitigate the impact caused by attacks and anomalies. Mainly applied to economics, the Game Theory method is increasingly gaining space among network management approaches due to its power on decision-making processes and high efficiency on optimizing outcomes. Bruce [27] explains the basics of Game Theory and highlights the mechanisms used for its application to big data analytics and decision making of remote sensing and geosciences field. Hamdi and Abie [15] propose a game-based model for adaptive security in IoT paradigm, focusing on eHealth systems. In [28] and [29], Game theoretical models against DoS and DDoS attacks have been proposed, where the attacker aims to impair the network operation, and the defender counteracts to optimize the firewall configuration. Particularly Bedi *et al.* [29] present a defense framework called GIDA, which not only drop packages from potentially malicious hosts but also redirect a part of them to a honeypot for further analysis of the attack.

In this paper, we propose a supervising system capable of not only rapidly detecting and identifying network anomalies and attacks such as DoS and DDoS but also autonomously taking countermeasures to mitigate them. The framework of the presented system is similar to the one proposed by Bedi *et al.* [29]. However, unlike the work developed by the authors, this paper presents a system based on a deeper search over the characteristics of the attacks through the HWDS method, which provides a more precise detection and identification of DoS and DDoS attacks. Furthermore, DoS and DDoS attacks based on UDP protocol are analyzed, which represent a much stealthier kind of denial of service since the bandwidth of the network is barely impacted. Finally, we use IP flow records collected from a real large-scale network environment along with simulated attacks to evaluate the effectiveness and efficiency of the proposed supervising system, instead of using probability distributions to model the behavior of the legitimate network users.

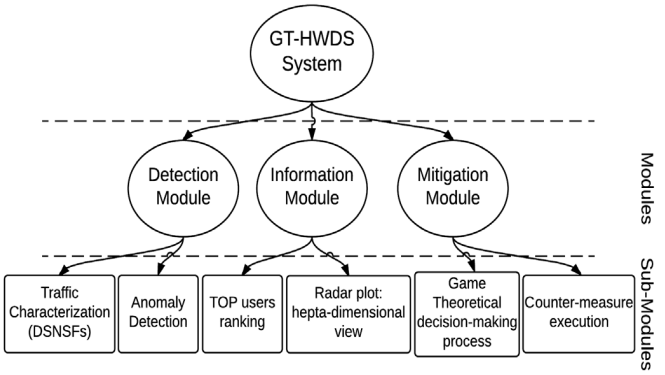


FIGURE 1. GT-HWDS system.

III. PROPOSED GT-HWDS SYSTEM

The attack detection and mitigation system presented in this paper aredesignedover two main approaches. The first of them is the HWDS system, responsible for the detection and identification of anomalies/attacks. This system is based on the analysis and characterization of seven IP flow dimensions. The second one is the GT-based method, responsible for the selection of the optimal countermeasure for an attack. Thus, the GT-HWDS system can be defined as the interaction of 3 modules: Detection, Information and Mitigation modules, as depicted in Fig. 1.

The proposed system aims to mitigate DoS and DDoS attacks that occur on Software-Defined Networks (SDNs). As previously discussed, this flexible architecture separates the data plane (packet-forwarding process) and the control plane (routing tasks) of the network. Thus, packet forwarding is implemented in a software level by a central controller. DDoS attacks target this central controller, overwhelming it with a huge amount of package transmission. Thus, in order to protect SDNs, incoming data must be analyzed before entering the network environment, *i.e.*, at the gateway connected to a border router.

The network topology considered is the same as the one analyzed by Bedi *et al.* in [29], which consists of a

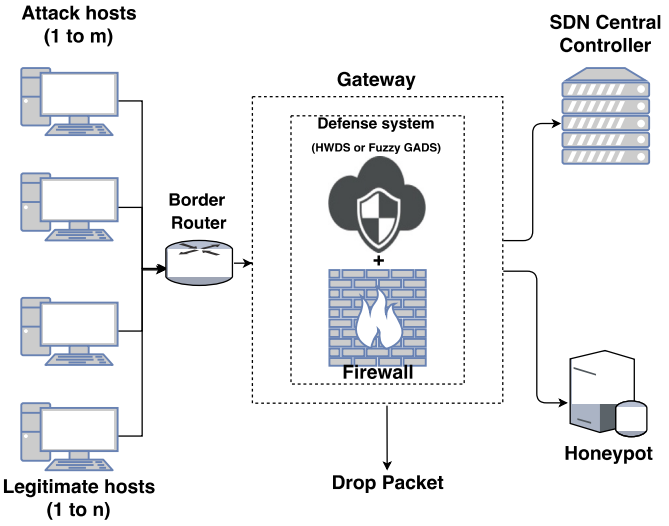


FIGURE 2. Network topology and defense system organization.

Gateway control using GT-HWDS System along with a Firewall to secure the connection between the Gateway controller with the SDN central controller. The Mitigation Module of GT-HWDS system decides, through a game theoretical approach, which packets should proceed to the SDN central controller, be redirected to a Honeypot or be dropped by the firewall. The network topology is represented in Fig. 2. It is important to highlight that the planes' construction, software and hardware requirements for the SDN architecture are not considered in this paper since the detection, identification and mitigation processes are performed before the entrance of data into the network. In other words, it is able to operate with any SDN configuration.

A. DETECTION AND INFORMATION MODULES

The HWDS system executes both detection and information modules. In this section, we briefly discuss the operation of the HWDS system. For detailed information, please refer to our previous works [8], [9]. The detection module is an approach that analyses seven IP flow dimensions in parallel to create a traffic characterization schema. The IP Flow dimensions analyzed are bits, packets and flows per second and the Shannon Entropy of the source and destination IP addresses and ports. To characterize the behavior of these dimensions, the system uses a modification of the statistical forecasting method Holt-Winters, namely Holt-Winters for Digital Signature (HWDS). Through the utilization of this procedure, the system generates a signature, specifically a Digital Signature of Network Segment using Flow analysis(DSNSF) for every analyzed dimension.

By a parallel analysis of the seven created DSNSFs, the HWDS system compares real collected IP flows with the signature generated every minute. If the observed traffic differs from the DSNSFs generated, then the system compares the current signature with the signature of known attacks, such as DoS and DDoS. At this point, the Information and the Mitigation Module are triggered.

The Information module provides relevant information about the anomaly detected to the network administrator and the mitigation module. It provides the TOP 3 (most relevant) elements of the dimensions: source and destination IP addresses, source and destination ports, and protocols, alongside a "Global View," a radar-plot of the network state at the detection moment, *i.e.*, the graphical signature of the attack/anomaly. Furthermore, this module stores the source IP addresses of the previous 5 minutes the analyzed time interval, to help differentiate legitimate users from malicious ones.

B. MITIGATION MODULE

The mitigation module is responsible for autonomously taking countermeasures to mitigate the impact of the DoS/DDoS attack. Through the use of a GT approach, the system analyzes a set of possible actions for both attacker and defense system, calculates the rewards and costs of every action and executes the optimal countermeasure.

Before we proceed to describe our GT approach, it is important to define some terms. According to [30], "Game Theory describes multi-person decision scenarios as games where each player chooses actions which results in the best possible reward for itself, while anticipating the rational actions from other players." In other words, it is a method for translating a real world problem into a game, where two or more players are trying to win. In our GT approach, two players play the game: the attacker (malicious user) and the defense mechanism. The objectives of the game are different for each player, but they complement each other. As the attacker tries to maximize the damage caused to the network and reduce its chance of being detected, the defense aims to reduce the impact posed by the attacker and preserve the network's normal operation.

The GT approach that composes the Mitigation Module of the presented system can be defined as a 4-tuple vector:

$$G = (A_{att}, A_{def}, P_{att}, P_{def}) \tag{1}$$

The elements of the vector G are detailed as follows.

The A_{att} element stands for the set of possible actions that the attacker can perform, *i.e.*, all the possible attacker's strategy. Two actions compose this set:

- Change the intensity (u) of the attack, *i.e.*, the number of packets per second directed to the network by each attacking node;
- Modify the number of attacking nodes (m).

Similarly, the A_{def} element stands for the set of possible actions that the defense can perform, *i.e.*, all the possible defense's strategy. Three actions compose this set:

- Allow packets to pass to the SDN central controller;
- Drop packets at the firewall to protect the SDN central controller through a particular dropping rate (D);
- Redirect packets to the Honeypot for further analysis of the attack behavior, motivation, and source.

It is important to highlight that only new users (new source IP addresses) are dropped or redirected to the Honeypot. New hosts can be identified due to the Information Module, which keeps a historical database of the last five minutes containing the source IP addresses of the hosts that used the network in this period. Thus, users that were accessing the network's service before the attack began do not have their packets dropped.

The elements P_{att} and P_{def} are the Payoffs or Utility Functions for the attacker and the defense, respectively. According to [30], Payoff is the positive or negative reward to a player for a given action within the game. The Payoffs of attacker and defense are usually represented by (2):

$$P = Reward - Cost \tag{2}$$

Thus, the Payoffs for the attacker and the defense mechanism can be respectively represented as (3) and (4):

$$P_{atk} = w_1^{atk} \cdot E - w_2^{atk} \cdot BC - w_3^{atk} \cdot AC + w_4^{atk} \cdot PL \tag{3}$$

$$P_{def} = -w_1^{def} \cdot E + w_2^{def} \cdot BC + w_3^{def} \cdot AC - w_4^{def} \cdot PL \tag{4}$$

where w^{atk} and w^{def} are weight parameters for each metric for the attacker and defense mechanism, respectively. Furthermore, the remaining variables stand for: E represents the normalized error between the expected network behavior and the current network state; BC is the average Bandwidth Consumption of the legitimate users in comparison to malicious ones; $AC$is the attack cost for the attacker, and; PL is the estimated Packet Loss of legitimate users through the packetdropping during the mitigation process.

The metric of normalizer Error E is defined in (5) and it is calculated by comparing the number of packages that was expectedPkt_{exp} at the current time interval (Signature or DSNSF of the Packets/s dimensions, calculated by the HWDS method) with the number of packets observed after the mitigation process. Furthermore, the resulting value should be normalizedby the maximum between expected and observed packets aimingto normalize the metric to interact with the other cost/reward functions. Finally, the absolute value of the normalized error is considered to ease the error minimization problem, since dropping an excessive number of packets will also negatively impact the network operation (optimal value achieved when $E = 0$).

$$E = \left| \frac{Pkt_{end} - Pkt_{\exp}}{\max(Pkt_{end}, Pkt_{\exp})} \right| \tag{5}$$

where the packets observed after the mitigation process is provided by:

$$Pkt_{end} = Pkt_{leg} + Pkt_{new} * (1 - D) \tag{6}$$

withPkt_{leg} being the number of packets of legitimate users, known through the analysis performed by the Information Module, as previously described. Pkt_{new} is the number of packets from new users, which merges new legitimate users and malicious users. Finally, D is the dropping rate of new packages, varying from 0 to 100%.

The average Bandwidth Consumption (BC) can be calculated using data provided by the Information Module and observed at the current time interval. First of all, the bandwidth difference (7) between the expected number of bits and the observed number of bits can be expressed by:

$$d_{bits} = B_{\exp} - B_{obs}, \quad where \tag{7}$$

$$B_{obs} = B_{leg} + B_{new} \tag{8}$$

With the difference d_{bits} calculated, the proportion Pb_{leg} (9) of bits from legitimate users among the total bits of new users can be determined:

$$Pb_{leg} = \frac{(B_{new} - d_{bits})}{B_{new}} \tag{9}$$

Finally, the average bandwidth consumption BC, measured in bits/s, can be calculated considering the drop rate of new packets D, the number of attacking hosts m and the intensity of the attack u:

$$BC = \frac{B_{leg} + D\left[B_{new} \cdot Pb_{leg} - m \cdot u \cdot (1 - Pb_{leg})\right]}{B_{leg} + D\,(B_{new} + m \cdot u)} \tag{10}$$

The Attack Cost AC is considered linear to the number of hosts m controlled by the attacker, as proposed by [20]. This number is normalized to ease the interaction with the other metrics. Thus, this parameter can be obtained through:

$$AC = \frac{m}{\max(m)} \tag{11}$$

Finally, the estimated Packet Loss ratePL can be achieved considering the expected and observed packets along with the attack parameters, including the number of attacking hosts m and intensity of the attack u, herein measured in packets/s), and the defense parameter D (dropping rate of new packages).First of all, the distance d_{pkt} is calculated through:

$$d_{pkt} = (Pkt_{leg} + Pkt_{new}) - Pkt_{\exp} \tag{12}$$

Then, we apply the result achieved by (12) to calculate the estimated packet proportion of legitimate users among the new packets:

$$Pp_{leg} = \frac{Pkt_{new} - d_{pkt}}{Pkt_{new}} \tag{13}$$

Thus, the result obtained through (13) can be applied in (14) to calculate the PL rate:

$$PL = 1 - \frac{Pkt_{leg} + (1 - D)Pkt_{new} \cdot Pp_{leg}}{Pkt_{\exp}} \tag{14}$$

At the end of the payoff's calculation, given by (3) and (4), the GT method generates a matrix containing the calculated payoffs of each different attack possibility (combination of different m and u values) with each defense strategy available (each possible value for D). The cells of this matrix are organized as a pair of information (P_{att}, P_{def}).

The optimal defense strategy must be a Nash Equilibrium obtained through the payoff matrix. Nash Equilibrium is a

steady situation where no rational player would choose to modify its strategy since any possible action would decrease its utility function. As described by Equations (3) and (4), the reward of a player is the cost of another. If the weight parameters w_i^{atk} and w_i^{def} are equal for all values of i, then the problem is configured as a zero-sum game [28]. According to [31], the Nash equilibrium of zero-sum games existsand can be achieved by transforming the problem into a linear optimization problem, which is solved by the Minimax theorem. For proof of the existence and the number of Nash Equilibrium on the analyzed problem, refer to Appendix I.

Finally, with the optimal defense strategy calculated, the GT-HWDS system performs the dropping and redirecting processes along with the network's Firewall. A certain percentage of the dropped flows may be redirected to a Honeypot for further analysis but, since the redirect rate does not influence the optimal defense strategy, this rate is out of the scope of this research, and will be addressed in future works.

The defense strategy is performed for one hour, and the network comes back to a normal state after that. This time interval was chosen because, even though the attack stops within a few minutes, the impact suffered by the network is small, as shown on Section V. If a new attack is detected within this period, the Mitigation Module will be triggered again, and a new optimal defense strategy will be calculated updating the defense parameters of the network server.

IV. FUZZY-GADS METHOD

To test the effectiveness of the proposed game theoretical approach as a standalone mitigation module for other anomaly detection mechanisms besides HWDS, we use it along with the Fuzzy-GADS method, a two-phase system, which is described in this section. The Genetic Algorithm (GA) is applied to generate the network characterization, namely DSNSF, and a Fuzzy Logic approach is used to determine if an anomaly is present in a given time interval.In this method, a six-dimensional analysis of IP flows is employed, instead of seven as described in the HWDS approach.

A. GA FOR DIGITAL SIGNATURE

The concept of Genetic Algorithm (GA) was first proposed by Holland [32] in 1972. GA is a meta-heuristic search approach successfully applied to optimization problems,mimicking the steps observed in Darwin's theory of evolution. This algorithm starts with an initial set of solutions and optimizes them through genetic operations (selection, crossover, mutation) until an acceptable solution is reached.

The presented GA for DSNSF generation [17] uses the network's flows records of the past four weeks. As an example, to generate the DSNSF of a given Monday, the previous four Mondays are analyzed and used as input to the GA to create the DSNSF. Using the information available through IP flows, a six-dimensional analysis is performed, and these dimensions are: bits per second; packets per second; IP source entropy; IP destination entropy; port source entropy, and; port destination entropy.

When a GA is designed to solve a problem, the implementation of the genetic operators are not the only fundamental parameters to be defined. There is also the chromosome encoding and the fitness function. In the deployed GA, a numerical encoding is used, as the DSNSF is a real value containing the expected behavior of the computer network in a time period. The fitness function defines how appropriate the solution is to the problem in question. In this case, the Euclidian Distance was used, represented by:

$$f = \sqrt{\sum_{i=0}^{n} (y - x_i)^2} \tag{15}$$

where y is the value of the chromosome, x_i represents each element of the input data i of that time interval and n is the number of inputs.

In most cases, the population generation is random and uses the scope of the problem as the parameter. The DSNSF generated deploying the GA strategy uses a lower and an upper boundary values, randomly generatedin that interval. It ensures that the values of the solutions are neither too low or too high.

To increase the diversity and chances to produce fitter solutions to the problem, the genetic operations are applied iteratively to the initial set of solutions. The selection methods widely used are roulette wheel and tournament [33]. Roulette wheel uses the fitness value of the chromosome as the parameter to determine the likelihood of selection. The tournament selection method compares the fitness of two or more chromosomes chosen at random, selecting the one with the best fitness. In the proposed GA, the tournament selection is applied, which has a lower computational cost.

The crossover operation uses the chromosomes chosen through selection to create new chromosomes. As the DSNSF is a value that best represents the network behavior in a given time interval, the crossover methods used is the mean between the selected chromosomes. After the crossover operation, the new chromosome has a chance to suffer mutation, which can increase or decrease a small value to itself. After a certain number of generations, the algorithm stops and returns the fittest solution, which is the DSNSF.

B. ANOMALY DETECTION USING FUZZY LOGIC

The application of Fuzzy Logic for network anomaly detection is justified for two main reasons, as observed by Wu and Banzhaf [34]. First, the information of network traffic is collected and measured through statistics, which includes some level of uncertainty and errors. Second, there is no clear boundary between what is a normal behavior from abnormal, adding another layer of uncertainty to the problem. Due to these reasons, Fuzzy Logic is appropriate for the context of network anomaly detection, which is a method known for its performance involving uncertainty and partial truths, as stated by Zadeh [35].

In set theory, the membership value represents the set to which a certain element belongs. In Boolean Logic, an element either belongs or not to a set (0 or 1). On the other hand, Fuzzy Logic uses membership values that usually range from 0 to 1, indicating degrees of membership. The membership degrees in Fuzzy Logic are assigned using a membership function, such as Gaussian, Generalized Bell, triangular and trapezoidal. In this work, the membership degree measures the anomaly score of each network attribute, used to decide if a time interval is anomalous.

The DSNSF and a threshold are used to calculate the anomaly scores. The thresholds indicate normal fluctuations of the normal behavior from the predicted. This approach assigns different weights to the data used to generate the DSNSF; the further it is from the date to be analyzed the less it will affect the threshold. The threshold defined as ϕ is calculated by:

$$\varphi_k = DSNSF_k \pm L\sigma_k \sqrt{\frac{\lambda}{(2-\lambda)}} \tag{16}$$

in which k is network dimension indexer, L is the width of the threshold, σ is the standard deviation of the input data and λ is the weight. The values used for L and λ are 3.0 and 0.25 respectively, as suggested by Montgomery [37].

The DSNSF value is generated as outputs for the GA,while the threshold is calculated through EWMA, a membership function is used to determine the anomaly score of each dimension in a time interval. The anomaly scorecan be calculated by:

$$\zeta_k = 1 - e^{\frac{-(x_k - DSNSF_k)^2}{x\varphi_k^2}} \tag{17}$$

where k is network attribute indexer, x is the real traffic value, DSNSF is the prediction and ϕ is the threshold.

The anomaly score of every attribute is aggregated using a sum, and an alarm is generated if the total score is higher than a cutoff value. It is important to highlight that the Fuzzy-GADS method uses a six-dimensional analysis of IP flows for DSNSFs generation, unlike the seven-dimensional analysis of the HWDS.The rules for alarm generation are given by:

$$\begin{aligned} Rule_1 &: IF \rightarrow \sum_{k=1}^{6} \zeta_k \geq \Gamma, \quad THEN \rightarrow \text{“}anomalous\text{”} \\ Rule_2 &: IF \rightarrow \sum_{k=1}^{6} \zeta_k < \Gamma, \quad THEN \rightarrow \text{“}normal\text{”} \end{aligned} \tag{18}$$

where the cutoff value Γ was defined using a precision-recall curve, with a dataset with five weekdays with injected anomalies, as depicted in Fig. 3. This curve varies the cutoff values and calculated the precision and recall for each of these values. The optimal cutoff value is defined as $argmax(P\Gamma + R\Gamma)$, where $P\Gamma$ is the precision and $R\Gamma$ is the recall for value Γ. In the test performed to determine Γ, the value achieved is 3.9644, which is used in the experiments shown in Section V. The network environment in which this test was conducted is also described in the following section.

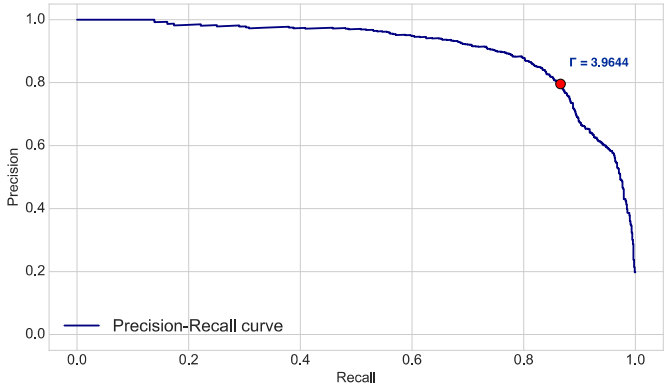


FIGURE 3. Precision-Recall curve for the estimation of Γ.

V. RESULTS AND ANALYSIS

To execute a performance analysis of the proposed system, we collected real IP flow data from the State University of Londrina (Brazil), which is a large-scale network composed of about 7000 different active hosts. The flows are collected by the usage of the sFlow protocol through a packet sampling scale of 1:512 due to the high data traffic volume. As the presented defense system operates directly at the network's gateway, the fact that the tested network is not a real SDN is not relevant for the results since the mitigation process occurs before any packet arrives at the SDN central controller.

The collected days are related to the Wednesdays of August 2015. These days represent a state of normal behavior of the network and were chosen arbitrarily due to the fact that this behavior was detected on all other collected days. Furthermore, HWDS and Fuzzy-GADS models need only three days of history to generate the DSNSFs that represent the network's normal behavior. Since, during the collection period, no DoS or DDoS attacks occurred on the analyzed network, we used an anomaly simulator called Scorpius [16] to inject the anomalous behavior of these attacks into real IP flow data.UDP-based denial of service attacks are harder to detect and less common than TCP-based DoS/DDoS. Given that the goal of our work is to detect and mitigate anomalous behaviors, we focused on a common threat to network security using a stealthier approach, *i.e.*, UDP-based denial of services.

The attacks were performed to simulate a denial process over a DNS/Cache server, the most accessed address into this network. For the DDoS tests, several intensities were discussed, using 512, 1024, 2560 and 5120 different hosts simultaneously attacking the selected target with UDP packets (experiments 1 to 4). For the DoS test, a single IP address of origin transmits a high amount of UDP packets over the selected target (experiment 5). All the attacks began at 14:00 and were finished at 14:30, to test the impact of the filters over non-anomalous hosts (from 14:30 to 15:00).

Before proceeding to the complete system analysis using both HWDS and Fuzzy-GADS, we carried out a detection performance analysis of the methods using Accuracy and Precision metrics. In classification problems, Accuracy measures

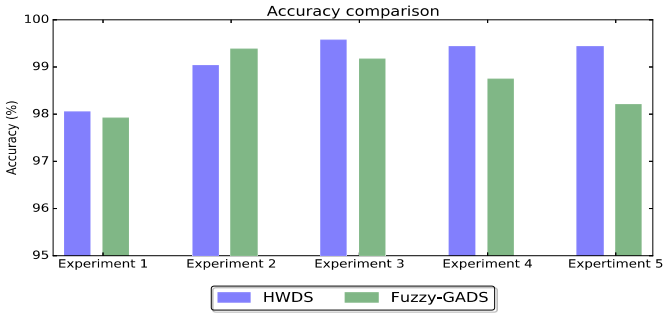


FIGURE 4. Accuracy rates for anomaly detection of HWDS and Fuzzy-GADS.

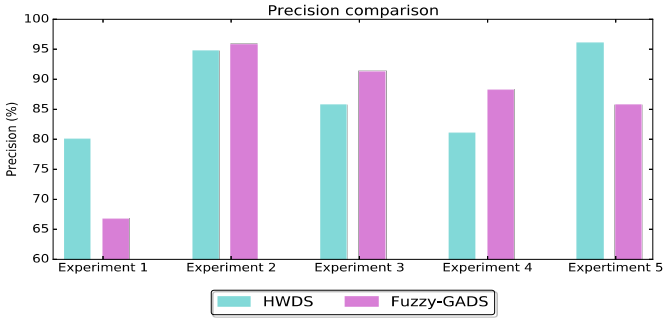


FIGURE 5. Precision rates for anomaly detection of HWDS and Fuzzy-GADS.

the overall capability for correctlyclassifying the samples in the test set, both anomalous and normal behaviors. On the otherhand, Precision measures the percentage of samples classified as anomalies (alarmsgenerated by the system) are in fact anomalous. The results achieved byboth methods are shown in Fig. 4 and 5.

As observed, both methods achieved good results for both Accuracy and Precision metrics. Regarding Accuracy, HWDS and Fuzzy-GADS attained Accuracy higher than 98% for most of the experiments. Concerning Precision, Fuzzy-GADS presented an inferior performance in comparison with HWDS, especially for Experiments 1 and 5. In contrast, Fuzzy-GADS obtained better Precision for Experiments 3 and 4, with similar results in Experiment 2. In general, both methods displayed good results for anomaly detection concerning Accuracy and Precision metrics.

To demonstrate the presented Game Theoretical approach for decision-making process, we will discuss in detail one of the result tests, namely the DDoS attack using 2560 different attacking hosts (Experiment 3).

As previously mentioned, the attacks were performed at 14:00. The Detection and Information Modules of the GT-HWDS system, as well as the Detection Module provided by the Fuzzy-GADS model, triggered an alarm on the 841-minute of the day, *i.e.*, at 14:01. As the GT-HWDS can identify the attack as a DDoS, itprovides relevant information to the Mitigation Module, while GT-Fuzzy-GADS only trigger an alarm pointing out the occurrence of an anomaly. The information received by the Mitigation Module through its parameters,using HWDS, include:

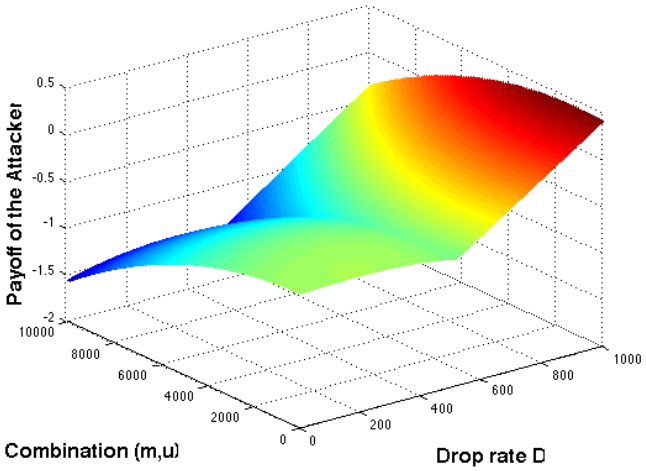


FIGURE 6. An example of Mesh plot for the variables D, m, u and P_{atk} used for Nash Equilibrium calculation.

- the DSNSFs (expected behavior) for bits and packets per second;
- a list of legitimate hosts (source IP addresses of flows from 5 minutes before the alarm triggering) that cannot be dropped on the mitigation process;
- the number of packets and bits belonging to legitimate hosts on the analyzed time interval (members of the previous list);
- the number of packets and bits belonging to unknown hosts (legitimate and potentially malicious hosts) on the analyzed time interval;

Since Fuzzy-GADS method does not use an Information Module, it provides the Mitigation Module with:

- the DSNSFs (expected behavior) for bits and packets per second;
- the number of packets and bits belonging to unknown hosts (legitimate and potentially malicious hosts) on the analyzed time interval;

Once the parameters are sent to the Mitigation Module, it triggers the GT-based decision-making sub-module. As discussed in Section III, the game is modeled as a one-shot game where the Defense System needs to choose a Dropping rate D based on the number of attacking hosts m and the intensity u of these attacks. For implementation purposes, we set the ranges for $D_{\%}$ from 0 to 100% with a step size of 0.1%, for m from 1 to 100 and for $u_{\%}$ from 1 to 100%. For the variable m, we consider its value representing the number of different attacking hosts per minute on the average, and, for variable $u_{\%}$, its value represents the intensity of the attack from 1 to 100% of the hosts' transmission capacity. The possible values for D are stored in an array with 1000 elements. Since both variables m and u interact with each other in the game, we stored them into an array with 10000 elements containing the combinatorial of them in the form (m, u).

By setting the weight parameters w^{atk} and w^{def} from (3) and (4) with the value 1, we assure that all the Reward and

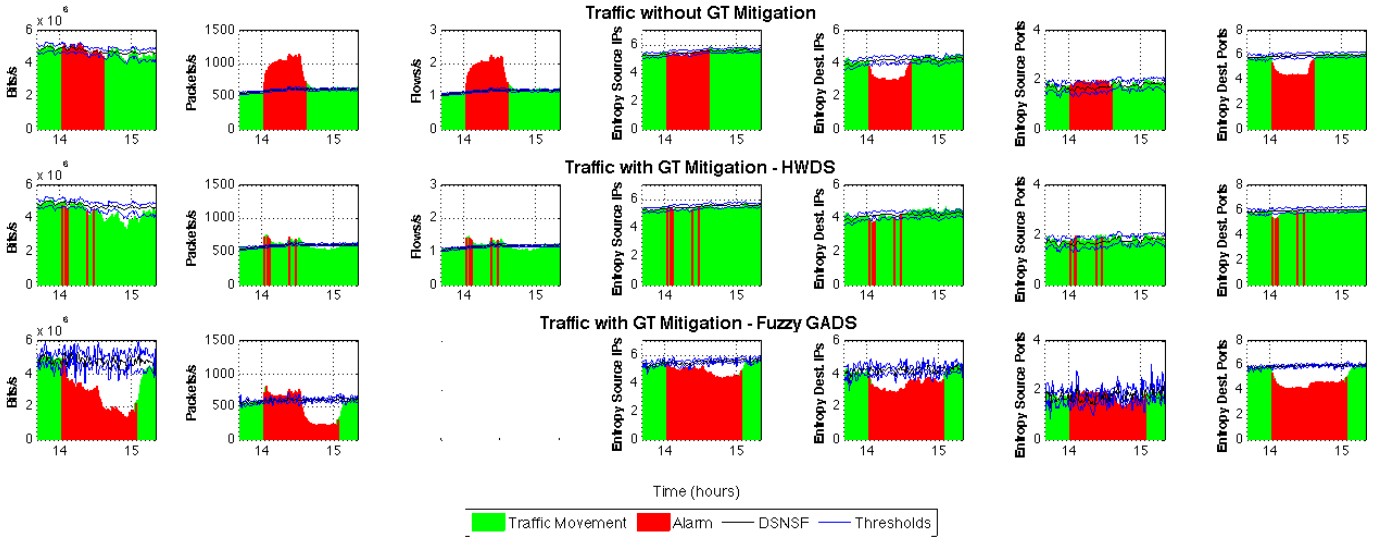


FIGURE 7. Comparison between Traffic Movement and Alarms with and without the presented GT-based decision-making approach for HWDS and Fuzzy-GADS after the second played game in Experiment 3 (DDoS 2560 attacking hosts).

Cost functions have the same weight. One can change these values to favor a certain function according to the need of aparticular network. Besides, by setting w^{atk} and w^{def} the same value, the problem turns into a zero-sum game. This specific kind of game can be solved through a minimax theorem, achieving a Nash Equilibrium (steady state or optimal solution) when the Defense system minimizes the maximum payoff of the Attacker [31].

Thus, a matrix of 10000x1000 elements is generated relating the Attacker optimization variables m and u with the Defense System's optimization variable D, and the Payoffs of the Attacker (3) and Defense system (4) are calculated as described in Section III. Fig. 6 illustrates an example of mesh plot for the variables D, m, u and the Attacker's Payoff P_{atk} used to calculate the Nash Equilibrium of the problem.

At this stage, the Mitigation Module found the solution of the problem with $D_{\%} = 40.6\%$ packets (30 flows per minute), $m = 8$ and $u_{\%} = 100\%$ for the HWDS, and with $D_{\%} = 20.6\%$ packets (18 flows per minute), $m = 14$ and $u_{\%} = 100\%$ for the Fuzzy-GADS. The difference between the methods' results occurs due to the difference of the provided information to the Mitigation Module. Furthermore, the variable u selected for both methods was defined as 100% because it is an UDP DDoS attack and, thus, the intensity of the attack has small influence over the available bandwidth.

As a one-shot game, this is the final result, and the dropping process can now be performed. The dropping process is performed by the Firewall controlled by the Mitigation Module. For the HWDS, this module selects, through data collection conducted by the Information Module [9], the flows directed to the attacked server and, excluding legitimate flows from known hosts, randomly drops flows until the limit of 30 flows per minute is achieved. For the Fuzzy-GADS, as the Mitigation Module does not have any additional information about the attack, a random drop process is performed until the limit of 18 flows per minute is achieved.

This process is performed every minute for 1 hour. This period was chosen in order to test the influence of the dropping process over legitimate hostssince the injected attacks last only 30 minutes.

At this point, the Defense system continues to monitor through the Detection Module the behavior of the network through the generated DSNSFs. If a new anomaly is detected, a new game is played between Attacker and Defense system, and a new Dropping rate is chosen. In our analysis of experiment 3, new alarms were triggered by the HWDS at 14:04 and by Fuzzy-GADS at 14:03. Thus, a new game was played, and the Mitigation Module found the Nash Equilibrium of the problem when $D_{\%} = 30.6\%$ packets (20 flows per minute), $m = 7$ and $u_{\%} = 100\%$ for the HWDS, and when $D_{\%} = 27.9\%$ packets (27 flows per minute), $m = 13$ and $u_{\%} = 100\%$ for the Fuzzy-GADS.

After that, no more alarms were triggered, and the comparison results are shown in Fig. 7. As observed, the random dropping process performed by the GT-Fuzzy-GADS significantly impacts legitimate users, since a considerable volume losscan be observed in the "Bits/s" plot. The information provided by HWDS greatly improves de mitigation process, directly impacting into the GT-HWDS outcomes.

Fig. 8 and 9 depict the complete achieved results for GT-HWDS and GT-Fuzzy-GADS, respectively. As observed, GT-HWDS fared better due to the information provided to the Mitigation Module. However, both approaches successfully mitigated the attacks performed from experiments 1 to 5, guaranteeing the operation of the SDN central controller. The GT-Fuzzy-GADS, however, have a disadvantage of impacting a higher number of legitimate users. Furthermore, it is important to highlight that HWDS can identify the source

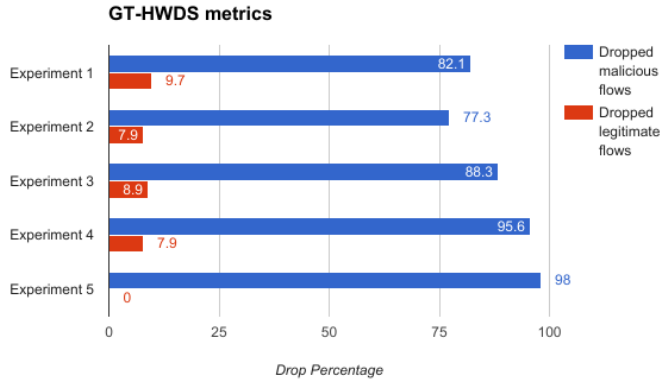


FIGURE 8. Results achieved for GT-HWDS system.

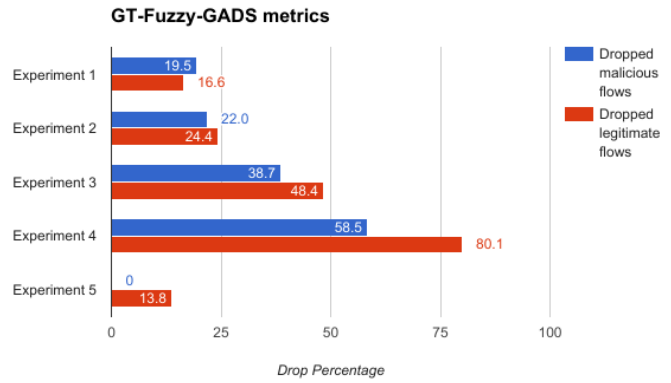


FIGURE 9. Results achieved for GT-Fuzzy-GADS system.

of a DoS attack. Thus, in experiment 5 for HWDS, there was no need to play the game since a directed dropping process solves the problem, which reflects on the numerical outcomes. Finally, even though GT-HWDS fared better in most performance tests, GT-Fuzzy-GADS triggered alarms faster on Experiments 1 and 2, demonstrating its efficiency on detecting the occurrence of anomalies even when they are starting, a criticalperiod where abnormalities tend to be stealthier.

VI. CONCLUSION

In this paper, we presented GT-HWDS, an autonomous supervising system able to detect, identify and mitigate the impact caused by DoS and DDoS attacks on SDNs. The system deploys the HWDS method to detect and identify anomalies based on a seven-dimensional traffic characterization process. After this step, the system triggers a Mitigation Module based on a GT decision-making approach to choose the best defense strategy, which is autonomously applied to the network as an immediate countermeasure. To evaluate the performance of the proposed system, we used five different test scenarios, characterized by one DoS and four DDoS attacks with different intensities. Furthermore, we used the Fuzzy-GADS method instead of HWDS to verify the applicability of the presented game theoretical approach as a standalone Mitigation Module. The obtained results corroborated the effectiveness of both methods, which achieved similar outcomes from anomaly detection metrics, butGT-HWDS fared better for most of the performance tests. Such performanceresults are due to the availability of the Information Module as part of the HWDS method, which provides relevant information about the attack and the SDN to the Mitigation Module. Thus, we conclude that GT-HWDS system is an efficient and powerful defense mechanism for SDNs, avoiding congestion over its central controller by precise mitigation actions. Furthermore, it is possible to conclude that the presented GT-based decision-making approach can be used as a standalone Mitigation Module able to guarantee the proper operation of a SDN, even though it may directly impact on the user experience of legitimate hosts depending on the attack intensity.

For future works, we intend to analyze an adaptable period for the mitigation process to be applied at the SDN while evaluate the game theoretical approach as a standalone mitigation module coupled with different anomaly detection and identification methods. Furthermore, we intend to model different games to enable the mitigation module to counteract different kinds of network anomalies, such as Port Scans and Worms.

APPENDIX
PROOF OF NASH EQUILIBRIUM EXISTENCE

As stated by Nash [38], there always exist Equilibrium points in N-person games. Specifically, on the problem reported in this paper, which is defined as a zero-sumtwo-person game (also known as matrix game), one or more equilibrium points may exist. However, all equilibrium points yield the same payoff for the players, *i.e.*, a Nash Equilibria of the problem is always the optimal outcome. Formally:

Theorem 1: Let G be a two-player zero-sum game defined by $G = (A_{att}, A_{def}, P)$. *Let* (η_{att}, η_{def}) *and* $(\kappa_{att}, \kappa_{def})$ *be two Nash Equilibria of G, then:*

1. *P is the general Payoff since the attacker's payoff is the opposite of the defender's.*
2. $P(\eta_{att}, \eta_{def}) = P(\kappa_{att}, \kappa_{def})$

Proof: The first part of the Theorem 1 is achieved by the definition of zero-sum games, where:

$$P_{att} + P_{def} = 0 \tag{19}$$

As (η_{att}, η_{def}) is one of Nash Equilibria, the attacker player, who tries to maximize P, cannot change its strategy without reducing P, *i.e.*:

$$P(\eta_{att}, \eta_{def}) \geq P(\kappa_{atk}, \eta_{def}) \tag{20}$$

However, $(\kappa_{att}, \kappa_{def})$ is another Nash Equilibrium of the problem, and the defense system player, who aims to minimize P, cannot change its strategy without increasing P:

$$P(\kappa_{atk}, \eta_{def}) \geq P(\kappa_{atk}, \kappa_{def}) \tag{21}$$

Combining these two inequalities we achieve:

$$P\left(\eta_{atk}, \eta_{def}\right) \geq P\left(\kappa_{atk}, \eta_{def}\right) \geq P\left(\kappa_{atk}, \kappa_{def}\right) \tag{22}$$

which proves the part 2 of the Theorem.

Thus, although we only found one of the Nash Equilibria on all performed games through the tests of our proposed decision-making system, there may be two or more equilibrium points [38]. However, as the game described in this paper is a zero-sum two-player game, all Nash Equilibria points have the same Payoff value, i.e., it is irrelevant which one is chosen as the game answer.

REFERENCES

[1] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, "Botnet in DDoS attacks: Trends and challenges," *IEEE Commun. Surveys. Tuts.*, vol. 17, no. 4, pp. 2242–2270, 4th Quart., 2015.

[2] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[3] W. Li, W. Meng, and L. F. Kwok, "A survey on OpenFlow-based software defined networks: Security challenges and countermeasures," *J. Netw. Comput. Appl.*, vol. 68, pp. 126–139, Jun. 2016.

[4] S.-S. Seo, Y. J. Won, and J. W.-K. Hong, "Witnessing distributed denial-of-service traffic from an attacker's network," in *Proc. 7th Int. Conf. Netw. Ser. Manage. (CNSM)*, Oct. 2011, pp. 1–7.

[5] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Trans. Netw. Ser. Manage.*, vol. 6, no. 2, pp. 110–121, Jun. 2009.

[6] M. H. A. C. Adaniya, M. F. Lima, J. J. P. C. Rodrigues, T. Abrão, M. L. Proença, Jr., "Anomaly detection using DSNS and firefly harmonic clustering algorithm," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 1183–1187.

[7] Q. Guan and S. Fu, "Wavelet-based multi-scale anomaly identification in cloud computing systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 1379–1384.

[8] M. V. O. de Assis, J. J. P. C. Rodrigues, and M. L. Proença, Jr., "A novel anomaly detection system based on seven-dimensional flow analysis," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 735–740.

[9] M. V. O. de Assis, J. J. P. C. Rodrigues, and M. L. Proença, Jr., "A seven-dimensional flow analysis to help autonomous network management," *Inf. Sci.*, vol. 278, pp. 900–913, Sep. 2014.

[10] Y. Cui *et al.*, "SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks," *J. Netw. Comput. Appl.*, vol. 68, pp. 65–79, Jun. 2016.

[11] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments," *Comput. Netw.*, vol. 62, pp. 122–136, Apr. 2014.

[12] M. Lee, X. Ye, D. Marconett, S. Johnson, R. Vemuri, and S. J. B. Yoo, "Autonomous network management using cooperative learning for network-wide load balancing in heterogeneous networks," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Nov. 2008, pp. 1–5.

[13] J. Gao, S. A. Vorobyov, and H. Jiang, "Game theoretic solutions for precoding strategies over the interference channel," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Nov. 2008, pp. 1–5.

[14] O. E. Ferkouss and W. Ajib, "Game theory based resource allocation for cognitive radio networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 1174–1179.

[15] M. Hamdi and H. Abie, "Game-based adaptive security in the Internet of Things for eHealth," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 920–925.

[16] M. V. O. de Assis and M. L. Proença, Jr., "Scorpius: sFlow network anomaly simulator," *J. Comput. Sci.*, vol. 11, no. 4, pp. 662–674, Apr. 2015.

[17] A. H. Hamamoto, L. F. Carvalho, and M. L. Proença, Jr., "ACO and GA metaheuristics for anomaly detection," in *Proc. 34th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Nov. 2015, pp. 1–6.

[18] C. Basile, D. Canavese, A. Lioy, C. Pitscheider, and F. Valenza, "Inter-function anomaly analysis for correct SDN/NFV deployment," *Int. J. Netw. Manage.*, vol. 26, no. 1, pp. 25–43, Jan. 2016.

[19] J. Li, J.-H. Yoo, and J. W.-K. Hong, "Dynamic control plane management for software-defined networks," *Int. J. Netw. Manage.*, vol. 26, no. 2, pp. 111–130, Mar. 2016.

[20] S. Song, H. Park, B.-Y. Choi, T. Choi, and H. Zhu, "Control path management framework for enhancing software-defined network (SDN) reliability," *IEEE Trans. Netw. Ser. Manage.*, to be published, doi: 10.1109/TNSM.2017.2669082.

[21] G. Poulios, K. Tsagkaris, P. Demestichas, A. Tall, Z. Altman, and C. Destré, "Autonomics and SDN for self-organizing networks," in *Proc. 11th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2014, pp. 830–835.

[22] Z. Tan, A. Jamdagni, X. He, P. Nanda, and R. P. Liu, "A system for denial-of-service attack detection based on multivariate correlation analysis," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 2, pp. 447–456, Feb. 2014.

[23] M. Long, C.-H. Wu, and J. Y. Hung, "Denial of service attacks on network-based control systems: Impact and mitigation," *IEEE Trans Ind. Informat.*, vol. 1, no. 2, pp. 85–96, May 2005.

[24] L. F. Carvalho, S. Barbon, Jr., L. S. de Mendes, and M. L. Proença, Jr., "Unsupervised learning clustering and self-organized agents applied to help network management," *Expert Syst. Appl.*, vol. 54, pp. 29–47, Jul. 2016.

[25] C. Röpke and T. Holz, "On network operating system security," *Networks*, vol. 26, no. 1, pp. 6–24, Jan. 2016.

[26] A. Lara and B. Ramamurthy, "OpenSec: Policy-based security using software-defined networking," *IEEE Trans. Netw. Ser. Manage.*, vol. 13, no. 1, pp. 30–42, Mar. 2016.

[27] L. M. Bruce, "Game theory applied to big data analytics in geosciences and remote sensing," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2013, pp. 4094–4097.

[28] Q. Wu, S. Shiva, S. Roy, C. Ellis, and V. Datla, "On modeling and simulation of game theory-based defense mechanisms against DoS and DDoS attacks," in *Proc. Spring Simul. Multiconf.*, San Diego, CA, USA, 2010, pp. 159:1–159:8.

[29] H. S. Bedi, S. Roy, and S. Shiva, "Game theory-based defense mechanisms against DDoS attacks on TCP/TCP-friendly flows," in *Proc. IEEE Symp. Comput. Intell. Cyber Secur. (CICS)*, Apr. 2011, pp. 129–136.

[30] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu, "A survey of game theory as applied to network security," in *Proc. 43rd Hawaii Int. Conf. Syst. Sci. (HICSS)*, Jan. 2010, pp. 1–10.

[31] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*, 1st ed. Cambridge, MA, USA: MIT Press, 1994.

[32] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[33] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms*, 2nd ed. Hoboken, NJ, USA: Wiley, 2004.

[34] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *Appl. Soft Comput.*, vol. 10, no. 1, pp. 1–35, Jan. 2010.

[35] L. A. Zadeh, "Is there a need for fuzzy logic?" in *Proc. Annu. Meeting North Amer. Fuzzy Inf. Process. Soc. (NAFIPS)*, May 2008, pp. 1–3.

[36] R. Matias, A. M. M. Carvalho, L. B. Araujo, and P. R. M. Maciel, "Comparison analysis of statistical control charts for quality monitoring of network traffic forecasts," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2011, pp. 404–409.

[37] D. C. Montgomery, *Introduction to Statistical Quality Control*, 6th ed. Hoboken, NJ, USA: Wiley, 2008.

[38] J. F. Nash, Jr., "Equilibrium points in n-person games," *Proc. Nat. Acad. Sci. USA*, vol. 36, no. 1, pp. 48–49, 1950.

MARCOS V. O. DE ASSIS received the master's degree in computer science from the State University of Londrina, Brazil, where he is currently pursuing the Ph.D. degree with the Electrical Engineering Department. He is currently a Professor with the Engineering and Exact Department, Federal University of Paraná, Brazil. He is part of the research group Computer Networks and Data Communication. His research interest is in the management and security of large-scale computer networks.

ANDERSON H. HAMAMOTO received the M.Sc. degree in computer science from the State University of Londrina in 2017, where is currently pursuing the Ph.D. degree in electrical engineering. He has experience in computer science with an emphasis in computer networks. He is part of the research group Computer Networks and Data Communication. His main research interests are the management and security of computer networks.

TAUFIK ABRÃO (M'97–SM'12) received the B.S., M.Sc., and Ph.D. degrees in electrical engineering from the Polytechnic School, University of São Paulo, São Paulo, Brazil, in 1992, 1996, and 2001, respectively. Since 1997, he has been with the Communications Group, Department of Electrical Engineering, State University of Londrina, Paraná, Brazil, where he is currently an Associate Professor of Telecommunications and the Head of the Telecomm and Signal Processing Laboratory. From 2007 to 2008, he was a Post-Doctoral Researcher with the Department of Signal Theory and Communications, Polytechnic University of Catalonia, Barcelona, Spain. In 2012, he was an Academic Visitor with the Southampton Wireless Research Group, University of Southampton, Southampton, U.K. He has participated in several projects funded by government agencies and industrial companies. He is involved in editorial board activities of six journals in the telecommunications area and has served as a TPC Member in several symposiums and conferences. He has supervised 21 M.Sc., six Ph.D., and three post-doctoral students, and coauthored ten book chapters on mobile radio communications and over 180 research papers published in specialized/international journals and conferences. His current research interests include communications and signal processing, especially massive MIMO and OFDM/OFDMA systems, detection and estimation methods, cooperative communication and relaying, resource allocation, and heuristic and convex optimization aspects of 4G and 5G wireless communication systems. He is a Senior Member of the Brazilian Telecommunication Society. He has also served as an Editor of the IEEE Communications Surveys and Tutorials since 2013, the IEEE Access since 2016, and the *IET Journal of Engineering* since 2014.

MARIO LEMES PROENÇA JR. received the M.Sc. degree in computer science from the Informatics Institute, Federal University of Rio Grande do Sul, in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas in 2005. He is currently an Associate Professor and the Leader of the research group that studies computer's network in the Computer Science Department with the State University of Londrina (UEL), Brazil. He has supervised 12 M.Sc. and two Ph.D. students. He has been a master's supervisor of computer science with the State University of Londrina and Ph.D. supervisor with the Department of Electrical Engineering, UEL. He has authored or coauthored over 90 papers in refereed international journals and conferences, books chapters, and one software register patent. His current research interests include computer network, network operations, management and security, and IT governance.

APÊNDICE B - *Fast Defense System Against Attacks in Software Defined Networks*

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 28, 2018, accepted October 20, 2018, date of publication October 29, 2018,
date of current version December 7, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2878576

Fast Defense System Against Attacks in Software Defined Networks

MARCOS V. O. DE ASSIS[1], MATHEUS P. NOVAES[2], CINARA B. ZERBINI[2], LUIZ F. CARVALHO[2], TAUFIK ABRÃO[3], AND MARIO L. PROENÇA Jr.[2]

[1]Engineering and Exact Department, Federal University of Paraná, Palotina 85950-000, Brazil
[2]Computer Science Department, State University of Londrina, Londrina 86057-970, Brazil
[3]Electrical Engineering Department, State University of Londrina, Londrina 86057-970, Brazil

Corresponding author: Mario L. Proença Jr. (proenca@uel.br)

This work was supported in part by CAPES, due to the concession of scholarships, in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grants 308348/2016-8 and 304066/2015-0, and in part the Federal University of Paraná (UFPR) under Project Banpesq/2014016797.

ABSTRACT With the ever-growing data traffic in computer networks nowadays, the management of large-scale networks is a challenge for guaranteeing the quality of the provided services. This is due to the increasingly usage of connected applications, such as Internet of Things and cloud computing environments. Software-defined networking (SDN) is a new paradigm that aims to make this management process easier by centralizing the configuration of all network devices into a single programmable central controller. However, as any centralized service, this architecture is susceptible to security vulnerabilities, such as distributed denial of service (DDoS) and port scan attacks. Thus, security methods are necessary to guarantee the normal operation of SDN's central controller. Furthermore, networks are transporting an increasingly amount of information day by day, which could mean data loss in case of long network unavailability. For this reason, security mechanisms must operate online, with fast-responding countermeasures to mitigate the impact of the detected attacks over the SDN. In this paper, we present a fast SDN defense system against DDoS and port scan attacks, which runs directly into the central controller and uses a game theoretical approach for attack mitigation. For the detection, we compare three different approaches, particle swarm optimization, multi-layer perceptron neural network, and discrete wavelet transform. We test our approach over IP flow data generated over Mininet network emulator, along with floodlight controller, and the presented defense system achieved good outcomes for both detection and mitigation processes.

INDEX TERMS DDoS, DWT, MLP, port scan, PSO, SDN.

I. INTRODUCTION

The amount of network applications and connected devices using the Internet as data transmission environment is rapidly increasing. Web applications, such as online banking, social networks and e-commerces, as well as mobile usage and the emergence of the Internet of Things (IoT) paradigm, are increasing in popularity every day. However, the network performance and the demands required by the referred applications are becoming a complex task for the network administrators to handle due to the heterogenous and static infrastructure of the traditional networks.

Software Defined Networking (SDN) is an emerging network architecture aiming to supply the demands of existing and future connected applications. This new paradigm has as main characteristic the division between the network's planes. In other words, the control and the data planes are decoupled from the network devices through an abstraction plane [1]. This division allows to control, modify and manage the network behavior through a dynamic software interface, unlike the traditional networks where network devices are proprietary locked boxes, which limits its flexibility relating to its internal control [2].

New monitoring and management resources that are able to improve the performance and reduce networks bottlenecks are present in SDN. Despite the discussed characteristics, such as control centralization and network programming, these networks are also subject to threats and security vulnerabilities. Due to the centralized nature of the network intelligence through an SDN controller, as any centralized service, this controller can be targeted by Denial of Service (DoS)

attacks [3], [4]. The DoS attack attempts to exhaust the network resources and it is more powerful when performed in a distributed way (Distributed DoS, or DDoS). When attacking servers, the attacker aims to make a service unavailable by sending several requests, while in infrastructure attacks, the attacker overwhelms a network link [5]. Furthermore, DDoS attacks are frequently followed by port scan attacks, where an attacker scans the server's ports in order to find an opening for an intrusion process.

However, the management of the network's information security is a task of high complexity, since it is necessary to guarantee the availability, reliability and integrity of the network services provided to end users. Thus, it is necessary the usage of efficient techniques to help on the autonomous management and security processes, such as anomaly detection and mitigation, on SDN environments. Anomaly detection systems can be classified in two main groups: signature-based and based on the networks normal operation. The first one uses a database which contains the patterns of known anomalies. The second one generates a network's traffic profile, which represents its behavior in normal conditions and does not require knowledge of the anomalies to detect them [6], [7]. The main disadvantage of the profile based approach is the occurrence of false-positive alerts, when the traffic of legitimate users are detected and classified as abnormal [8], [9].

In this paper, we present a system for fast detection and mitigation of DDoS and port scan attacks on SDN environments. The presented system analyzes IP flow data in five-second intervals, providing a faster detection mechanism than traditional anomaly detection approaches, such as [10] and [11], which operate with five-minute time intervals. For this, the presented system is divided into three main modules: Detection, Identification and Mitigation modules.

On the Detection module, we compare the usage of three different models using a multidimensional IP flow analysis. This approach is based on the management of six different IP flow features: bits/s, packets/s, source/destination IP addresses and source/destination ports. The first method uses the Particle Swarm Optimization (PSO) on a non-supervised learning approach based on the data clustering for traffic characterization and an approach based on the Chebyshev Inequality for anomaly detection. The second one is an artificial neural network which uses Multi-Layer Perceptron (MLP), a supervised machine-learning process for anomaly detection. Finally, the third one uses the Discrete Wavelet Transform (DWT), a technique based on signal processing which decomposes the input traffic data into its constituent parts based on its frequency in order to characterize the traffic and detect the presence of anomalies.

On the Mitigation module, we use the game theoretical (GT) approach presented in [12] for DDoS mitigation, which proved itself to be efficient on preventing DDoS attacks over SDN border gateways. In this paper, we test the GT approach efficiency on mitigation DDoS attacks directly into the SDN central controller, which also prevents internal attacks. Furthermore, we extend the operation of the model to provide defense against port scan attacks.

To test the efficiency of the presented system, as well as the performance outcomes of the different methods tested for anomaly detection, we use simulated SDN data generated by Mininet network emulator, together with Floodlight SDN controller and OpenFlow IP flow data.

The main contributions of this paper are:

- A system for SDN defense against DDoS and port scan attacks;
- The performance comparison of three different fast anomaly detection methods on an SDN environment;
- Efficiency analysis on the usage of the mitigation approach presented in [12] directly into the SDN central controller instead of on the border gateway;
- Usage of reliable and replicable data through Mininet network emulator, since it is one of the most used mechanisms in SDN researches nowadays.
- Comparison between the presented anomaly detection methods and classic literature methods.

The remainder of this paper is organized as follows: Section II shows the related works; Section III describes the presented defense system for SDN environments; Section IV describe the anomaly detection methods used on the Detection Module of the presented SDN defense system; Section V discuss the performance results achieved; finally, Section VI presents the conclusions and future works.

II. RELATED WORKS

Software-Defined Networking (SDNs) is a new network paradigm that improves network control. SDN helps solve several problems faced nowadays with our traditional large-scale networks, such as resource allocation and online configuration. Thus, several researches are being performed within this area. Cox *et al.* [13] presented a survey on the state of the art of SDN. They highlighted the efficiency of this architecture, also pointing out implementation cases outside the academia, on companies like Google, AT&T and Microsoft. Furthermore, they describe the advantages and the challenges faced by this technology. Paliwal *et al.* [14] addressed SDN paradigm by presenting an extensive review report on various available central controllers. For each analyzed controller, the authors discussed their design aspects and architecture overview, besides evaluating their efficiency over performance metrics. Zhang *et al.* [15] introduced the concept of SD-ICN networking, which is the junction of SDN paradigm with Information Centric Networking (ICN). ICN is also an emerging network paradigm which uses features like in-network caching and name-based routing to support the ever increasing growth of Internet traffic. According to the authors, the junction of these two promising paradigms is able to improve management and security processes.

However, the centralized architecture in which SDNs operate brings possible security threats, such as Denial of Service (DoS) attacks. As security is a major concern for most network environments, several papers address this issue.

Xu *et al.* [16] proposed a Smart Security Mechanism (SSM) to defend SDN-based Internet of Things (IoT) environment against the new-flow attack. The authors performed simulations and testbed, and the achieved results pointed out the feasibility of the proposed system. Zhang and Sun [17] presented an SDN-based integrated IP source address validation architecture (ISAVA) which can cover both intra and inter-domain areas and effectively lower SDN devices deployment cost. This approach helps protect SDN networks against IP spoofing attacks. Conducted experiments proves that the proposed method was successful in solving the stated problem. Yu *et al.* [18] addressed the security of SDN vehicular networks against DDoS attacks. The authors highlight the vulnerability of the SDN environment against this attack and propose a detection mechanism based on OpenFlow messages, flow feature extraction and Support Vector Machine (SVM) classification. Through a simulation environment, the performed tests achieved effective results. Peng *et al.* [19] presented an anomaly detection method for SDN environments based on double P-value of transductive confidence machines for K-nearest neighbors (K-NN) algorithm. The proposed method is able to detect anomalies and to perform a classification of the detection over IP flows, and the test's outcomes points out a better performance than similar detection approaches. Carvalho *et al.* [20] presented an SDN-based ecosystem able to monitor the traffic of the network and proactively detect anomalies. After an anomalous behavior is detected, a deeper analysis is performed through the usage of multiple OpenFlow features, which are used to optimize mitigation policies to reduce the impact of the attack over the SDN operation.

As network's traffic are increasing day-by-day, the amount of information traveling on them is massive and any problem that causes the network services to become unavailable signify a huge amount of lost data. For instance, on a 10Gb link, up to 3Tb of data may be lost on a 5 minute interval (a common analysis time interval on traditional management and security systems) on a network stoppage. Thus, fast-response or online management systems are required to guarantee the quality of the network provided services. Zhao *et al.* [21] presented a novel framework for real time network traffic anomaly detection using machine learning algorithms. They collected and analyzed in real-time data from the University of Missouri-Kansas City, using big data processing frameworks along with machine learning tools to detect anomalies within the analyzed data. Wang *et al.* [22] proposed a network anomaly traffic detection method based on the IP flow template, capturing and analyzing network traffic in real-time. The authors performed tests in a controlled network environment, and highlight that the proposed approach accurately detect the anomaly network traffic.

One of the most important steps on mitigating the effect of network attacks is the anomaly detection. There is a vast amount of researches in this area due to its high importance and difficulty on providing efficient and fast-responsive solutions. In this paper, we test three different approaches on the anomaly detection step of our proposed SDN defense system: Particle Swarm Optimization (PSO), Multi-Layer Perceptron (MLP) and Discrete Wavelet Transform (DWT). These approaches are widely used in several computer networks study areas.

Several approaches apply clustering algorithms on the development of anomaly detection systems. K-means algorithm is a widely used approach on classification and detection of data anomalies in different areas. However, one of its limitations is the local convergence and sensitivity relating to the centroids of each cluster. Karami and Guerrero-Zapata [23] presented a new anomaly detection system that operates in two phases. In the first one, they applied a hybrid approach of PSO and K-means with two cost functions, one to find the distance between the clusters and another to set the local optimization, which determines the ideal number of clusters. On the second phase, they applied fuzzy logic for the classification on the anomaly detection. Experimental results demonstrated that the proposed algorithm is able to achieve the ideal amount of well-separated clusters, as well as to elevate the detection rate and lower false-positive rates. Lima *et al.* [24] applied PSO together with K-means for a baseline generation for backbone management applied to network's SNMP data. The objective of the PSO usage was to improve the clustering solutions and the cluster's centroids calculation. Numerical results show that detection and false-positive alarms was promising.

Some anomaly detection approaches apply a combination of machine learning techniques that are called ensembles methods. These approaches aims to achieve better predictive performance outcomes. Aburomman and Reaz [25] proposed an intrusion detection system (IDS) applying three different machine learning techniques: Support Vector Machine (SVM), PSO and K-NN. On the training phase, six K-NN classifiers and six SVM classifiers were used on the same data set. Then, the PSO method was applied by combining the output of the twelve classifiers on the generation of a final classifier. To validate the proposed system the authors uses five random subgroups of the KDD99 dataset. The evaluation metric used was the accuracy, which achieves results of 92% on the average.

A Multi-layer Perceptron (MLP) network alongside a Genetic Algorithm (GA) optimization method was proposed by Singh and De [26] for detecting DDoS attacks on the application layer. The algorithm was developed based on the analysis of the fields of received packets, such as HyperText Transfer Protocol (HTTP), the number of IP addresses during a time interval, port number mapping and size of the incoming packets. These four features were used as input for the construction of a classifier. Using MLP and GA, the dataset is classified as attacks or normal users. Experimental results show that MLP-GA provides a 98.04% efficiency rate on detecting DDoS attacks. Siaterlis and Maglaris [27] proposed a MLP classification network for DDoS detection using IP flow data. According to the authors, the number of neurons composing the MLP hidden layer influences the

classification results up to $2N + 1$ neurons, where N is the number of neurons on the input layer. Adding more neurons to the hidden layer implies no further improvement on the classification results. The authors highlight the efficiency of the proposed method on detecting DDoS behaviors. Jadidi *et al.* [28] used a MLP network on detecting anomalies alongside with the Gravitational Search Algorithm (GSA) to optimize the neural weights of the MLP, highlighting the efficiency of the method on the stated problem. Furthermore, Nikravesh *et al.* [29] investigated the accuracy of the characterization process of mobile network traffic using the methods MLP, MLP with Weight Decay (MLPWD) and Support Vector Machine (SVM).

Tian and Ding [30] developed a method for network anomalies detection using two tools, a Traffic Matrix (TM) and wavelets. A TM corresponds to five minutes of network traffic, which may contain normal or anomalous traffic. A wavelet transform was performed in this matrix, producing coefficients that provide historical traffic information. Through this historical data, some parameters are collected. A comparison between these coefficients and abnormal coefficients was performed, aiming to identify DDoS attacks. This technique showed high detection rates, close to one hundred percent, and a false alarm rate close to six percent. The disadvantage in using this approach lies in the fact that the matrices have samples of five minute intervals, which in a current network with links of 10Gb, 100Gb, or even 400Gb, means the exchange of large amounts of information and data, causing the detection occurrence late and therefore ineffective. Still using wavelets, it is important to highlight the work of Kanarachos *et al.* [31] and Gao *et al.* [32], both developed in the traditional network environment. Kanarachos *et al.* [31] proposed a system that uses a combination of three techniques: wavelets, neural networks and Hilbert transform. The model is divided into three stages. In the first stage a wavelet transform of Daubechies with eight levels of decomposition was performed, and then a noise removal technique was applied. In the second stage, a subset of the noise-free data was chosen for neural network training, generating a traffic forecast. The third and last stage used the Hilbert transform in the signal error, which is the signal resulting from the difference between the filtered signal and the training output of the neural network. Gao *et al.* [32] presented an anomaly detection model using *wavelet packet*, which is a generalization of the pyramidal algorithm of the traditional wavelet transform. The authors have defined that the proposed method is capable of detecting "long-term" anomalies and medium frequencies. The system also guarantees an improvement in the reliability of the detection using an adaptive reconstruction of the detail coefficients from the wavelet transform, including anomaly. The presented methods scored satisfactory detection rates, but as they were developed for traditional networks, mitigation routines became more complex to implement.

In this paper, we present a defense system for SDN controllers able to detect and mitigate both DDoS and port scan attacks. The system operates online, collecting and analyzing IP flow data directly into the SDN controller every 5 seconds to detect these kinds of anomaly. Unlike other traditional anomaly detection systems, our proposal quickly responds to the detected threat by triggering a mitigation process, performed by a game theoretical approach, five seconds after the detection.

III. SDN DEFENSE SYSTEM

In this section, we describe the presented SDN defense system, its organization and operation. It aims to help defending the SDN central controller against DDoS and port scan attacks. To achieve this objective, an hexa-dimensional IP flow analysis is used through the collection of different IP flow features. These dimensions are used to characterize the network's normal behavior and, later, to detect the occurrence of a network anomaly or attack.

The presented SDN defense system is mainly composed of an IP flow exporter and three modules. The flow exporter protocol used on the development of this paper was OpenFlow. Each one of the three modules are composed of two other sub-modules, as described by Fig. 1. They are the Detection, Identification and Mitigation modules.

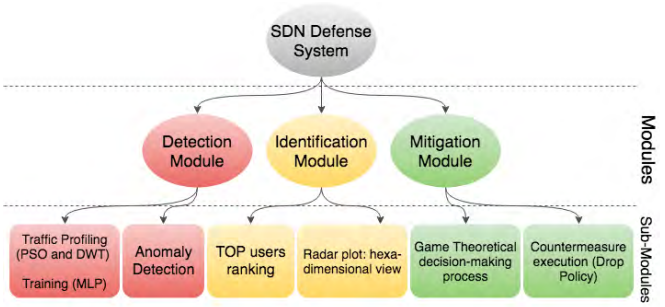


FIGURE 1. General view of the presented SDN Defense System.

The Detection Module performs the detection of the anomaly/attack on the SDN. To perform this task, different methods were applied and compared in this paper to find which one is the most efficient approach. They are the Particle Swarm Optimization (PSO), the Multi-Layer Perceptron (MLP) neural network and the Discrete Wavelet Transform (DWT). Each one of them will be further described on Section IV. As shown by Fig. 1, this module is composed by two sub-modules: "Traffic profiling / training" and "anomaly detection." The first one is divided into two parts due to the fact that the mentioned methods operates differently in this step. The PSO and DWT are non-supervised methods, which generate a normal online profile (traffic profiling every 5 seconds) of the analyzed SDN, *i.e.*, the network profile is generated on each IP flow collection. On the other hand, the MLP method is a supervised learning approach, requiring previous information (training) about the network's normal operation, as well as the anomalies and attacks (here defined as DDoS and port scan). The second sub-module, the anomaly detection, is responsible for detecting the

abnormal behavior of the six analyzed dimensions that occurs when an attack is being performed.

The Identification Module deals with the identification of the attack on the SDN. It is an essential step towards a good mitigation process since different kinds of attacks require specific mitigation policies to achieve a satisfactory outcome. As observed in Fig. 1, it is composed of the sub-modules ''Top users ranking'' and ''Radar plot.'' The first one stores the three most frequent source and destination ports and IP addresses, as well as the three most frequent protocols. The second one provides a general view of the network behavior on a single time interval. Together, both sub-modules help the identification of the attack, as well as the likely attackers and victims. PSO and DWT methods rely on this module for correct mitigation guidance, since they only detect the occurrence of anomalies. MLP method operates as a classifier, enabling the detection step in the identification of the detected anomaly if it is known by the system (present on the training step). Finally, when no attack is detected by the Detection Module, the Identification Module uses the collected data to feed a list of all source IP addresses analyzed on the past 5 minutes, here called ''safe list.'' This list is used to prevent the packet drop of legitimate users.

Finally, the Mitigation Module is responsible for taking the optimal countermeasures against the detected and identified attack. As described in Fig. 1, it is composed of the sub-modules ''Game Theoretical decision-making process'' and ''countermeasure execution.'' For the decision-making process, we used a game theoretical approach, presented in [12], that aims to mitigate DDoS attacks at SDN border gateways in order to protect the central controller. However, in this paper, we applied this approach directly into the SDN controller, which analyzes the traffic data with the three presented detection methods and triggers an alarm in case of a DDoS or a port scan detection. This alarm will activate the mitigation module, which will provide the SDN controller an optimal packet drop rate and a list of legitimate users or ''secure hosts.''

If the detected attack is a port scan, the countermeasure approach is to simply drop all packets of the attacker's source IP address. This identification is possible due to the operation of the Identification Module, as shown by de Assis *et al.* [12], where, using this approach, it was possible to identify significant information about DoS, DDoS, port scans and flash crowd anomalies. However, if the detected attack is a DDoS, then the game theory is invoked.

The game theoretical approach we use is a two-player game. As the attacker (malicious user and first player) tries to maximize the damage caused to the network while reducing its chance of being detected, the defense mechanism (second player) tries to reduce the impact posed by the attacker and preserve the SDN normal operation. Furthermore, it is a zero-sum game, *i.e.*, the gain of one player is the loss of another.

Each player has a set of possible actions that must be performed to increase its gain or payoff. For the attacker, it is possible to:

- Change the number of packets per second directed to the network by each attacking node;
- Modify the number of attacking nodes;

On the other hand, the defense mechanism is able to:

- Allow packets to traffic through the SDN controller;
- Drop packets to prevent them from being further processed by the SDN controller;

In order to measure the impact of each one of these actions, several metrics are used. They are i) the normalized error between the expected SDN behavior and the analyzed time interval, ii) the average bandwidth consumption of legitimate users in comparison to malicious ones, iii) the attack cost for the attacker, iv) and the estimated packet loss of legitimate users through the dropping process. For more implementation details, please refer to [12].

The second sub-module (countermeasure execution) generates as outcome a set of packet dropping policies that is provided to the SDN's central controller for instant implementation. It is important to highlight that the data provided by the Identification Module prevents the SDN controller from dropping some known legitimate users, which greatly improves the results of DDoS attacks, as shown by de Assis *et al.* [12].

An important characteristic of the presented defense system is its speed on detecting and taking the adequate countermeasure to mitigate the attack. It was designed to operate online and, thus, the entire process occurs in an autonomic way, *i.e.*, no human intervention occur besides receiving the alarms and reports about the detected attacks.

To enable this online characteristic, the network controller collects and exports IP flows every 5 seconds, submitting this data to a detection analysis. Thus, the mitigation process may quickly start and prevent further damage by the attacks.

The overall operation of the presented SDN defense system is shown by Fig. 2.

As shown, every 5 seconds the SDN controller exports through OpenFlow six flow dimensions: bits/s, packets/s, source IP address, destination IP address, source port and destination port. The first two dimensions are quantitative values, while the remaining are qualitative ones. To enable their usage on the anomaly detection process, they need to be converted into quantitative data. Thus, we apply the Shannon Entropy [33], which enables the information extraction relating to concentration and dispersion of data in these flow dimensions. For this purpose, given an feature $X = \{x_1, x_2, \ldots, x_n\}$ in which x_i represents the number of occurrences of the sample i at the time interval, the entropy H for X is given by:

$$H(X) = -\sum_{i=1}^{N} \left(\frac{x_i}{S}\right) log_2 \left(\frac{x_i}{S}\right) \quad (1)$$

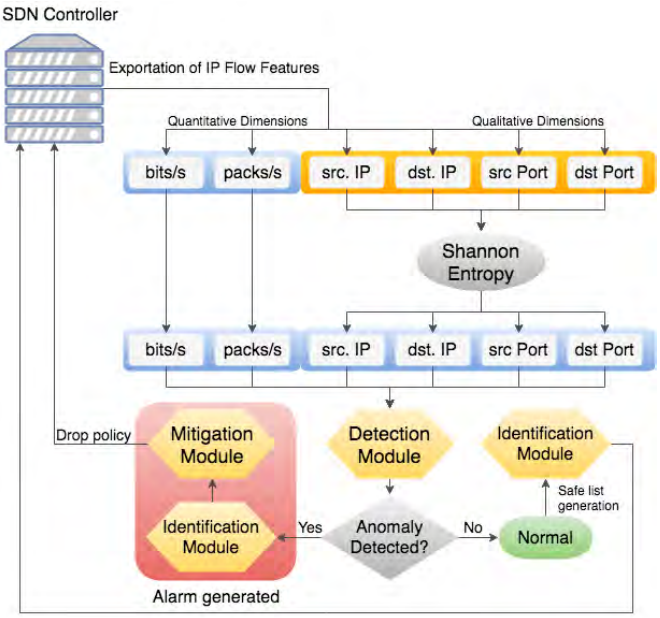


FIGURE 2. SDN defense system operation.

where $S = \sum_{i=1}^{N} x_i$ is the sum of all the values present on the histogram.

Then, the six dimensions are submitted to the Detection module. If an anomaly/attack is detected, then an alarm is generated and the data is submitted to the Identification Module for the attack identification and information collection. After that, these data are submitted to the mitigation module for countermeasure definition, generating a set of packet dropping policies. These policies are, then, sent to the SDN's central controller for mitigation implementation.

On the other hand, if no anomaly/attack is detected on the Detection Module, the network is considered to be operating normally. Then, the analyzed data is submitted to the Identification Module for the generation of the previously described "safe list," which is sent to the SDN's central controller to avoid future packet dropping of legitimate users.

IV. ANOMALY DETECTION METHODS

In this section, we detail the three anomaly detection methods used on the Detection Module of the presented system. The performance analysis of them is shown on Section V.

A. PARTICLE SWARM OPTIMIZATION FOR DIGITAL SIGNATURE

Several pieces of research were developed through the application of Swarm Intelligence (SI) to propose methods aiming to solve complex optimization problems, which are of difficult solution through classic optimization algorithms. Swarm Intelligence are nature-inspired metaheuristics based on the collective behavior of natural agents relating to their iteration mechanisms and environmental organization. The PSO metaheuristic was first introduced by Eberhart and Kennedy [34]. It was inspired by the social behavior of a flock of birds or shoal of fishes, introducing a new approach on functions optimization. On PSO method a flock of birds is randomly initialized into a search space in which each bird is referred as a particle and the set of particles is called "swarm."

PSO aims to optimize a specific function, known as fitness. At each iteration this function is used to measure the efficiency of the generated solutions, *i.e.*, the fitness is used to guide the particles movement towards the problem's solution. Thus, the fitness function measure how close the particles are from the solution (solution's performance), where each particle has an update speed that guides its movement along the search space. Consider a particle population P, where v_p and p_p represents the speed and the position of the particle p. The movement of each particle is performed by updating its movement speed and position through the Eq. (2) and (3), respectively.

$$v_{p+1} = wv_p + c_1 r_1 \left(p_{best_p} - x_p\right) + c_2 r_2 \left(g_{best} - x_p\right) \quad (2)$$

$$x_{p+1} = x_p + v_p \quad (3)$$

where w is the inertia coefficient, c_1 and c_2 are the acceleration constants, r_1 and r_2 are random numbers defined by the interval $[0, 1]$, p_{best_p} is the best position occupied by the particle p until the given iteration and g_{best} represents the global solution at the given iteration by the swarm. According to [34] c_1 and c_2 can receive the value 2.05 and w equals to 0.5.

The particles p_{best_p} and g_{best} are evaluated each iteration through the fitness function. They are updated only in case the current solution presents a better outcome than the values already found until that iteration [35]. The update of the particles p_{best_p} and g_{best} are performed using the Eq. (4) and Eq. (5), respectively.

$$p_{best_p} = p'_{best_p} \quad if\ f(p'_{best_p}) < f(p_{best_p}) \quad (4)$$

$$g_{best} = g'_{best} \quad if\ f(g'_{best}) < f(g_{best}) \quad (5)$$

In several cases the system's convergence can be quickly achieved. The fast convergence makes this method an efficient optimization mechanism. As the approach used in this paper is based on an online multidimensional traffic characterization, this fast convergence is an essential factor. PSO was developed to be a simple method, implemented with few lines of code. It only requires primitive mathematical operations, also representing a low computational cost algorithm able to find optimal regions in multidimensional search spaces.

1) CHARACTERIZATION AND DETECTION MODULE

The anomaly detection method using PSO presented in this paper is divided into three steps:

1) The first step is the online traffic characterization, using the PSO for flow data optimization;
2) The second step is the anomaly detection using Chebyshev's inequality;

3) Finally, the mitigation module is activated when the an anomaly is detected on the previous step.

The traffic characterization is generated using IP flow data collected from the SDN controller, using both quantitative (bits/s and packets/s) and qualitative dimensions (source and destination IP addresses and ports).

The SDN traffic characterization was performed through the organization of the data into clusters. To optimize the clustering, the PSO method was used to find the centroid that best represents this flow set. The traffic characterization is performed each 5 seconds and uses a time window of n past minute to obtain the signature of the next second. This signature is here called as Digital Signature of Network Segment using flow analysis (DSNSF), and represents the networks normal operation behavior. The time window used in this paper is $n = 5$ (minutes). Each DSNSF point is achieved through the mean of the centroids of the C clusters obtained after the PSO optimization process. Algorithm 1 shows the process of DSNSF generation using PSO.

Algorithm 1 - PSO Used to Generate DSNSF

Require: Set of network information extracted from network flows

Ensure: Arrays representing the DSNSF with 17280 samples

```
for i = 1 : 17280 do
    Calculate inferior limit
    Calculate superior limit
    Generate population for time interval
    while a termination criterion is not met do
        Update pBest (4)
        Update gBest (5)
        Update the particle's velocity (2)
        Update the particle's position (3)
        Evaluate population fitness
    DSNSF_i ← average among the centroids
return DSNSF
```

The fitness function applied on the optimization process was the Euclidean distance between IP flow data and the centroids, represented by the equation:

$$J = \sum_{i=1}^{E}\sum_{j=1}^{C}\sqrt{\sum_{a=1}^{A}(c_{ja} - x_{ia})^2} \tag{6}$$

in which E is the amount of flows to be clustered, C represents the number of clusters (for this method, we used the value $C = 2$) and A represents the amount of flow attributes or dimensions. As previously discussed, in this paper we use a six-dimensional analysis. The variable c_{ja} indicates the value of the cluster j belonging to the $a - th$ dimension and x_{ia} is de value of the feature a relating to the element i. The anomaly detection approach of this method is based on the Bienaymé-Chebyshev's inequality. This inequality is used to find behaviors that differs from the generated signature for the quantitative and qualitative dimensions. The Bienaymé-Chebyshev inequality determines a limiar of the data percentage that exists inside the $\pm k\times$ standard deviations interval around the mean. The inequality can be applied for outliers detection [36] when the data distribution is unknown.

The equation that describes Bienaymé-Chebyshev's inequality is:

$$P(|\ X - \mu\ | \geq k\sigma) \leq \frac{1}{k^2} \tag{7}$$

where X is a random variable, μ is the mean, $k > 0$ is the deviation parameter and σ is the standard deviation. If we set the parameter $k = 4.47$ on Eq. (7), the resultant probability will be equal to 0.05, which is the usual cut-off point for statistical significance [37]. In case that a sample is higher than k standard deviations relating to the average, this point is considered anomalous.

On the construction of the anomaly detection module for this method, an adaptation was performed on the Bienaymé-Chebyshev's inequality to create an upper and a lower threshold to determine what is considered a normal traffic behavior based on the generated DSNSF. Eq. (8) and Eq. (9) are used to determine the upper and lower limits, respectively. A dimension is detected as being anomalous when the actual traffic is higher than the *UPPER* threshold or lower than *LOWER* limit.

$$UPPER = DSNSF + k\sigma \tag{8}$$

$$LOWER = DSNSF - k\sigma \tag{9}$$

After the individual detection of each one of the flow features, it is necessary to define when in fact a general anomaly occurred. According to [10], [11], and [38], each type of anomaly affects the traffic flows in different ways. For instance, in a Denial of Service (DoS) attack a high concentration (low entropy values) on the features "source IP addresses" and "destination ports," while on a Flesh Crowd event occurs a higher dispersion (high entropy values) of the features "source IP addresses" and "source ports." We summarize the types of anomalies and flow attributes affected by them on Tab. 1.

TABLE 1. Type of anomalies and affected attributes.

Type of anomaly	Affected attributes
DoS	Source IP, destination IP, destination port and quantitative attributes
DDoS	Source IP, destination IP, source port, destination port, and quantitative attributes
Port Scan	Source IP, destination IP, destination port and quantitative attributes

Based on the behavior of the flow features when an anomaly occurs, we consider that at least three dimensions are detected as anomalous to activate the mitigation module, *i.e.*, if in a given time interval any three traffic features was

detected as anomalous, then an alarm is triggered, and it is considered that a global anomaly occurred on this time interval.

B. MULTI-LAYER PERCEPTRON FOR DIGITAL SIGNATURE (MLP-DS)

MLP for Digital Signature (MLP-DS) is the term we designate the entire process of traffic characterization and anomaly detection using MLP network. The MLP is a neural network based on Perceptron networks operation which has at least one hidden neuron layer on its topology. According to Haykin [39], they are characterized by its wide application range within different areas, such as universal function approximator, pattern recognition, control and process identification, time series forecasting and systems' optimization.

One of the main characteristics that distinguish MLP from traditional Perceptron networks is the presence of hidden neuron layers. According to Haykin [39] the hidden neurons act as characteristics detectors, performing a key role in the network operation.

In brief, MLP are neural networks of supervised learning operating without feedback. The data are input separately through the ''input layer.'' These data tend to be normalized to optimize the learning process of the network. Furthermore, the MLP topology is composed of at least one neuron's hidden layer and by an output layer, which will present the results of the network's classification. The topology is represented as a fully connected graph, *i.e.*, each input signal is connected to each one of the intermediary layer neurons. In turn, these neurons may be connected to each one of the neurons of a second hidden layer (if applicable) or each one of the output signals. A topology example is depicted in Fig 3.

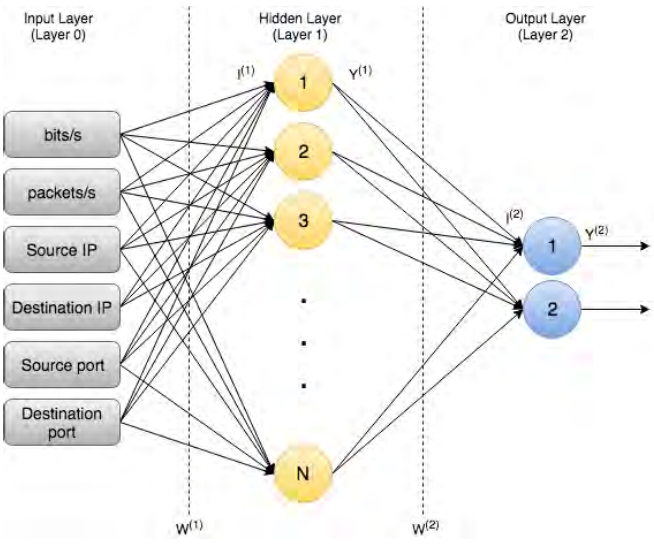


FIGURE 3. MLP topology with one hidden layer with N neurons and binary output.

Furthermore, each one of the connections is initialized with a random value from 0 to 1, representing the synaptic weights of each connection. These weights are adjusted during the learning process in order to allow the group classification, as previously described.

In this paper, MLP is used for SDN traffic characterization and anomaly detection processes on the analyzed network segment. It is important to highlight that this process is entirely performed in an autonomic way, without any network administrator's interference in the processes described herein.

As previously discussed on Section III, after the exportation of the collected flows into files, data relating to the analyzed IP flow dimensions are extracted in separate files so that they can be subjected to a traffic characterization process. This process is performed by using the Multi-Layer Perceptron (MLP) method, representing the training step.

Six flow dimensions are applied to the MLP method, responsible for learning the pattern behavior (DSNSF) of the SDN's normal and abnormal operation. The MLP-DS approach is able to detect not only DDoS attacks but also DoS and port scans. Thus, to enable the classification of four different states, the MLP was designed with two neurons. The outputs of the neurons are coded following Tab. 2. Fig. 3 shows the MPL topology used in this paper.

TABLE 2. Output Encoding for MLP-DS.

Detected behavior	Neuron $Y_1^{(2)}$	Neuron $Y_2^{(2)}$
Normal	0	0
DoS	0	1
DDoS	1	0
Port scan	1	1

The learning process of the MLP consists in submitting the neural network to a set of labeled data, *i.e.*, data of the six analyzed IP flow dimensions in addition to a label that describes whether this combination represents a normal traffic, a DDoS or a port scan behavior. However, even normal traffic data present different behaviors along a day of analysis. This occurs due to the fact that, for instance, the traffic early in the morning is different from the traffic on working hours, which does not make them anomalous. Similarly, different intensities of the same attack may generate different signatures for the same type of attack. In this manner, to improve the classification results of the MLP, the learning process is submitted through a clustering approach.

The first step is to separate the anomalous from the normal traffic training (labeled) data. Then, both groups are submitted to a clustering process using the K-means algorithm [40] in order to identify similar groups within the analyzed data. After this step, a sampling is performed within the different clusters in order to generate a reduced group that is able to represent the entire analyzed data. This sampling is important to the MLP learning to avoid a problem known as over-fitting, which impairs the classification results when the training dataset is too large. Finally, the sampled data from normal and anomalous traffic are united into a single training group, which is submitted as input for the MLP training method. The basic operation of the training process can be described in Algorithm 2.

Algorithm 2 - MLP Training Phase

1: Receive the input training (sample) data set
2: Associate the desired output (label) to all training data
3: Initialize the synaptic weights with random small values
4: Specify the learning rate η
5: Specify the required precision ϵ
6: **while** $|MSE_{current} - MSE_{previous}| > \epsilon$ **do**
7: $MSE_{previous} \leftarrow MSE$
8: **for** each sample data **do**
9: Calculate $I_j^{(1)}$ and $Y_j^{(1)}$; *(Eq. (10) and (12))*
10: Calculate $I_j^{(2)}$ and $Y_j^{(2)}$; *(Eq. (11) and (13))*
11: Calculate $\delta_j^{(2)}$; *(Eq. (16))*
12: Adjust synaptic weights $W^{(2)}$; *(Eq. (18))*
13: Calculate $\delta_j^{(1)}$; *(Eq. (15))*
14: Adjust synaptic weights $W^{(1)}$; *(Eq. (17))*
15: Calculate $Y_j^{(2)}$ through steps 9 and 10;
16: Calculate MSE; *(Eq. (19))*
17: $MSE_{current} \leftarrow MSE$;
return Trained synaptic weights

In this algorithm, I_j^k stands for the weighted sum performed by the neuron j at the layer k (Eq. (10) and (11)):

$$I_j^{(1)} = \sum_{i=0}^{D} W_{ji}^{(1)} \cdot x_i \tag{10}$$

$$I_j^{(2)} = \sum_{i=0}^{N} W_{ji}^{(2)} \cdot Y_i^{(1)} \tag{11}$$

where D is the number of analyzed flow dimensions, $W_{ji}^{(1)}$ are the synaptic weights that connects neuron j to neuron i of the following neural layer, and x_i is the i-est neuron at the input layer. Y_j^k stands for the calculated output of the Perceptron j at the layer k (Eq. (12) and (13)):

$$Y_j^{(1)} = g(I_j^{(1)}) \tag{12}$$

$$Y_j^{(2)} = g(I_j^{(2)}) \tag{13}$$

where $g(\cdot)$ is an activation function. In this paper, we adopt the logistic activation function with inclination parameter β of 1, defined by Eq. (14).

$$g(u) = \frac{1}{1 + e^{\beta \cdot u}} \tag{14}$$

The variable δ_j^k is the local gradient of the neuron j at the layer k (Eq. (15) and (16)):

$$\delta_j^{(2)} = (d_j - Y_j^{(2)}) \cdot g'(I_j^{(2)}) \tag{15}$$

$$\delta_j^{(1)} = \left(\sum_{i=0}^{N} \delta_i^{(2)} \cdot W_{ij}^{(2)} \right) \cdot g'(I_j^{(1)}) \tag{16}$$

where the variable d_j represents the expected output for neuron j, $Y_j^{(2)}$ represents the output calculated for neuron j by the MLP and $g'(.)$ is the derivative of Eq. (14) regarding I_j.

The synaptic weights W_{ji} connecting neurons j to i of each layer are updated through the Eq. (17) and (18):

$$W_{ji}^{(1)} \leftarrow W_{ji}^{(1)} + \eta \cdot \delta_j^{(1)} \cdot x_i \tag{17}$$

$$W_{ji}^{(2)} \leftarrow W_{ji}^{(2)} + \eta \cdot \delta_j^{(2)} \cdot Y_i^{(1)} \tag{18}$$

where η is the learning rate, herein defined as 0.2 after exhaustive performance testing, and x_i is the i-th neuron at the input layer. The value η directly influences the convergence outcomes. When this value is high, the convergence can be quickly achieved. However, in this case, it is possible for the method to be unable to find the convergence due to the learning rate step. Lower η values mean a slower and more secure convergence process.

The Mean Square Error (MSE) between the desired (step 2 on Algorithm 2) and the achieved output is performed computing:

$$MSE = \frac{1}{p} \sum_{k=1}^{p} \varepsilon(k) \tag{19}$$

where p is the number of data samples analyzed and $\varepsilon(\cdot)$ is the square error, achieved through Eq. (20).

$$\varepsilon(k) = \frac{1}{2} \sum_{j=1}^{N2} (d_j(k) - Y_j^{(2)}(k))^2 \tag{20}$$

In this Equation, $N2$ is the number of neurons at the output layer. Finally, the variable ϵ (line 6 of Algorithm 2) represents the required precision of the results, herein defined as 10^{-7}.

After the training process, the calculated synaptic weights can be imported into the SDN controller, which will execute the classification process using the steps described by Algorithm 3. The computational cost of the MLP is high only in the training process, which only needs to be performed once. After this, a lightweight process of classification is performed every 5 seconds by the SDN controller and, if a DDoS or a port scan attack is detected, an alarm is triggered to invoke the Identification and the Mitigation modules, as described in Section III.

Algorithm 3 - MLP Operation Phase

1: Receive the input sample to classify;
2: Import the synaptic weights calculated with Algorithm 2;
3: Calculate $I_j^{(1)}$ and $Y_j^{(1)}$; *(Eq. (10) and 12)*
4: Calculate $I_j^{(2)}$ and $Y_j^{(2)}$; *(Eq. (11) and 13)*
return Classification provided by $Y_j^{(2)}$;

C. DISCRETE WAVELET TRANSFORM (WAVEDETECT)

The proposed anomaly detection called WaveDetect uses Discrete Wavelet Transform (DWT) [41] for anomaly detection. DWT consists of a mechanism to decompose or break

signals (data) into their constituent spectral parts, *i.e.*, into different frequencies components [42].

These constituent parts are called coefficients, and the different frequencies are obtained throughout DWT decomposition levels. These coefficients can be of two types: approximation (scaling), or detail (wavelet). The approximation coefficients ($c_{j,k}$) are responsible for the coarsest information of the input signal, which consists of the lower frequencies. The detail coefficients, represented by $d_{j,k}$, carry the high frequencies of the previous level data.

These coefficients are obtained by a filter bank application, which consists of a matrix multiplication of high-pass (h) and low-pass (g) filters by the input data. By high-pass filters we obtain detail coefficients ($d_{j,k}$), and by low-pass filters, the approximation coefficients ($c_{j,k}$) at DWT decomposition level j with k elements, where $k = N/2$ and N being the input data size. The filter is defined by the wavelet function used [43]. In the proposed solution, the wavelet function chosen was Haar, because this kind of wavelet is computationally tractable and provides a low computational cost [44], [45]. DWT decomposition is depicted in Fig. 4.

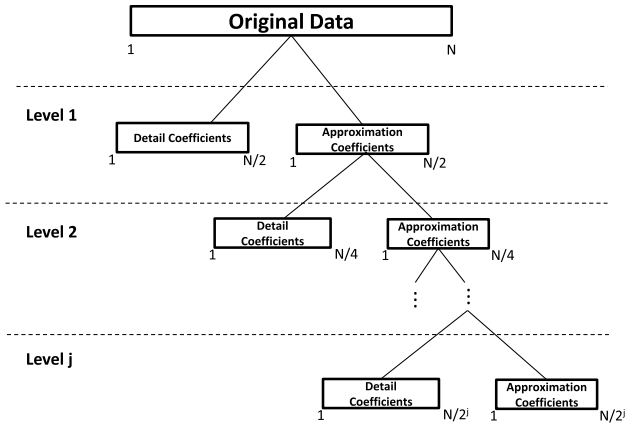


FIGURE 4. DWT decomposition process.

DWT can also be represented by Eq. (21), where $\varphi_{j_0,k}(t)$ is the scale function, also known as father wavelet, generating approximation coefficients $c_{j_0,k}$ from level j_0 and $\psi_{j,k}(t)$, which is the wavelet function, or mother wavelet, defining the filters from the DWT, and generating detail coefficients $d_{j,k}$ for all DWT decomposition levels; both wavelet coefficients are composed by k elements, where k depends of input data size.

$$f(t) = \sum_{k=1}^{\frac{N}{2}} c_{j_0,k}(k)\,\varphi_{j_0,k}(t) + \sum_{j=j_0}^{\infty}\sum_{k=1}^{\frac{N}{2}} d_{j,k}\psi_{j,k}(t) \quad (21)$$

The proposed WaveDetect method is divided in two stages, the first is the Traffic Characterization level and the second is the Anomaly Detection level.

The WaveDetect method is basically a comparison between two sliding windows, the first (W_f) representing the traffic forecast, and the second (W_d) carrying the traffic with the sample that will be analyzed. Both windows and levels are going to be explained following.

1) FIRST LEVEL: TRAFFIC CHARACTERIZATION

It is responsible for sliding W_f and W_d. The traffic forecast window W_f will only slide to incorporate the forwarding traffic sample if this sample was previously classified as a normal point; if so, the oldest W_f sample is replaced by the new sample, *i.e.*, W_f will always carry the last M samples with normal traffic.

The second window, W_d has the same size of W_f and contains the last $M-1$ points from W_f and its last point is the sample of interest, that is, the sample that is going to be analyzed. This window slides excluding the oldest point and including the following point. Fig. 5 depicts a visual explanation. Also, a mathematical explanation of both windows is depicted in Eq. (22)–(24), for a sample t.

$$|W_f| = |W_d| = M \quad (22)$$

$$W_f = \left\{w_f(t-1)-M, w_f(t-M), \ldots, w_f(t-1)\right\} \quad (23)$$

$$W_d = \{w_d(\text{t-M}), w_d(t-M)+1, \ldots, t\} \quad (24)$$

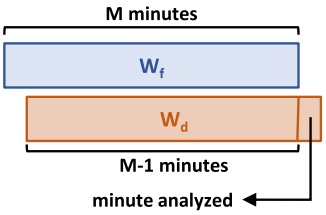


FIGURE 5. Sliding windows W_f and W_d.

As the solution uses previous data to generate W_f and W_d, when starting the Traffic Characterization level, it is required a database with the last M minutes considered within the pattern, for a bias-free forecast. As explained previously, W_f and W_d have size M. The value of M ranges between 16 and 8192. The proposal of using different sizes for M was to find the amount of historical traffic which best describes the network traffic. Also, all values of M range in values multiples of a power of 2, to facilitate the DWT decomposition process, as this process is similar to binary tree division. After tests that will be further explained in Results and Analysis section, the value chosen for M was 1024.

2) SECOND LEVEL: ANOMALY DETECTION

This level aims for the detection of DDoS and port scan attacks. This level is divided into two main stages: DWT and then DDoS couplet with port scan detection. The first stage performs a one-dimensional DWT in W_f and W_d, with one decomposition level. The decision of use one decomposition level was made based on tests explained in Results and Analysis section.

The second stage performs anomaly detection for each dimension. This process is divided in three steps. The first step calculates an interquartile range (IQR), obtained through

the approximation coefficients calculated using W_f. IQR is detailed in (25) and (26) and, according to Hoaglin [46], this approach was proposed by John Tukey based on normal patterns of diastolic blood pressures.

$$IQR = Q3 - Q1 \tag{25}$$

$$(Q1 - (IQR \cdot 1.5)) \leq X \leq (Q3 + (IQR \cdot 1.5)) \tag{26}$$

where X is the data analyzed, $Q1$ is the first quartile and $Q3$ is the third quartile.

After this, the second step of DDoS and port scan detection compares the last value of approximation coefficients ($c_{1,N/2}$) from W_d with the interval obtained in Eq. (25). If it is within the interval, the traffic is classified as normal traffic, otherwise, the traffic is classified as anomalous for an specific dimension. Also, if the analyzed traffic is classified as anomalous, WaveDetect evaluate if the traffic is higher or lower than the normal behavior. It will help on a later anomaly classification.

The third and last step of detection accomplishes classification of anomalous traffic in DDoS or port scan. When a DDoS or a port scan attack is being performed, some changes occur in specific IP flow traffic dimensions (features). A DDoS attack increases source port entropy, decreases destination IP and port entropies and discreetly decreases the source IP entropy. A port scan attack modifies basically three dimensions, increasing destination port entropy, and decreasing source and destination IP entropies. By comparing this information with the anomaly detection output, it is possible to identify which anomaly is on the traffic sample. Algorithm 4 details the stages from Anomaly Detection level.

Algorithm 4 - WaveDetect Detection Phase

1: Calculate $f_{W_f}(t)$ and $f_{W_d}(t)$
2:
3: Calculate $Q1$ and $Q3$ of $c_{(j,k)}$ from W_f
4: **if** $\left((c_1, \frac{N}{2}\right)$ from W_d is greater than $(Q1 - \text{IQR} \cdot 1.5)$ and smaller than $(Q3 + \text{IQR} \cdot 1.5)$) **then**
5: Classifies traffic as Normal
6: **else**
7: Classifies traffic as Anomalous
8: Identifies DDoS or port scan pattern

V. RESULTS AND ANALISYS

In this section we discuss the achieved results of performance tests of the presented SDN defense system. This analysis is performed in two sub-sections. In the first one, we discuss the parameters used by the methods described in Section IV. In the second one, we show the performance outcomes achieved by the presented system on anomaly detection through the comparison between the three presented detection methods and classic anomaly detection methods in literature. Furthermore, the mitigation performance over the port scan and DDoS attacks is discussed.

A. PARAMETERS ESTIMATION

The first step performed by any swarm based optimization algorithm, like PSO-DS, is the estimation of the individuals population, *i.e.*, the possible problems' solution. Knowing that the population of individuals is randomly generated, the population size must be chosen with caution. Through empirical tests applied by Bratton and Kennedy [47], the authors define the number of individuals comprises between 20 and 100 particles. To evaluate the amount of particles, we applied the Normalized Mean Square Error (NMSE) [48] for a day of PSO-DS traffic characterization. The NMSE calculates the absolute difference between the generated DSNSF and the actual SDN traffic. This metric generates outcomes between zero to infinite, where values close to zero indicate a good traffic characterization, while higher values indicate that the forecasting performed by the characterization process diverges from the actual traffic behavior. According to Fig. 6, the number of particles with lower NMSE outcome, *i.e.*, that generates the best traffic characterization for PSO-DS, is 50 particles.

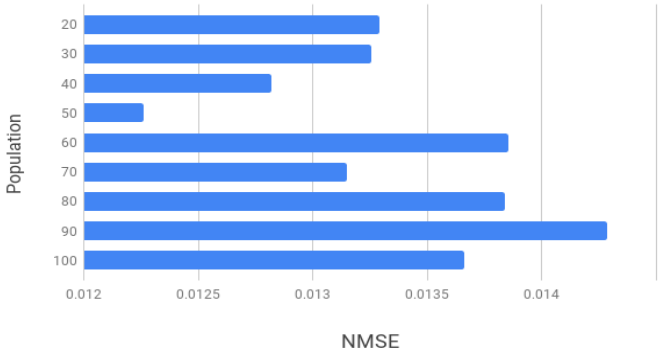


FIGURE 6. Estimation of population size for PSO-DS.

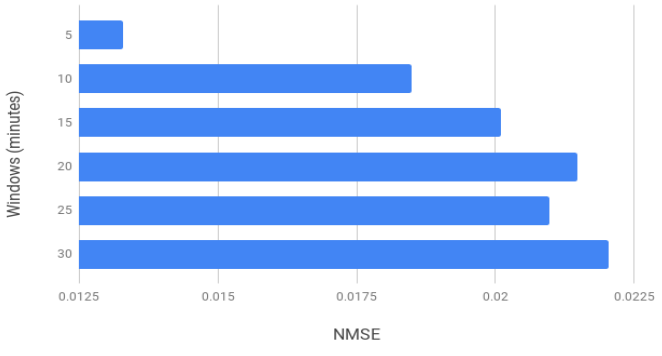


FIGURE 7. Estimation of time window size for PSO-DS.

For the generation of network traffic signatures, the PSO-DS uses data from the last n past minutes of the real traffic. The convergence evaluation of the used time window was performed through the NMSE. The values tested for n comprise between 5 and 30 minutes. As observed in Fig. 7, the time window which achieved best results was $n = 5$ minutes.

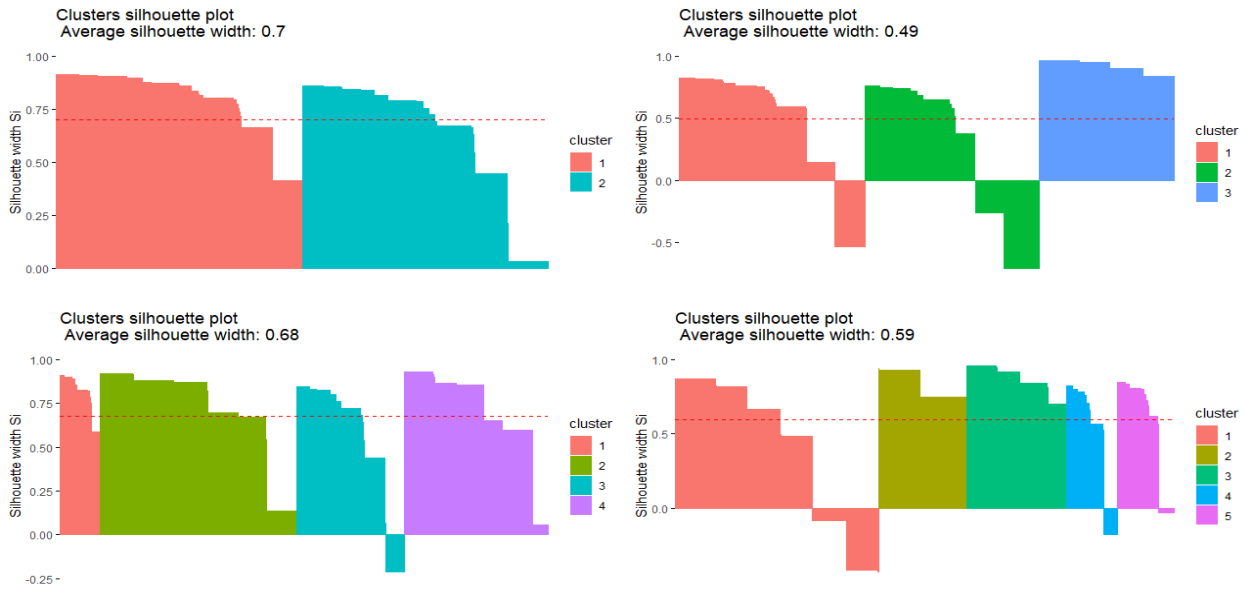


FIGURE 8. Silhouette technique used for estimation of the amount of clusters.

PSO-DS is based on the traffic characterization through the clusterization of flows extracted form the SDN controller. Thus, the Silhouette technique [49] was used to estimate the number of clusters used by PSO-DS. The application of this technique provides a graphical representation of the elements' arrangement inside each cluster. This graphical representation is useful when the proximity metric is in scale (like in the case of Euclidean distance) and when compact and clearly separated clusters are required. The outcome of this function comprise between $-1 \leq f(s) \leq 1$, and have three interpretations for the results. When $f(s)$ is close to -1, it means that the sample i was misclassified, *i.e.*, the element should be assigned to another cluster. When the value of $f(s)$ tends to zero, it is an intermediate case, which means that the element i could be assigned to more than one cluster. The best case is when $f(s)$ is close to 1, which indicates the existence of a high similarity between the element i and the other elements belonging to the cluster. Fig. 8 presents the values of C (number of clusters) varying from 2 to 5. From the analysis of C, it is possible to note that the best achieved outcome was achieved using 2 clusters. According to Fig. 8, the function did not obtain zero or negative outcomes, *i.e.*, no element was assigned to the cluster erroneously.

For the MLP-DS method, some of the variables are given by the stated problem. The number of input neurons is defined as 6 since there is six different analyzed flow dimensions used on traffic characterization and anomaly detection problem. The number of neurons on the output layer is defined as 2 due to the codification used on the classification process, as described by Tab. 2. As discussed in [27], the number of neurons on the hidden layer influences the MLP outcomes up to $2N + 1$ neurons, where N is the number of neurons on the input layer. This can be observed in Fig. 10.

As observed, using $2N$, $2N + 1$ or $2N + 2$ neurons on the hidden layer does not further improve the MLP classification outcomes. Thus, the number of neurons on the hidden layer was defined as 12. Another parameter used on the MLP-DS method is the number of clusters used on the training process. Fig. 11 shows the results relating to the estimation of this parameter.

As shown, there is an improvement on the classification efficiency of the MLP when using the described clustering approach on the analyzed data in comparison with the case where no clustering is used (when the number of clusters is 1). From 2 to 5 clusters, there was no considerable efficiency difference on the classification outcomes. Thus, the number of clusters was defined as 2. Fig. 12 shows as example a radar-plot of the network's normal behavior before and after the clustering process.

To evaluate the window size and the DWT decomposition level that provides the best detection, some tests were performed. For this, ten different window sizes (M) were defined, which are: $M = 16, 32, 64, 128, 256, 512, 1024, 2048, 4096$ or 8192 samples. For each M, tests with different DWT decomposition levels, from one to four levels were performed as well. To assess all tests, six metrics (Precision, Accuracy, False Positive rate[FP-rate], Recall, F-measure and Area Under the ROC Curve [AUC]) were applied. Fig. 9 and 13 present the best result for each value of M, *i.e.*, the level which provided the best result for each M. By analyzing both figures it is possible to conclude that using DWT with small windows (from 16 to 64 historical traffic's samples) presented bad detection results and high false-positive rates. Values of M greater than 64 and smaller than 4096 provided better results than using smaller windows, but its best result was achieved using $M = 1024$, which provided a detection rate of $99,32\%$, false-positive rate of $0,03\%$ and AUC equal to $99,45\%$. Windows bigger than 4096 showed a decreasing on Detection rates, with an average detection of $93,58\%$. An analysis of decomposition

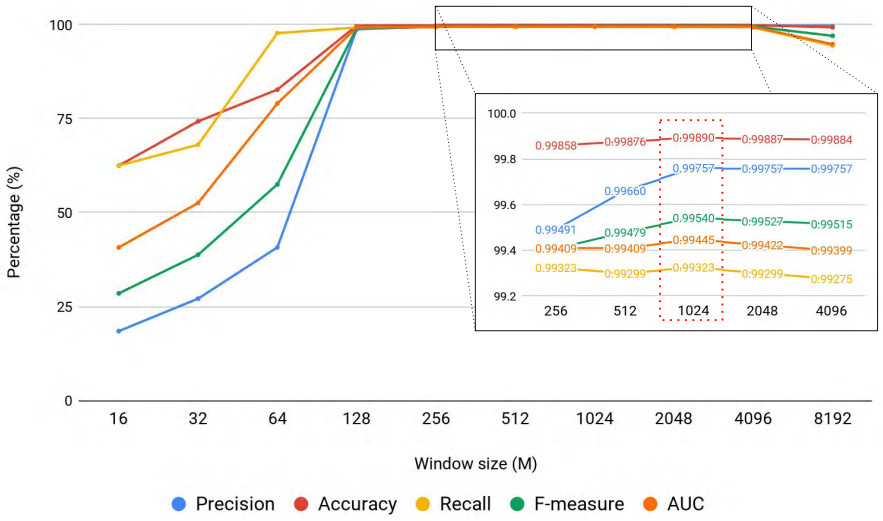


FIGURE 9. Results from five metrics to different window size and decomposition level of WaveDetect.

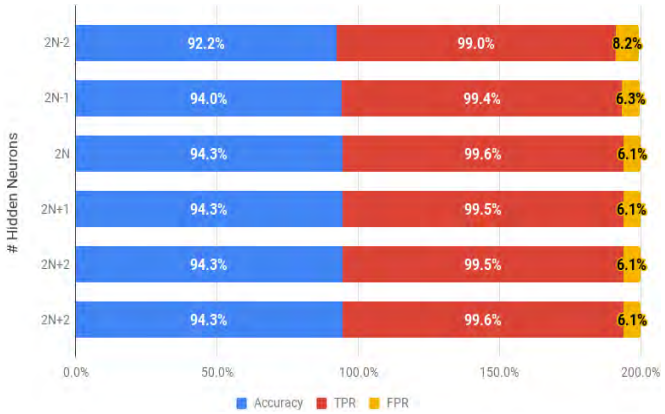


FIGURE 10. Estimation of required number of neurons on the hidden layer for MLP-DS.

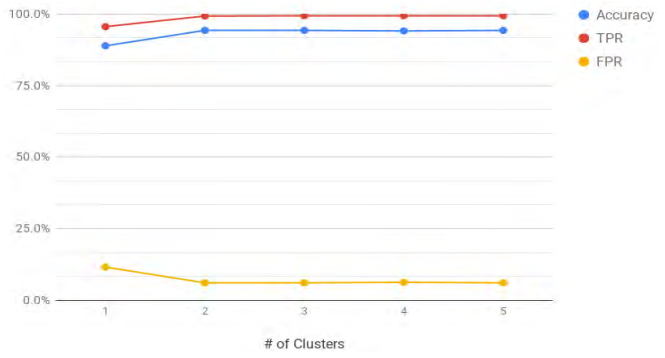


FIGURE 11. Estimation of required number of clusters for MLP-DS training process.

levels were also made. Using one level regardless the window size, detection rates were better than using other levels, and false-positive rates were better using four decomposition levels. Considering all metrics and analyzing the use of each level, regardless the window size, one level of decomposition

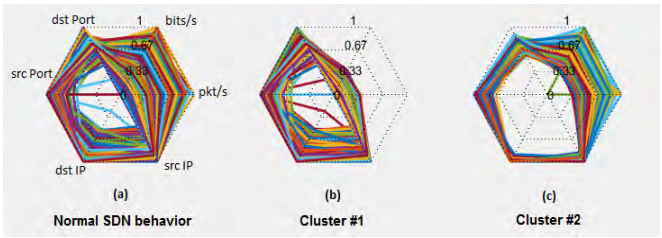


FIGURE 12. Radarplot describing a normal SDN behavior, showing the analyzed day without clustering (a) and the two clusters generated using k-means method (b and c). The lines represent the 6-dimensional view of the SDN behavior in each analyzed time interval.

presented the best results. It can be explained by the fact that approximation coefficients carry the coarsest part of input data, so the deeper the level, the coarser will be the representation from original data. So by all these analysis, the value of M and the level chosen were 1024 and one, respectively.

B. PERFORMANCE OUTCOMES

Here we discuss the achieved outcomes of the performance tests. To perform these tests, we used simulated data generated by Mininet network emulator [50], a tool that allows the creation of realistic virtual networks composed by controllers, hosts, links, and switches in a single virtual machine. Mininet uses a lightweight virtualization on the creation of custom topologies through simple command lines. The experiments conducted in this paper used the Open vSwitch to control the network's switches, as Mininet offers support for it.

To implement the anomaly detection and mitigation mechanism, we used the SDN controller Floodlight, a Java-based controller widely used in literature. Finally, the data collected and managed are from the OpenFlow protocol. The SDN topology emulated is composed of four switches in a tree-like

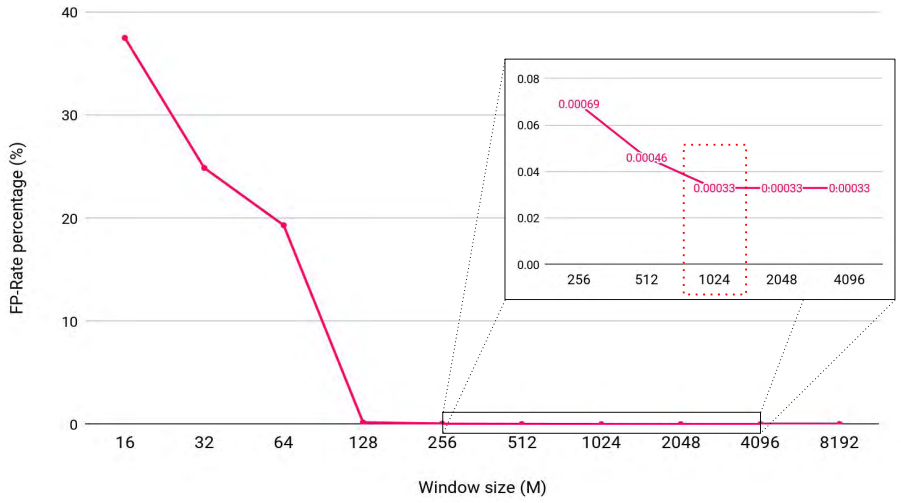


FIGURE 13. Results from FP-rate to different window size and decomposition level of WaveDetect.

TABLE 3. Description of test data.

	Attack 1	Attack 2	Attack 3
Day 2	Type: DDoS Attackers: 5 Attacking IPs: 10.0.0.20 - 10.0.0.24 Destination IP: 10.0.0.58 Duration: 07:00:00 - 07:59:40	Type: DDoS Attackers: 10 Attacking IPs: 10.0.0.21 - 10.0.0.30 Destination IP: 10.0.0.10 Duration: 12:00:00 - 13:02:20	Type: DDoS Attackers: 15 Attacking IPs: 10.0.0.21 - 10.0.0.35 Destination IP: 10.0.0.1 Duration: 15:00:00 - 15:59:05
Day 3	Type: Port Scan Attacking IP: 10.0.0.60 Destination IP: 10.0.0.30 Ports: 1 - 14961 Seconds to wait between 2 packets: 0.2 Duration: 06:00:00 - 06:59:40	Type: Port Scan Attacking IP: 10.0.0.41 Destination IP: 10.0.0.35 Ports: 1 - 19999 Seconds to wait between 2 packets: 0.1 Duration: 10:00:00 - 10:46:30	Type: Port Scan Attacking IP: 10.0.0.35 Destination IP: 10.0.0.1 Ports: 1 - 19126 Seconds to wait between 2 packets: 0.15 Duration: 16:00:00 - 16:56:55

topology, where one root switch connects the other three, each one connecting twenty different hosts. One of these switches represents a border gateway, and between its hosts are normal and malicious users. To guarantee that the emulated scenario is as close as possible to a real SDN environment, with high traffic rates passing through the network, in our experiments we used a tool named Scapy [51] to inject the emulated network with traffic. The data collection was performed using a REST API provided by Floodlight, which sends requests to a flow controller of a switch every five seconds.

Three days (72 hours) of SDN traffic were generated and used on our performance tests. Each one of the generated days emulates the normal behavior observed at the State University of Londrina (UEL), Brazil, where there is an increase in network usage in the morning (from 8:00 to 11:30) and in the afternoon (from 14:00 to 17:00). In the evening, the network usage is less intense, but similar to mornings.

The first generated day was injected with two occurrences of port scan and DDoS attacks of different intensities. This day was used for MLP-DS training process, since it is a supervised machine learning method. As PSO-DS and WaveDetect need no previous training, this data was not used by them.

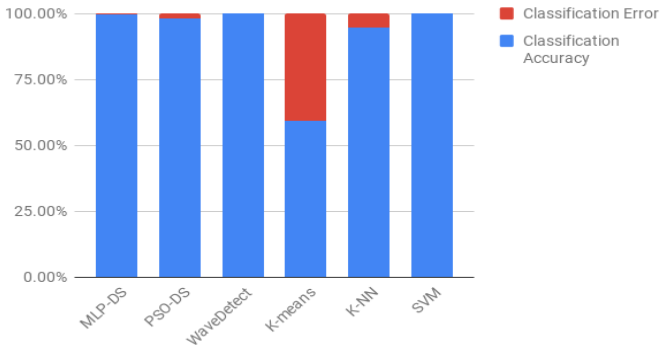


FIGURE 14. Classification Accuracy (CA) and Classification Error (CE) on anomaly detection for the analyzed methods.

The next two generated days were used on testing the efficiency of both anomaly detection and attack mitigation of the presented system. Three DDoS and port scan attacks of different intensities and in different time intervals were injected into SDN traffic, separating the DDoS attacks in one SDN traffic day and the port scan attacks on the another one. Tab. 3 describes the parameters of the performed attacks.

TABLE 4. Achieved results on DDoS and Port Scan detection of the tested methods.

	MLP-DS	PSO-DS	WaveDetect	K-means	K-NN	SVM
True Positive rate (TPR)	99.71%	99.66%	99.32%	64.80%	76.84%	99.37%
True Negative rate (TNR)	99.74%	97.85%	99.97%	58.62%	96.90%	99.98%
False Positive rate (FPR)	0.26%	2.15%	0.03%	41.38%	3.10%	0.02%
False Negative rate (FNR)	0.29%	0.34%	0.68%	35.20%	23.16%	0.63%
Positive Prediction Value (PPV)	98.12%	86.29%	99.76%	17.55%	77.14%	99.83%
Negative Prediction Value (NPV)	99.96%	99.95%	99.91%	92.45%	96.85%	99.91%
Classification Accuracy (CA)	99.74%	98.06%	99.89%	59.36%	94.50%	99.90%
Classification Error (CE)	0.26%	1.94%	0.11%	40.64%	5.50%	0.10%

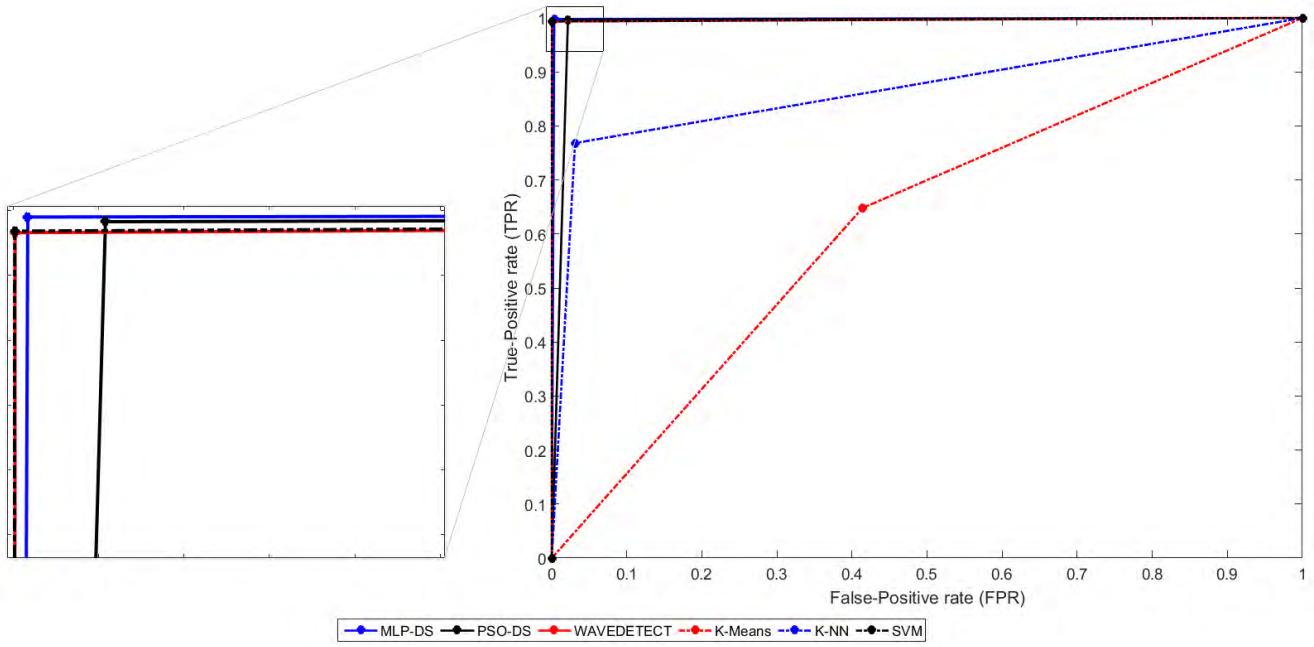


FIGURE 15. Roc Curve of the tested methods.

1) ANOMALY DETECTION

To test the efficiency of the presented methods on detecting port scans and DDoS attacks, we compare them with well stated anomaly detection algorithms, such as K-Means [40], K-Nearest Neighbors (K-NN) [52] and Support Vector Machine (SVM) [53]. The metrics used are classical anomaly detection statistic techniques [54], such as True and False Positive Rates (TPR and FPR), True and False Negative Rates (TNR and FNR), Positive and Negative Prediction Value (PPV and NPV), Classification Accuracy (CA) and Classification Error (CE). The results achieved by them are described on Tab. 4.

As observed, the presented methods fared better than K-means and K-NN classic approaches on most analysis scenarios. The SVM achieved performance is very similar to the one generated by WaveDetect method. The overall outcomes were good for MLP-DS, PSO-DS and SVM methods, with high TPR and TNR rates, and low FPR and FNR rates.

Fig. 14 shows the classification accuracy and classification error of the tested methods. As shown, the most inaccurate

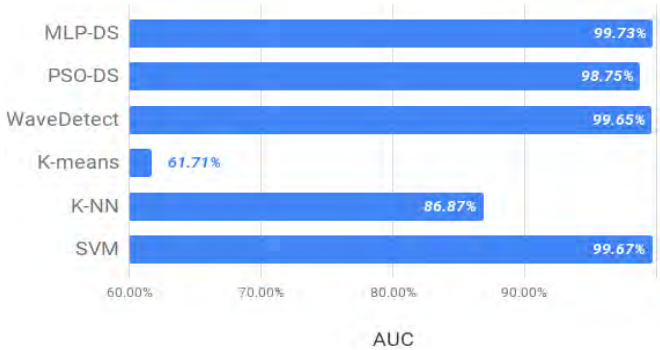


FIGURE 16. Area Under the Curve (AUC) of the generated ROC curves.

method on detecting anomalies was the K-means approach, with CA rate of nearly 60%, followed by K-NN method that, even though achieved an accuracy rate of 94.5%, misclassified around 23% of normal traffic intervals (FNR). MLP-DS, PSO-DS, WaveDetect and SVM methods achieved similar results, although PSO-DS faring slightly worse than the others due to its FPR rate of 2.15%.

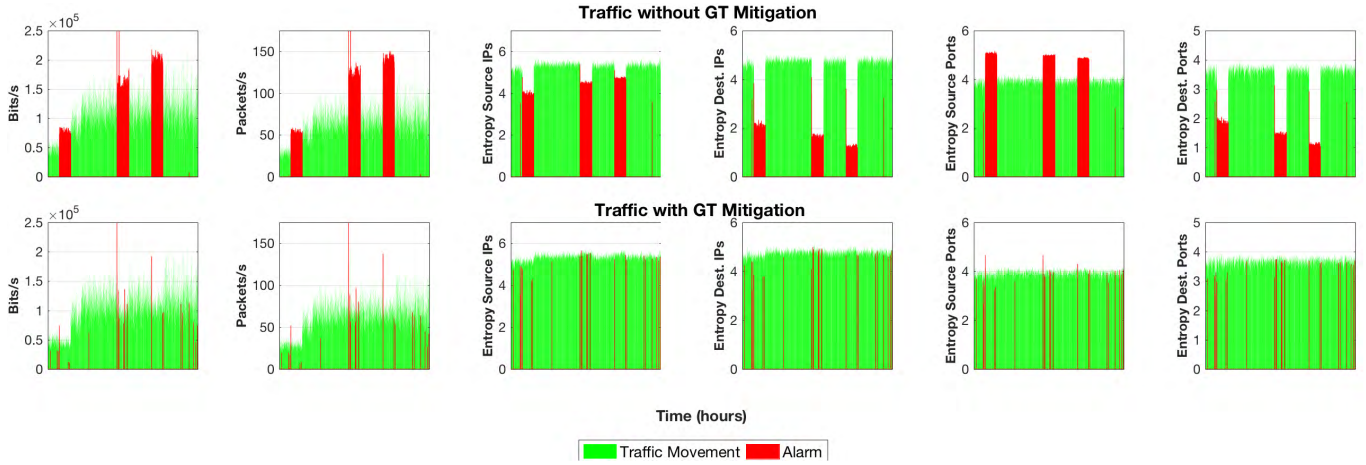


FIGURE 17. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using PSO-DS and GT-approach.

Furthermore, a Receiver Operating Characteristic (ROC) curve was constructed using the TPR and FPR rates for all tested methods to measure the classification efficiency of them. The ROC curve is shown in Fig. 15. As seen, the classification outcomes of MLP-DS, PSO-DS, WaveDetect and SVM are visually better than the ones achieved by K-Means and K-NN approaches. Their outcome was so similar that a zoom was needed in order to analyze their differences. As observed, PSO-DS method have the higher FPR rate, followed by MLP-DS, while WaveDetect and SVM achieved similar results for this metric. However, PSO-DS and MLP-DS achieved higher TPR rates than WaveDetect and SVM approaches.

To better quantify the efficiency of the tested methods, we analyze the area under the curve (AUC) of the ROC curve. The outcomes of this analysis is shown by Fig. 16. This figure shows that, relating to TPR and FPR rates, MLP-DS have the better trade-off, followed by SVM, WaveDetect, PSO-DS, K-NN and K-means.

It is noteworthy the differences between the analyzed methods. MLP-DS, K-NN and SVM methods needed to be trained before applied on the presented SDN defense system. In turn, PSO-DS, WaveDetect and K-Means need no previous data training for the anomaly detection. Furthermore, MLP-DS and K-NN methods are able to directly detect if the detected anomaly is a DDoS or a port scan attack, while the other methods only detects the anomaly occurrence. For those methods, the Identification Module of the presented SDN defense system is responsible for identifying the type of the anomaly in order to trigger the correct countermeasure on the Mitigation Module, as described in Section III.

Finally, to compare the computational efficiency and detection speed of the presented methods we compare their computational complexity. Although the computational complexity of MLP-DS method is $O(W^3)$ for the training step, where W is the number of synaptic weights of the neural network, for the operation step its complexity is asymptotically given by $O(W)$. As the number of synaptic weights is constant, the overall complexity of the operation step is $O(1)$, the lowest complexity among the presented methods. The WaveDetect is a Wavelet-based approach that carries out one DWT decomposition level. Thus, its computational complexity is $O(N)$, which is a linear complexity, where N is the size of the sparse Wavelet filters matrix. Finally, evolutionary algorithms, such as PSO-DS, have the complexity of $O(N*P*F)$ for each iteration, where N is the dimension of the problem, P is the population size, and F is the objective function size. Thus, we conclude that MLP-DS is the fastest of the presented anomaly detection approach, followed by WaveDetect (linear complexity) and PSO-DS faring worse, due to the need of a clustering process on each iteration.

2) MITIGATION PROCESS

In this section we discuss the outcomes achieved by the Mitigation Module of the presented SDN defense system against DDoS and port scan attacks. As previously discussed in Section III, the Mitigation Module receives data from the Identification Module, which is responsible of providing the system with relevant information about the detected anomaly, such as the most frequent source and destination IP addresses and ports.

When a DDoS attack is detected, either by the Identification module or by the anomaly detection method itself, the presented SDN defense system triggers a game theoretical (GT) approach to automatically defines the optimal drop rate to mitigate the attack while minimizing impact on legitimate users. Figures 17, 18 and 19 show the traffic on the six analyzed SDN dimensions from 6am to 7pm, before and after the DDoS GT mitigation approach.

As observed, the three anomaly detection methods tested presented similar results on detecting the DDoS attack, correctly triggering the alarms. However, after the mitigation process, PSO-DS generated a higher amount of false-positive alarms than MLP-DS and WaveDetect methods. As the

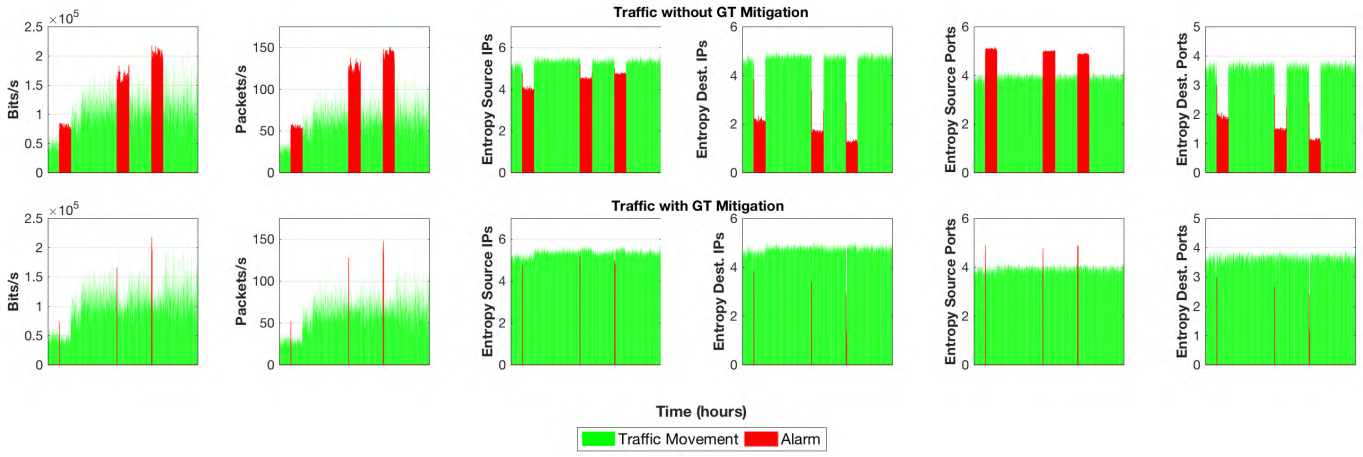


FIGURE 18. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using MLP-DS and GT-approach.

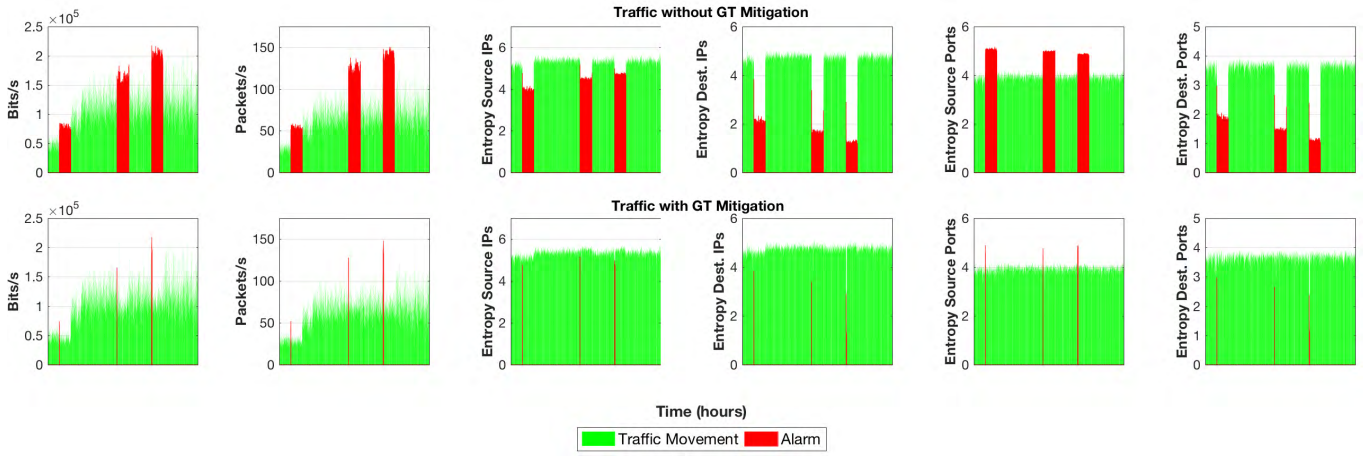


FIGURE 19. SDN six dimensional traffic movement, with three DDoS attacks, before and after the detection and mitigation processes using WaveDetect and GT-approach.

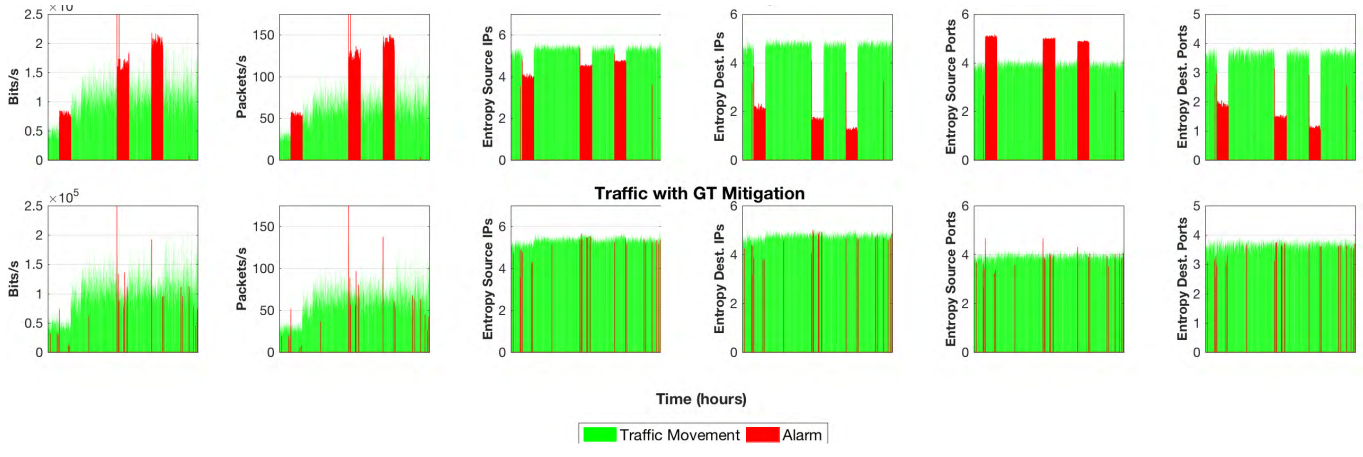


FIGURE 20. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using PSO-DS and directed drop policy.

mitigation succeeded on bringing the SDN back to a regular state, PSO-DS identified small traffic deviations as anomalous and, thus, generated the false-positive alarms. On the other hand, MLP-DS and WaveDetect achieved similar results, and the red lines present on the SDN traffic after the mitigation process are the time intervals where the attack was detected (if an attack is detected at 07:00:05, the mitigation starts only at the following time interval, *i.e.*, 07:00:10).

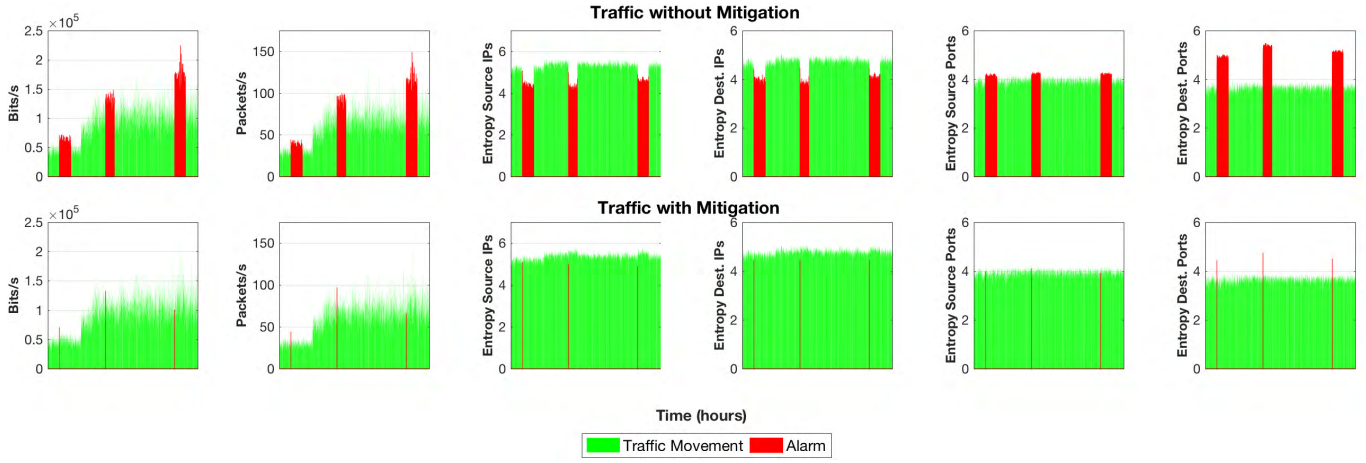


FIGURE 21. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using MLP-DS and directed drop policy.

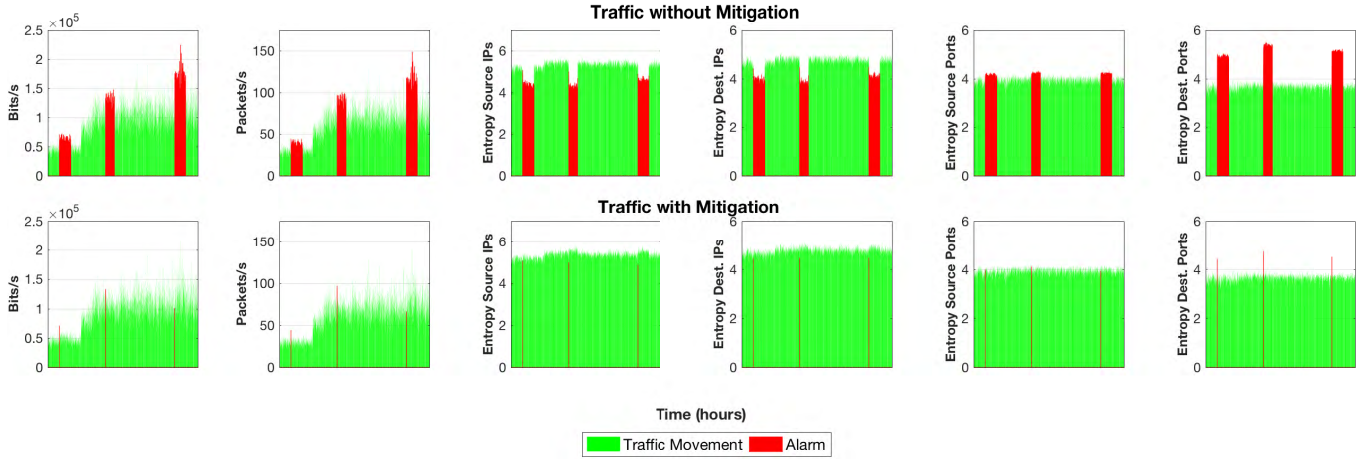


FIGURE 22. SDN six dimensional traffic movement, with three Port scan attacks, before and after the detection and mitigation processes using WaveDetect and directed drop policy.

Relating to port scan attacks, there is no need for the GT-approach on the mitigation process. This is due to the characteristics of this attack, since it is a centralized active scanning where a single host scans a range of ports of another. This generates a singular behavior, which is collected by the Identification module, enabling the isolation of the attacks' source IP address. With this feature, the Mitigation module sends a directed drop policy (drop of a single source IP address) to the SDN central controller. Figures 20, 21 and 22 show the traffic on the six analyzed SDN dimensions from 5:00:00 to 18:00:00, before and after the port scan's mitigation.

As observed, the directed drop policy was able to drop specifically the packets from the attacker, bringing the network to a normal state. As observed on the DDoS attacks, MLP-DS and WaveDetect methods achieved similar results, with just a few anomalous intervals observed in red lines (time interval when the port scan was detected). PSO-DS also achieved good mitigation outcomes for port scan attacks, although the false positive alarms generated may trigger unnecessary mitigation over legitimate users. The tests performed in this paper highlight the importance of the accuracy of anomaly detection methods. The more accurate the detection, the better the mitigation process.

VI. CONCLUSION

In this paper, we present an online defense system for SDN network environments against DDoS and Port Scans attacks. This system is able to analyze the SDN behavior in time intervals of five seconds, and is divided into three main modules, the detection, the identification and the mitigation modules. The first one is responsible for detecting abnormal SDN traffic behaviors, the second provides the system with relevant information about the anomaly and the third one mitigates its impact over legitimate users.

We present three methods for operating on the Detection module: MLP-DS, PSO-DS and WaveDetect. The first one is a supervised machine learning method which requires previous training data to operate, while the two others are unsupervised approaches of online detection. On the other hand, MLP-DS is able to directly identify the detected anomaly, while PSO-DS and WaveDetect need the Identification module to trigger the correct mitigation approach. We tested these methods against each other,

as well as with classic anomaly detection approaches, such as K-means, K-NN and SVM. As the DDoS and port scan detection outcomes fared worse for K-means and K-NN, the PSO-DS, ML-DS, WaveDetect and SVM methods achieved similar results. However, the SVM approach have the burden of a supervised learning without the benefit of directly identifying the detected anomaly, which makes the MLP-DS a more efficient method to be applied when past data is available for training. When there is no training dataset, PSO-DS and WaveDetect can be applied. Between this two methods, WaveDetect achieved better detection outcomes.

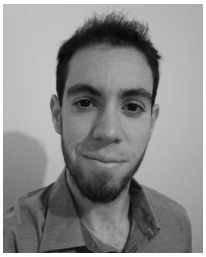
Furthermore, we analyze the Mitigation module outcomes for both DDoS and port scan attacks. For the DDoS attacks, a Game Theoretical approach were used directly into the controller to optimize the packet drop rate to minimize the impact of the attack over legitimate users. For the port scans, a directed drop policy (single source IP drop) was applied since the Identification module of the presented SDN defense system was able to detect the source IP of the attacker. The results show that the mitigation approaches were efficient on bringing back the SDN to its normal operation.

For future works, we intend to analyze the behavior of different anomalies into SDN environments, such as flash crowds and worms. Furthermore, we intend to improve the mitigation approach aiming to reduce even more the impact suffered by legitimate users on the process.

REFERENCES

[1] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.
[2] A. Hakiri, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, "Software-defined networking: Challenges and research opportunities for future Internet," *Comput. Netw.*, vol. 75, pp. 453–471, Dec. 2014.
[3] K. Kalkan, G. Gur, and F. Alagoz, "Defense mechanisms against DDoS attacks in SDN environment," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 175–179, Sep. 2017.
[4] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 27–51, 1st Quart., 2014.
[5] M. Jonker, A. King, J. Krupp, C. Rossow, A. Sperotto, and A. Dainotti, "Millions of targets under attack: A macroscopic characterization of the dos ecosystem," in *Proc. ACM Internet Meas. Conf. (IMC)*, New York, NY, USA, 2017, pp. 100–113.
[6] L. F. Carvalho, S. Barbon, Jr., L. de Souza Mendes, and M. L. Proença, Jr., "Unsupervised learning clustering and self-organized agents applied to help network management," *Expert Syst. Appl.*, vol. 54, pp. 29–47, Jul. 2016.
[7] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, and M. L. Proença, Jr., "Network anomaly detection system using genetic algorithm and fuzzy logic," *Expert Syst. Appl.*, vol. 92, pp. 390–402, Feb. 2018.
[8] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.
[9] M. Ahmed, A. N. Mahmood, and M. Islam, "A survey of anomaly detection techniques in financial domain," *Future Gener. Comput. Syst.*, vol. 55, pp. 278–288, Feb. 2016.
[10] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 4, pp. 217–228, 2005.
[11] S. Chang, X. Qiu, Z. Gao, F. Qi, and K. Liu, "A flow-based anomaly detection method using entropy and multiple traffic features," in *Proc. 3rd IEEE Int. Conf. Broadband Netw. Multimedia Technol. (IC-BNMT)*, Oct. 2010, pp. 223–227.
[12] M. V. O. De Assis, A. H. Hamamoto, T. Abrão, and M. L. Proença, Jr., "A game theoretical based system using holt-winters and genetic algorithm with fuzzy logic for DoS/DDoS mitigation on SDN networks," *IEEE Access*, vol. 5, pp. 9485–9496, 2017.
[13] J. H. Cox *et al.*, "Advancing software-defined networks: A survey," *IEEE Access*, vol. 5, pp. 25487–25526, 2017.
[14] M. Paliwal, D. Shrimankar, and O. Tembhurne, "Controllers in SDN: A review report," *IEEE Access*, vol. 6, pp. 36256–36270, 2018.
[15] Q.-Y. Zhang, X.-W. Wang, M. Huang, K.-Q. Li, and S. K. Das, "Software defined networking meets information centric networking: A survey," *IEEE Access*, vol. 6, pp. 39547–39563, 2018.
[16] T. Xu, D. Gao, P. Dong, H. Zhang, C. H. Foh, and H.-C. Chao, "Defending against new-flow attack in SDN-based Internet of Things," *IEEE Access*, vol. 5, pp. 3431–3443, 2017.
[17] P. Zhang and S. Sun, "Decentralized network anomaly detection via a riemannian cluster approach," in *Proc. IEEE Global Commun. Conf. GLOBECOM*, Dec. 2017, pp. 1–6.
[18] Y. Yu, L. Guo, Y. Liu, J. Zheng, and Y. Zong, "An efficient SDN-based DDoS attack detection and rapid response platform in vehicular networks," *IEEE Access*, vol. 6, pp. 44570–44579, 2018.
[19] H. Peng, Z. Sun, X. Zhao, S. Tan, and Z. Sun, "A detection method for anomaly flow in software defined network," *IEEE Access*, vol. 6, pp. 27809–27817, 2018.
[20] L. F. Carvalho, T. Abrão, L. de Souza Mendes, and M. L. Proença, Jr., "An ecosystem for anomaly detection and mitigation in software-defined networking," *Expert Syst. Appl.*, vol. 104, pp. 121–133, Aug. 2018.
[21] S. Zhao, M. Chandrashekar, Y. Lee, and D. Medhi, "Real-time network anomaly detection system using machine learning," in *Proc. 11th Int. Conf. Design Reliable Commun. Netw. (DRCN)*, Mar. 2015, pp. 267–270.
[22] Y. Wang, R.-J. Jin, and W.-J. Han, "An anomaly traffic detection method based on the flow template for the controlled network," in *Proc. 15th Int. Conf. Opt. Commun. Netw. (ICOCN)*, Sep. 2016, pp. 1–3.
[23] A. Karami and M. Guerrero-Zapata, "A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks," *Neurocomputing*, vol. 149, pp. 1253–1269, Feb. 2015.
[24] M. F. Lima, L. D. H. Sampaio, B. B. Zarpelao, J. J. P. C. Rodrigues, T. Abrao, and M. L. Proença, Jr., "Networking anomaly detection using DSNs and particle swarm optimization with re-clustering," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–6.
[25] A. A. Aburomman and M. B. I. Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," *Appl. Soft Comput.*, vol. 38, pp. 360–372, Jan. 2016.
[26] K. J. Singh and T. De, "MLP-GA based algorithm to detect application layer DDoS attack," *J. Inf. Secur. Appl.*, vol. 36, pp. 145–153, Oct. 2017.
[27] C. Siaterlis and B. Maglaris, "Detecting DDoS attacks using a multilayer Perceptron classifier," in *Proc. 20th Int. Conf. Artif. Neural Netw., III*, Mar. 2004, pp. 118–123.
[28] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan, and M. Sheikhan, "Flow-based anomaly detection using neural network optimized with GSA algorithm," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst. Workshops*, Jul. 2013, pp. 76–81.
[29] A. Y. Nikravesh, S. A. Ajila, C.-H. Lung, and W. Ding, "Mobile network traffic prediction using MLP, MLPWD, and SVM," in *Proc. IEEE Int. Congr. Big Data (BigData Congr.)*, Jun./Jul. 2016, pp. 402–409.
[30] H. Tian and M. Ding, "Diffusion wavelet-based anomaly detection in networks," in *Proc. 17th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2016, pp. 382–386.
[31] S. Kanarachos, J. Mathew, A. Chroneos, and M. Fitzpatrick, "Anomaly detection in time series data using a combination of wavelets, neural networks and Hilbert transform," in *Proc. 6th Int. Conf. Inf., Intell., Syst. Appl. (IISA)*, Jul. 2015, pp. 1–6.
[32] J. Gao, G. Hu, X. Yao, and R. K. C. Chang, "Anomaly detection of network traffic based on wavelet packet," in *Proc. Asia–Pacific Conf. Commun.*, Aug./Sep. 2006, pp. 1–5.
[33] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, 2001.
[34] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, Australia, vol. 4, Nov./Dec. 1995, pp. 1942–1948.
[35] R. Abdel-Kader, M. El-Tarabily, M. Marie, and G. Abdel-Azeem, "A PSO-based subtractive data clustering algorithm," *Int. J. Res. Comput. Sci.*, vol. 3, no. 2, pp. 1–9, 2013.
[36] B. G. Amidan, T. A. Ferryman, and S. K. Cooley, "Data outlier detection using the Chebyshev theorem," in *Proc. IEEE Aerosp. Conf.*, Mar. 2005, pp. 3–8, 2005.
[37] C. Taylor and J. Alves-Foss, "An empirical analysis of NATE: Network analysis of anomalous traffic events," in *Proc. Workshop New Secur. Paradigms (NSPW)*, New York, NY, USA, 2002, pp. 18–26.

[38] S. Chang, X. Qiu, Z. Gao, K. Liu, and F. Qi, "A flow-based anomaly detection method using sketch and combinations of traffic features," in *Proc. Int. Conf. Netw. Service Manage.*, Oct. 2010, pp. 302–305.
[39] S. Haykin, *Neural Networks & Learning Machines*. London, U.K.: Pearson, 2011.
[40] G. Münz, S. Li, and G. Carle, "Traffic anomaly detection using k-means clustering," in *Proc. GI/ITG Workshop MMBnet*, 2007, pp. 13–14.
[41] I. Daubechies, "Wavelets: An overview, with recent applications," in *Proc. IEEE Int. Symp. Inf. Theory*, Sep. 1995, p. 5.
[42] S. Jafarpour, G. Polatkan, E. Brevdo, S. Hughes, A. Brasoveanu, and I. Daubechies, "Stylistic analysis of paintings usingwavelets and machine learning," in *Proc. 17th Eur. Signal Process. Conf.*, Aug. 2009, pp. 1220–1224.
[43] W. Lu, M. Tavallaee, and A. A. Ghorbani, "Detecting network anomalies using different wavelet basis functions," in *Proc. 6th Annu. Commun. Netw. Services Res. Conf. (CNSR)*, May 2008, pp. 149–156.
[44] K. Limthong, P. Watanapongse, and F. Kensuke, "A wavelet-based anomaly detection for outbound network traffic," in *Proc. 8th Asia–Pacific Symp. Inf. Telecommun. Technol.*, Jun. 2010, pp. 1–6.
[45] T. Lotze, G. Shmueli, S. Murphy, S. Murphy, and H. Burkom, "A wavelet-based anomaly detector for early detection of disease outbreaks," in *Proc. 23rd Int. Conf. Mach. Learn. Workshop Mach. Learn. Algorithms Surveill. Event Detection*, 2006, pp. 1–6.
[46] D. C. Hoaglin, "John W. Tukey and data analysis," *Statist. Sci.*, vol. 18, no. 3, pp. 311–318, 2003.
[47] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp. (SIS)*, Apr. 2007, pp. 120–127.
[48] A. A. Poli and M. C. Cirillo, "On the use of the normalized mean square error in evaluating dispersion model performance," *Atmos. Environ. A, Gen. Topics*, vol. 27, no. 15, pp. 2427–2434, Oct. 1993.
[49] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.
[50] Mininet Team. (2018). *MiniNet Overview*. [Online]. Available: http://mininet.org/overview/
[51] Philippe Biondi and The Scapy Community. (2018). *Scapy*. [Online]. Available: http://www.secdev.org/projects/scapy/
[52] V. Hautamaki, I. Karkkainen, and P. Franti, "Outlier detection using k-nearest neighbour graph," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, vol. 3, Aug. 2004, pp. 430–433.
[53] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, New York, NY, USA, 2000, pp. 427–438.
[54] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.



MARCOS V. O. DE ASSIS received the master's degree in computer science from the State University of Londrina, Brazil, where he is currently pursuing the Ph.D. degree with the Electrical Engineering Department. He is part of the Computer Networks and Data Communication Research Group. He is currently a Professor with the Engineering and Exact Department, Federal University of Paraná, Brazil. His research interests are in the management and security of large-scale computer networks and software defined networks.

MATHEUS P. NOVAES received the B.S. degree in computer science from the State University of Londrina (UEL), Brazil, in 2017, where he is currently pursuing the M.Sc. degree in computer science. Since 2015, he has been a member of the Computer Networks and Data Communication Research Group, Computer Science Department, UEL. His research focus is on management and security of computer networks.

CINARA B. ZERBINI received the B.S. degree in computer science from the State University of Londrina, Brazil, in 2016, where she is currently pursuing the master's degree with the Computer Department. She is currently a member of the Computer Networks and Data Communication Research Group. Her main interests are signal processing and statistical tools, security of computer networks, and software-defined networking.

LUIZ F. CARVALHO received the master's degree in computer science from the State University of Londrina in 2014 and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas in 2018. He is currently a Lecturer and a member of the Computer Networks and Data Communication Research Group, State University of Londrina. His main research interests are management and security of computer networks and software-defined networking.

TAUFIK ABRÃO (M'97–SM'12) received the B.S., M.Sc., and Ph.D. degrees in electrical engineering from the Polytechnic School, University of São Paulo, São Paulo, Brazil, in 1992, 1996, and 2001, respectively. From 2007 to 2008, he was a Post-Doctoral Researcher with the Department of Signal Theory and Communications, Polytechnic University of Catalonia, Barcelona, Spain. In 2012, he joined the Southampton Wireless Research Group, The University of Southampton, Southampton, U.K., as an Academic Visitor. Since 1997, he has been with the Communications Group, Department of Electrical Engineering, State University of Londrina, Brazil, where he is currently an Associate Professor of telecommunications and the Head of the Telecomm and Signal Processing Laboratory. He has participated in several projects funded by government agencies and industrial companies. He has supervised 21 M.Sc., six Ph.D., and three post-doctoral students. He has co-authored 10 book chapters on mobile radio communications and over 180 research papers published in specialized/international journals and conferences. His current research interests include communications and signal processing, especially massive multi-in multi-out, and OFDM/OFDMA systems, detection and estimation methods, cooperative communication and relaying, resource allocation, and heuristic and convex optimization aspects of 4G and 5G wireless communication systems. He served as a TPC Member for several symposiums and conferences. He is a Senior Member of the Brazilian Telecommunication Society. He is involved in editorial board activities of six journals in the telecommunications area. He has been serving as an Editor for the IEEE Communications Surveys and Tutorials since 2013, the IEEE Access since 2016, and the *IET Journal of Engineering* since 2014.

MARIO L. PROENÇA Jr. received the M.Sc. degree in computer science from the Informatics Institute, Federal University of Rio Grande do Sul, in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas in 2005. He is a Master's Supervisor of computer science with the State University of Londrina and a Ph.D. Supervisor with the Department of Electrical Engineering, State University of Londrina (UEL), Brazil. He is currently an Associate Professor and a Leader of the research group that studies computer networks with the Computer Science Department, UEL. He has authored or co-authored over 100 papers in refereed international journals and conferences and books chapters, and holds one software register patent. He has supervised 12 M.Sc. and two Ph.D. students. His research interests include computer network, network operations, management and security, and IT governance.

APÊNDICE C - *Near Real-time Defense System Applied to SDN Environments in IoT Networks using Convolutional Neural Network*

Computers and Electrical Engineering 86 (2020) 106738

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Near real-time security system applied to SDN environments in IoT networks using convolutional neural network☆

Marcos V.O. de Assis[a], Luiz F. Carvalho[b], Joel J.P.C. Rodrigues[c,d], Jaime Lloret[e,*], Mario L. Proença Jr[f]

[a] Engineering and Exact Department, Federal University of Paraná (UFPR), Paraná, Brazil
[b] Federal University of Technology - Paraná (UTFPR), Apucarana, Brazil
[c] Federal University of Piauí, Teresina - PI, Brazil
[d] Instituto de Telecomunicações, Portugal
[e] Integrated Management Coastal Research Institute, Universitat Politecnica de Valencia, Valencia, Spain
[f] Computer Science Department, State University of Londrina (UEL), Paraná, 86057-970, Brazil

ARTICLE INFO

Article history:
Received 3 November 2019
Revised 22 June 2020
Accepted 23 June 2020
Available online xxx

Keywords:
Software-defined Network
Internet of Things
DDoS
CNN
Botnet
Deep Learning

ABSTRACT

The Internet of Things (IoT) paradigm brings new and promising possibilities for services and products. The heterogeneity of IoT devices highlights the inefficiency of traditional networks' structures to support their specific requirements due to their lack of flexibility. Thus, Software-defined Networking (SDN) is commonly associated with IoT since this architecture provides a more flexible and manageable network environment. As shown by recent events, IoT devices may be used for large scale Distributed Denial of Service (DDoS) attacks due to their lack of security. This kind of attack is commonly detected and mitigated at the destination-end network but, due to the massive volume of information that IoT botnets generate, this approach is becoming impracticable. We propose in this paper a near real-time SDN security system that both prevents DDoS attacks on the source-end network and protects the sources SDN controller against traffic impairment. For this, we apply and test a Convolutional Neural Network (CNN) for DDoS detection, and describe how the system could mitigate the detected attacks. The performance outcomes were performed in two test scenarios, and the results pointed out that the proposed SDN security system is promising against next-generation DDoS attacks.

1. Introduction

Internet of Things (IoT) is a paradigm defined by Tudosa et al. [1] as an evolving technology where every device can be both connected through a network and controllable from a remote station. It envisions a world where a significant amount of everyday objects communicate through wired and wireless networks [2]. The amount of information traveling over the Internet has been growing exponentially in past years, mainly due to the popularization of cloud computing solutions and connected applications, such as video conferences, IP video surveillance systems and so on. According to Chaabouni et al.

☆ This paper is for regular issues of CAEE. Reviews processed and recommended for publication to the Editor-in-Chief by Area Editor Dr. Huimin Lu.

* Corresponding author.
E-mail addresses: marcos.assis@ufpr.br (M.V.O. de Assis), luizfcarvalho@utfpr.edu.br (L.F. Carvalho), joeljr@ieee.org (J.J.P.C. Rodrigues), jlloret@dcom.upv.es (J. Lloret), proenca@uel.br (M.L. Proença Jr).

https://doi.org/10.1016/j.compeleceng.2020.106738
0045-7906/

[3], the IoT market is rapidly growing, starting with 2 billion devices by 2006 to a projection of 200 billion by 2020 (a 200% rise). With recent advances in communication technologies, the IoT is gaining visibility among researchers [1,4].

One of the main issues relating to IoT networks is its heterogeneous characteristic, since different applications have specific network requirements to optimize the system's operation. In [5] the authors highlight some examples of this IoT characteristic: in smart vehicles applications, the information exchange would require almost zero latency; in industrial sensor networks, besides the low latency, a minimal packet loss would be required; in mobile video surveillance network, the latency and packet loss are not critical, but would require a higher bandwidth. According to Caraguay et al. [6], these specific network requirements are incompatible with the traditional networking model, which have limitations regarding scalability, mobility and amount of traffic. Thus, traditional networks are inefficient to satisfy the new requirements of IoT environments.

A new paradigm that aims to provide scalability and flexibility to network's management process is the Software-defined networking (SDN). SDN enables centralized network management, allowing efficient configuration and optimization by transforming the traditional "black-box" network components into "white-box" software-controlled ones. This abstraction is possible by decoupling the control and data planes, where all control functions are implemented in a programmable central controller. This controller, in turn, sends packet forwarding and management policies to SDN controlled switches and routers, dynamically coordinating their operation and, consequently, the network behavior. These features make SDN a promising environment for the development and operation of IoT solutions [7].

Besides the advantages of the services provided by IoT, we recently witnessed its side effects relating to network security. According to Kim et al. [4], IoT devices may be susceptible to malware infections, which stealthily propagates between unsecured devices to create massive IoT botnets. The cause of this infection is mainly due to management vulnerabilities. According to CISCO [8], in 2018 about 83% of network devices in their partner sample were running with known vulnerabilities, and IoT devices are implemented without any security planning. These IoT botnets, in turn, are able to execute powerful Distributed Denial of Service (DDoS) attacks. Recently, IoT devices were used on a DDoS attack against the servers of Dyn Inc., a company that controls much of the Internet's DNS infrastructure [9]. This attack is considered to be one of the largest of its kind with a 1.2 Tbps rate.

The traditional DDoS protection approach is the detection and mitigation of the attack at the victim's server or network [10,11], which are able to detect the anomalous traffic pattern and apply mitigation policies. However, as previously discussed, DDoS attacks are becoming more and more powerful, and the attack volume can be larger than the security system is able to handle. A solution for this scenario is proposed by Mirkovic et al. [12], where a system called D-WARD is proposed to deal with DDoS attacks at the source-end network (stub networks or ISP networks). The idea behind this solution is to "divide and conquer", *i.e.*, each ISP network avoids the attack proceeding to the Internet, mitigating the DDoS impact before it reaches its target. As highlighted by the authors, the major challenge of this approach is incentive, since it directly benefits the victims instead of the deploying ISP, for instance.

However, DDoS attacks may also impair the operation of SDN environments since the traffic is managed by a central controller [13,14]. In addition, DDoS attacks may be performed by IoT devices. Finally, SDN is a viable environment to enable the operation of IoT solutions with customized network requirements. We believe that these statements are a great incentive for ISP network to deploy DDoS security systems on source-end networks, targeting the protection of its SDN controller and indirectly mitigating the attack over the Internet.

Thus, we propose a near real-time security system applied on SDN environments to mitigate DDoS attacks originated from inner devices, such as IoT botnets. The proposed system protects the SDN's central controller against flooding and prevents the attack from leaving the source-end network, indirectly protecting the victim's server. The system is divided into two sections, the Detection Module, responsible for detecting and identifying the attack occurrence, and the Mitigation Module, responsible for selecting drop policies to secure the SDN controller.

On the Detection module, we applied a deep learning method using a multidimensional IP flow analysis, called Convolutional Neural Network (CNN) [15]. This method is widely applied on image recognition/classification problems, and provides the system the ability to learn local patterns along the data set.

Although the proposal of a mitigation approach is not in the scope of this paper, we describe how a mitigation approach operates within the presented system.

To measure the efficiency of the presented CNN method on the Detection module, we used two test scenarios. On the first one, we applied SDN data, simulated through the usage of the network emulator Mininet, OpenFlow (IP flow data) and controlled by Floodlight (SDN controller) of different architectures and DDoS intensities. On the second scenario, we tested the CNN on the public DDoS database CicDDoS 2019 [16], since benchmark data sets are a good basis to evaluate and compare the quality of different network anomaly detection methods. Different methods are compared to the CNN on both scenarios for results comparison and data analysis.

The fundamental commitments of this paper are:

- A security system for SDN environments against inward DDoS attacks;
- The system indirectly protects victims' servers by mitigating the DDoS at the source-end network;
- The efficiency evaluation and comparison of distinct fast DDoS detection techniques applied on SDNs.

The remainder of this paper is organized as follows: Section 2 presents a study of related works; Section 3 describes the organization of the proposed system; Section 4 details the CNN method used for anomaly detection on the system's

Detection Module; Section 5 discusses the performance results achieved; Finally, Section 6 presents the conclusions and future works.

2. Related works

Distributed Denial of Service (DDoS) attack is a critical issue in network security that costs organizations and individuals a great deal of time, money, and reputation. Based on this assertion/concern, various techniques for detecting DDoS attacks and reducing its effects in different network environments have been proposed [17]. In [18], the authors used an Artificial Neural Network (ANN) to detect DDoS attacks classifying the traffic into anomalous and genuine. The implemented ANN was trained with old and up-to-date data sets, and it successfully obtained a high detection rate of both known and unknown attacks. Their solution was based on specific feature patterns as the source and destination addresses and ports. Their approach could not handle DDoS attacks where packet headers are encrypted.

Similarly, in [19] the authors detected application-layer DDoS attacks using a Multi-Layered Perceptron (MLP) classification algorithm. The proposed MLP used Genetic Algorithm (GA) as a learning algorithm. A data set generated by a real attack was used during the tests. The findings indicated that important characteristics for attack identification include the number of HTTP GET requests for a particular address in a 20s long time slot, the entropy of the requests, and variance of the entropy. The results showed that the MLP achieved high rates for accuracy and sensitivity.

Wang et al. [20] proposed an approach where raw IP flow data are represented as an image and used Convolutional Neural Network (CNN) to classify and identify malicious traffic. The authors achieved good outcomes on the detection of different malicious events. This representation is a promising approach using CNN. However, by submitting raw flow data on the training process, the method may learn that specific IP addresses are related to malicious behavior.

Liu et al. [21] propose two payload classification approaches based on Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), respectively. These approaches are used for attack detection, and the authors highlight that their ability to learn feature representations without feature engineering from the original data. The authors compare the proposed methods with different approaches, and achieved good results, with accuracy rates higher than 99% in tests using DARPA1998 data set.

Despite such a high detection rate, mitigation of DDoS attack was not discussed in the earlier presented studies. Currently, software-defined networking offers network programmability, making this communication paradigm a trend for traffic controlling [22] and, consequently, contributing to containing this attack. In [23], the authors presented a DDoS-type attack detection and mitigation system suitable for cloud computing environments using SDN structure. Named DaMask, this system is divided into two modules. The first is responsible for detecting the attack, performing statistical analysis. It uses a graphical structure to store the known traffic patterns and, when unusual behavior is observed, this graph determines if malicious connections are occurring. The second module, specifically targeted the dynamic network environment, performing countermeasures, and generating logs about the detected attacks. With a similar purpose, in the mechanism proposed by Cui et al. [24], once a DDoS attack is detected, the anomaly attenuation occurs by inserting entries in the switch table of the first recognized switch in the propagation path of the anomaly. These entries have rules that discard packets whose address and destination port match the attack target.

Joldzic et al. [25] proposed a three layers defense for SDN topology. The outmost is located at the network gateway. This layer has an OpenFlow switch to slice and deliver the resulting chunks of incoming traffic to the subsequent layer. The second layer has several devices called processors, responsible for traffic analysis and anomaly detection. The last layer contains an OpenFlow that aggregates the traffic forwarded by the processors and transmits it to the inside of the network. This approach presents two disadvantages. First, it is assumed that the computer network is free of internal anomalies. In this case, attacks can be launched by hosts inside the network and overwhelm the SDN controller. Second, splitting the traffic into different processors can also lead to the division of the attack between them. This may hinder the search for anomalies since such a division may accidentally mask the attack. Also, to mitigate DDoS attacks, Chen et al. [26] used specialized software boxes to protect the control plane from overloading during the attack by enhancing the ingress switches scalability.

Several works employed push-back schemes to mitigate denial of service attacks [24]. When an attack is detected, the push-back strategy eliminates the attack traffic and notifies other forwarding devices about such traffic. According to Hameed and Ahmed Khan [27], the push-back scheme imposes complexity and overhead in the network management because all forwarding devices in the attack path must be coordinated. To overcome this drawback, they proposed a collaborative DDoS mitigation scheme leveraging SDN. The authors deployed a secure controller-to-controller communication protocol lying in different autonomous systems to transmit attack information with each other. The authors designed a testbed to test three deployment approaches (linear, centralized, and mesh) for policy distribution to other autonomous systems. Experiments showed that mitigation was transferred from destination to source, saving valuable time, and network resources.

In addition to the traditional push-back mitigation scheme, DDoS security solutions were used at the victim-end because of the ease of deployment and availability of complete attack information. Behal et al. [28] argue the lack of sufficient computational resources at the victim-end along with the massive network traffic volume generated by DDoS attacks make security solution itself vulnerable. The authors proposed an ISP level distributed defense system to divide the computational complexity among the nearest point of presence (PoP) routers. Traffic is monitored at all the ingress points of an ISP and sent to a central coordinator in the victim's network.

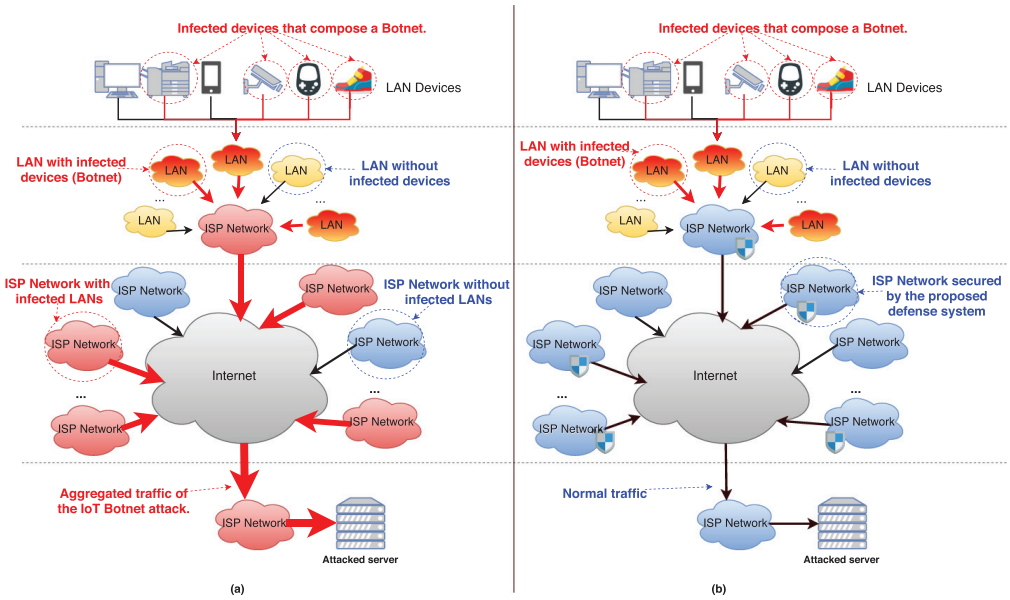


Fig. 1. (a) Representation of a DDoS attack performed by an IoT botnet using different ISP networks. (b) Representation of the previous scenario with DDoS mitigation at the SDN controller of each ISP network.

In this paper, we propose a security system able to protect the SDN controller against internal DDoS attacks targeting an external server. Differently from previous works, our proposal does not require the network infrastructure to be modified. Also, it eliminates the use of push-back to perform mitigation, thereby reducing the complexity of network management. By mitigating the attack at the source-end networks, the overall DDoS attack should be mitigated on the destination-end network, protecting both the targeted server and the SDN controller.

3. Proposed security system

In this section, we depict the functioning of the proposed SDN security system. It is designed to operate within the SDN central controller of ISP networks, helping to protect it against DDoS attacks. Furthermore, by preventing the DDoS attack to proceed to the Internet, the proposed system parallely mitigates DDoS attacks of external targets. This occurs through a "divide and conquer" approach, *i.e.*, the DDoS attack is mitigated inside the source-end network of several different ISP networks. Fig. 1 summarizes this idea.

In Fig. 1(a) represents a DDoS attack without the proposed protection, and (b) represents the attack after the mitigation process. The Internet interconnects several different ISP networks, which, in its turn, connects several LANs composed by heterogeneous devices. With the popularization of IoT solutions, these LANs tend to increase in size and traffic, which consequently make them powerful sources for DDoS attacks since IoT devices may be susceptible to malware infections [4].

These attacks lie upon their distributed architecture to impair the operation of the target's server. As shown in Fig. 1(a), different infected devices within several ISP LANs may target a single server, and the traffic aggregation between this attack and several others from different ISP networks provides a massive resource depletion on the destination-end network. Depending on the amount of infected devices inside the ISP network, this situation may as well impair the operation of its SDN central controller and, consequently, the quality of its provided services to final users.

By preventing these malicious packets from passing through the SDN controller, the DDoS attack is mitigated before it reaches the target's network, as shown in Fig. 1(b). Furthermore, through the usage of dropping policies, malicious traffic will not impair the SDN controller, guaranteeing the ISP network to operate in its normal state.

To provide this protection, the proposed system is based on the analysis of IP flow dimensions, using distinct features to recognize a pattern relating the network's normal operation and to detect the existence of DDoS attacks.

To reduce the DDoS impact over legitimate users, the proposed system operates in near real-time, extracting and analysing IP flow data in one-second intervals. This time interval analysis enables fast detection and mitigation, reducing the damage over both the SDN controller (and consequently its users) and the external attacked server.

It is important to highlight that, in order to enable the speed of the detection and mitigation processes, the system operates autonomously. Thus, even though the system generates an alarm to inform the network administrator when a DDoS is detected, no human interaction is required. A flowchart representing the functioning of the proposed system is described by Fig. 2.

As shown, every second IP flow dimensions or features are exported from the SDN controller through the OpenFlow protocol (6 features in this example). These dimensions are heterogeneous data that can be classified as quantitative (like the rate of packages and bits per second) and qualitative features (like source/destination ports and IP addresses). To enable

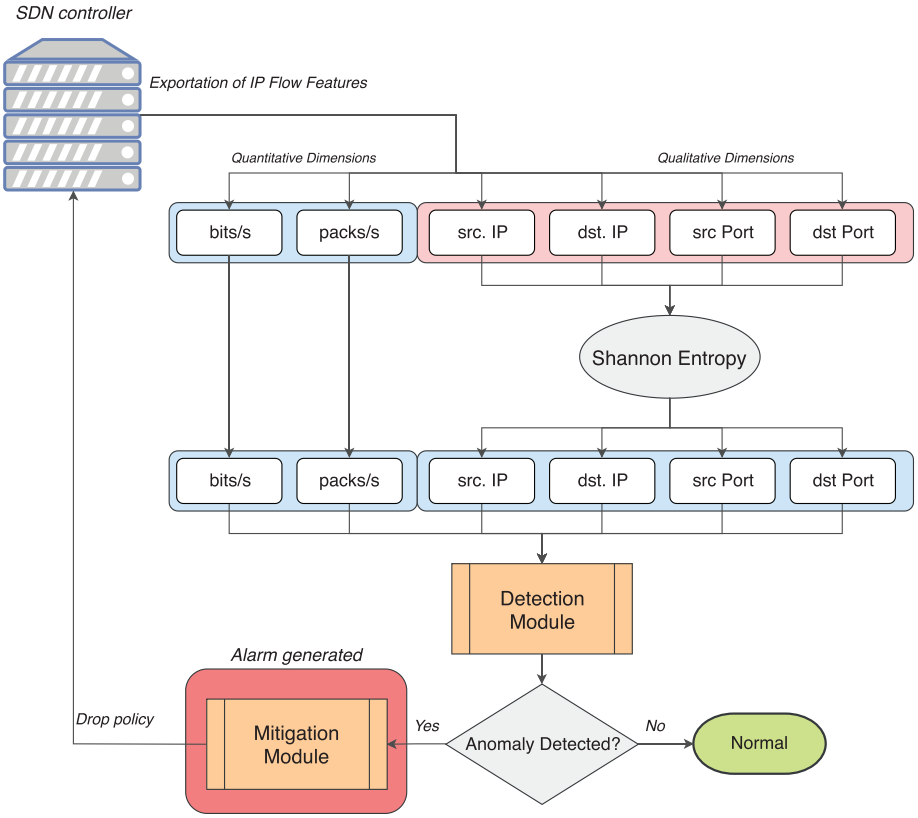


Fig. 2. Overall functioning of the proposed SDN security system.

the usage of this data by the Detection Module, the qualitative dimension must be converted to a quantitative one. Thus, the qualitative dimensions are submitted to the Shannon Entropy, which highlights the concentration or dispersion degree of the analyzed time interval. Given a dimension $X = \{x_1, x_2, ..., xn\}$ in which x_i is the occurrence frequency of the sample i at a given time interval, the Shannon entropy (H) for the dimension X is calculated through:

$$H(X) = -\sum_{i=1}^{N}\left(\frac{x_i}{S}\right)log_2\left(\frac{x_i}{S}\right), \tag{1}$$

where S is the sum of all the occurrence frequencies of the elements present on the analyzed time interval $\left(S = \sum_{i=1}^{N} x_i\right)$.

Thus, if there is a high concentration of a specific destination IP Address, this dimension will be represented with a low entropy value. On the other hand, if there is a high dispersion in the data relating to the same dimension, a higher entropy value will be retrieved.

After this step, the quantitative data relating to the analysed dimensions are submitted to the Detection Module, responsible for analyzing and detecting the occurrence of DDoS attacks. If a DDoS is detected, then the Mitigation Module is triggered, generating a countermeasure policy that must be incorporated by the SDN controller.

As observed by Fig. 2, the proposed system is composed of modules, and their relation is presented by Fig. 3.

As seen, the proposed security system is separated into two main sections: the Detection and the Mitigation modules, which are composed by complementary sub-modules that contributes to their operation.

As the name suggests, the Detection Module has the objective of detecting and identifying DDoS attacks. In this module, we applied the Convolutional Neural Network approach and compared it with other methods for a performance comparison, which is presented in Section 5. As shown by Fig. 3, the Detection Module is divided into two sub-modules: the "training process" and the "anomaly detection". As CNN is a supervised learning approach, it requires a prior step of training that will calibrate its classification outcomes. This process will be detailed in Section 4. The Anomaly Detection is the second sub-module, which is in charge of detecting DDoS events on the network over the analyzed IP flow features, as previously described.

The Mitigation Module is triggered when a DDoS event is identified by the Detection Module, being responsible for the decision-making process that will provide the optimal countermeasures for the attack.

As the proposal of a novel mitigation scheme is not on the scope of this paper, we introduce here an approach presented in [29] to illustrate this module's operation.

Thus, two other sub-modules compose the Mitigation one, the "Game Theoretical countermeasure approach" and the "countermeasure execution". On the first sub-module, to calculate the optimal countermeasure policy for DDoS mitigation, a

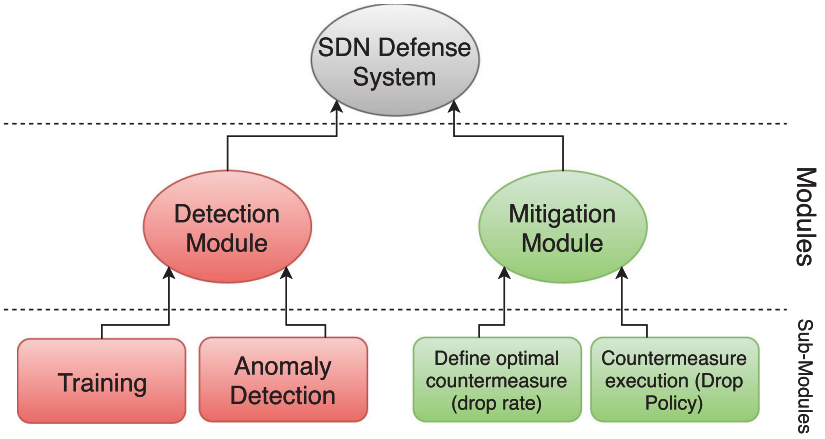


Fig. 3. Modular organization of the SDN security system.

game theory (GT) based approach, described in [29], could be applied, where the process is implemented at an SDN border gateway device and used on the mitigation of DDoS attacks, protecting the network's central controller against internal and external attacks. As previously discussed, these attacks may not be targeting the operation of the SDN itself but, as the communication traffic passes through its central controller, it may be also impaired. The output of this sub-module is an optimal packet drop rate.

The GT approach is a game of two players, the attacking user and the security system. As the DDoS traffic passes through the SDN controller before reaching its real target, for simplicity, we consider the attacker to be targeting the SDN controller. Thus, the attacking user aims to maximize the DDoS impact on the SDN controller while limiting its possibility of being identified. The security system, in turn, tries to limit the impact generated by the attacking user to guarantee the normal operation of the services made available by the SDN (ISP). Such games are classified as "zero-sum games" since the loss of one player is the gain of another, and this gain is commonly defined as payoff. Different metrics are used for the payoff calculation, taking into account i) the error between the analyzed time interval and the expected SDN behavior, ii) the cost of the attack for the attacker player, iii) the consumption of bandwidth by legitimate users compared to attacking ones on the average, iv) and legitimate users' estimated packet drop after the mitigation process [29].

The variables of these metrics are stated through a set of feasible moves executed by the players. The security system is able to:

- Discard packets to avoid their processing;
- Authorize packets to be processed by the central controller.

The attacking user, in turn, is able to:

- Change the attack intensity (amount of packets/s transmitted by each attacking host);
- Change the amount of attacking hosts.

Finally, the second Mitigation sub-module, the "countermeasure execution", provides the SDN controller with the optimal packet dropping policy achieved by the GT approach. In short, the first mitigation sub-module estimates the optimal drop policy, and the second one sends it to the SDN central controller for operation.

4. Anomaly detection approach

We describe in this section the anomaly detection approach applied on the proposed System's Detection Module. A performance comparison between the presented method and others is available on Section 5.

4.1. Convolutional neural network (CNN)

Among the different approaches applied to the detection of computer network attacks and anomalies, deep learning methods are becoming increasingly popular among researchers. Deep learning methods are a subclass of machine learning that is capable of extracting patterns in complex data. Thus, it is widely applied to image recognition and pattern classification problems. As stated by Chollet [15], the "deep" in deep learning stands for the idea of successive layers of representations, as the number of layers representing a method is known as its depth. Deep learning methods commonly use three or more layers of representation, while "shallow" learning methods, like Multi-Layered Perceptron (MLP), focus on learning through only one or two layers.

In [21], the authors highlight that the main benefit of deep learning methods is the absence of manual feature engineering. In other words, the technique is capable of finding patterns among massive data sets during the training process

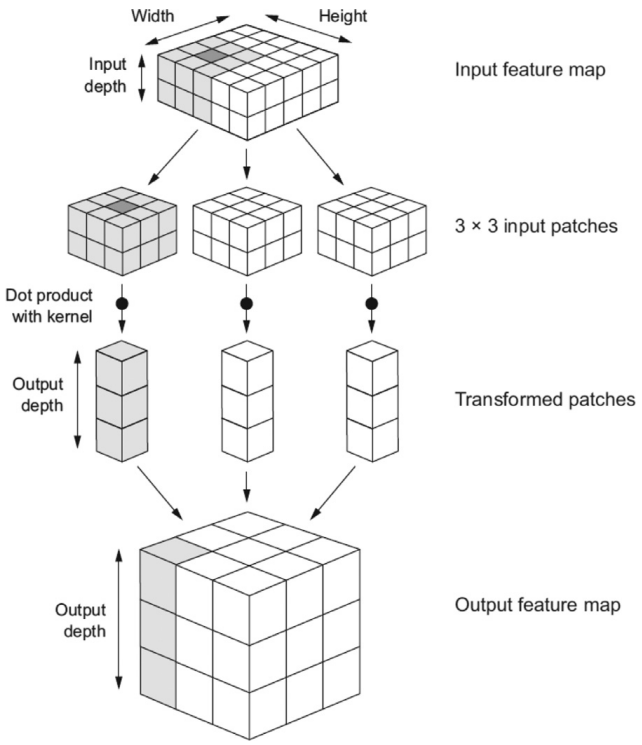


Fig. 4. Operation of the convolution process. [15]

by itself, giving more "importance" to features that are more relevant to the classification process. This characteristic dramatically increases the classification outcomes, since complex patterns, sometimes stealth for human eyes, can be extracted from the data set.

In this paper, we apply a deep learning method known as Convolutional Neural Network (CNN) on the detection of DDoS attacks. As described by Chollet [15], the fundamental difference between a fully connected layer (used by MLPs, for instance) and a convolutional layer is that the first learns global patterns in their input feature space, while the second is capable of learning local patterns. As CNN's are commonly applied to image processing environments, it can extract local patterns on the image, which significantly improves the accuracy of classification problems.

This precision is possible through the convolutional operations that compose CNN. A convolution is an operation between two functions that produces a third one, which expresses how the shape of one is modified by the other. As described by Chollet [15], convolutions operate over 3D tensors called *feature maps*. On image classification problems, for instance, they stand for two spatial axes (width and height) and a *channel* axis (for RGB images, the channel is 3, wherein black-and-white images, it is 1). To convolve this input, a *filter* is applied through dot products to extract local patterns. The filter operates like a sliding window, performing a dot product with all the unique positions where it can be put on the image, encoding specific characteristics of the input. In other words, a convolution works by sliding these filters of fixed size over the 3D input feature map, stopping at every possible location, and extracting the 3D patches, which are processed via dot products into 1D dimension outputs. These outputs, in turn, are reassembled into a 3D output map, as described in Fig. 4.

On CNN networks, Convolutional layers are commonly followed by Pooling layers, whose objectives are to reduce the spatial size of the representation. Thus, the pooling process reduces the number of parameters and computation in the CNN, *i.e.*, downsample feature maps. The most common approach used in this layer is the Max pooling, which consists of extracting windows from the input feature maps and outputting the max value of each channel [15], due to its efficiency.

However, IP flow traffic data are represented as a time series, not an image. Thus, a variation of the traditional 2D convolutional operation is used in this paper, the 1D-CNN [15]. In a straightforward comparison, it operates with the same structures and functions, but through 1-dimensional data (time series), like show by Fig. 5.

1D-Convolutional layers also receive 3D tensors as input: the first one representing the number of samples, the second standing for the time, and the third for the features [15]. In this paper, as the system is operating with one-second data, the input tensor is configured as (samples, features, channels). The architecture of the CNN implemented in this paper is described by Fig. 6.

As observed, the architecture of the CNN is composed of a stack of two *Conv1D* and *MaxPooling1D* layers. They are followed by a *Flatten* layer, responsible for transforming the 3D output of the previous layers into 2D inputs for the following layers, a *Dropout* layer, aiming to avoid over-fitting as CNN's tend to converge very fast to a solution, and a *Dense* or *Fully-Connected* layer, to perform a global model classification. The output is a single neuron with a sigmoid activation function for binary classification, *i.e.*, which classifies data as normal or DDoS.

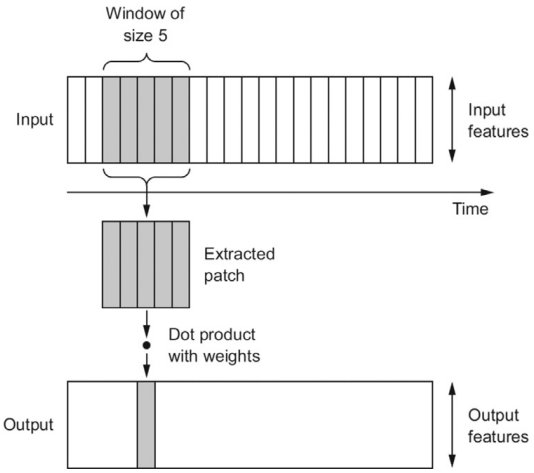


Fig. 5. Convolution in one dimension. [15]

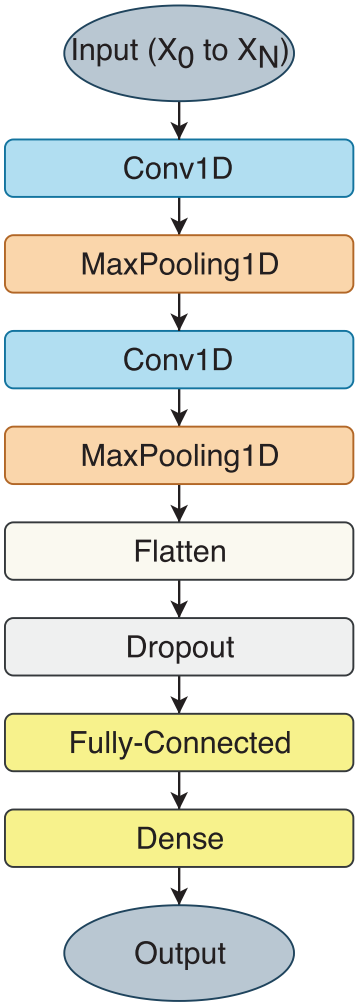


Fig. 6. CNN architecture.

5. Performance outcomes

We examine in this section the performance results relating to the proposed detection module for the security system. For this, we tested the system over two different scenarios and compared the CNN approach with different methods for performance evaluation. These methods are: the Multi-Layered Perceptron (MLP) Network, a machine learning method with one hidden layer composed of 10 neurons; the Deep Neural Network (DNN) or Dense MLP (D-MLP) [15,30], the deep learning version of the MLP, containing 3 hidden layers with 10 neurons on each one; and the Logistic Regression [13], which is a

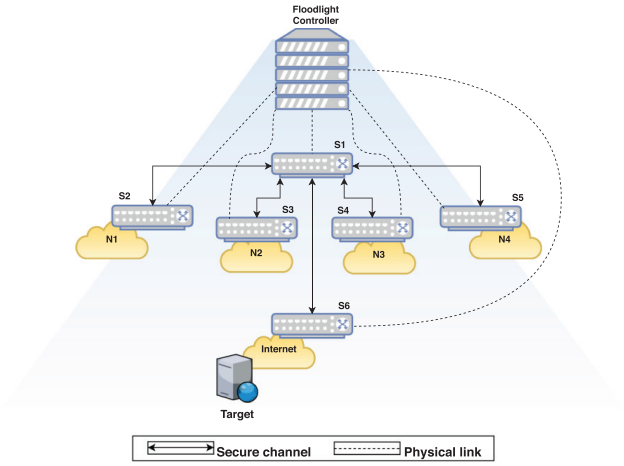


Fig. 7. Simulated SDN topology.

Table 1
Summary of training (Day 1) and test days (Days 2 to 7) on the first scenario.

Day 4	Day 5	Day 6	Day 7				
Switches	6	6	6	6	6	6	6
Hosts	120	200	200	150	150	150	150
# of DDoS attacks	2	1	1	1	1	1	1 (short)
# of attacking hosts	15	20	20	15	20	10	10

statistical model used to predict values taken by a categorical variable from a series of continuous or binary explanatory variables. All detection methods were implemented using Python and Keras on a computer using Windows 10 64bit, Intel Core i7 2.8GHz, and 8GB of RAM.

5.1. Scenario 1 - SDN simulated data

In this scenario, we applied simulated IP flows, generated through the network emulator Mininet, to perform the tests. Mininet is a lightweight software that enables the generation of realistic emulated SDN environments composed by hosts, links, switches, and controllers. One of its main advantages is the ease of the production of customized network topologies through a virtual machine. To control the simulated network switches, we used the Open vSwitch, which is compatible with the Mininet environment. Together with Mininet, we applied the Floodlight, an SDN controller extensively utilized in literature, where the proposed security system is implemented. The OpenFlow protocol performs data collection.

Six switches compose the SDN environment used for the system's analysis. As one central device interconnects the others, all the five remaining switches control from 24 to 40 hosts, totaling 120 to 200 connected hosts, respectively. One of these switches stands for a boundary gateway device, containing the external (Internet) hosts and the attacked server. Fig. 7 describes the network topology.

Seven days of SDN traffic (168 hours) were generated and used for both training and testing. Each day interval emulates the normal behavior of an ISP network, with higher data traffic on working hours and by the evening, while lower data traffic occurs early in the morning and by dawn.

Two distinct-intensity occurrences of DDoS attacks were injected on the data of the first generated day (Day 1), which was utilized for training and adjustment of the methods' anomaly detection procedures.

The following six generated days (Days 2 to 7) were applied to test the performance of the presented DDoS detection methods. A summary of each one of the training and test days is shown in Table 1. As observed, the testing days have different characteristics in comparison to the training day, all of them with a higher number of hosts, which should make the attacks stealthier, *i.e.*, harder to detect. Days 2 and 3 have an increased number of attacking hosts on different periods of the day. On days 3 and 4, we applied 150 hosts on the network, where 15 and 20 of them are malicious ones, respectively. Finally, Days 6 and 7 use 150 hosts on the architecture, in which 10 of them are malicious nodes. These are stealthier scenarios, in which their only difference is the attack duration: while the attack on Day 6 lasts for 3638 time intervals, the same attack lasts for 618 time intervals on Day 7.

In this first scenario, we exported six IP flow dimensions from the SDN controller, which are: Bits and Packets per second, Source and Destination IP addresses and Ports. With this amount of analyzed features, the number of computed parameters is low. So, we suppressed the first MaxPooling layer of Fig. 6 to avoid data loss. The parameters were set with 16 and 8 filters for the first and second Conv1D layers, respectively, both with kernel (filter) size 3. The second MaxPooling1D was

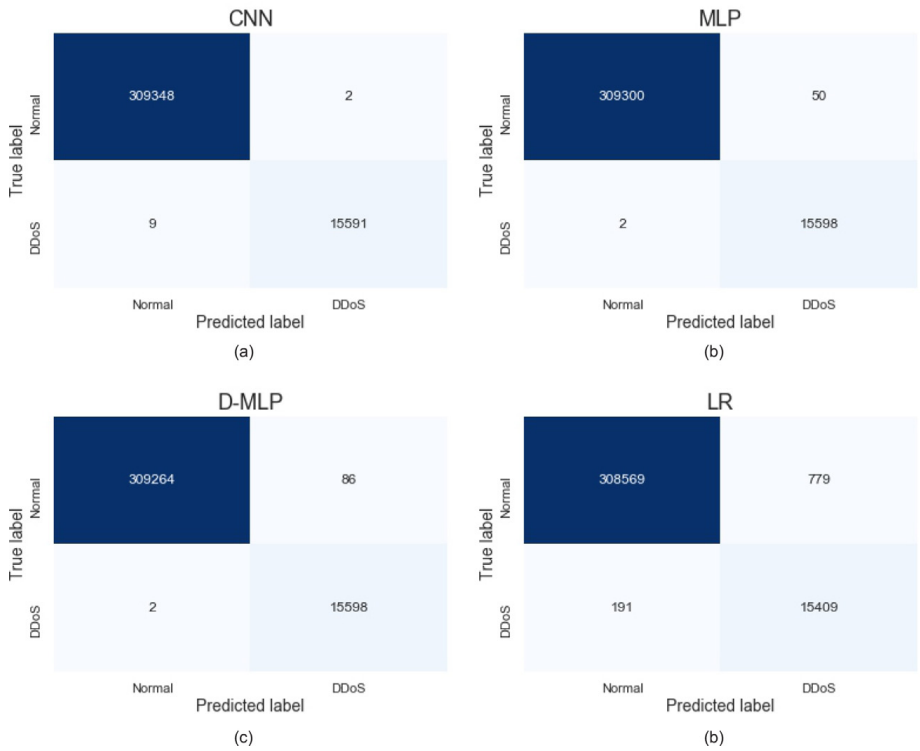


Fig. 8. Confusion Matrices of the tested methods relating the first test scenario.

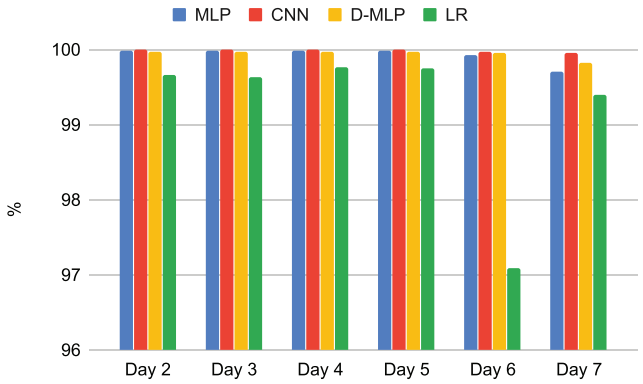


Fig. 9. Methods' accuracy outcomes for the first test scenario.

set with a pool size of 2, and the Dropout with a rate of 0.5. The Fully-connected layer is composed of 10 neurons, and the output layer is formed of 1 neuron to generate a binary result. All methods were tested through 1000 epochs.

We applied classical anomaly detection statistic techniques to measure their efficiency in detecting DDoS attacks targeting an external server. Fig. 8 shows the outcomes achieved by each one of the tested methods through confusion matrices.

As observed, the CNN method fared better, achieving the lowest false-positive (FP) rate, *i.e.*, when benign data are classified as a DDoS attack, followed by MLP, D-MLP, and LR methods. MLP and D-MLP methods achieved lowest false-negative (FN) rates, *i.e.*, when a DDoS interval is classified as normal, but the difference between them and CNN is small.

Other classical metrics used to measure anomaly detection performance are the accuracy, precision, recall and f-measure techniques. Accuracy shows the percentage of time intervals correctly classified. Precision estimates the ratio of intervals correctly recognized as DDoS among all the samples classified as DDoS. The recall metric represents the percentage of correctness for DDoS intervals. Finally, F-measure represents the harmonic mean between recall and precision. The results achieved by the tested methods using these metrics on each day are shown by Figs. 9–12.

In this test scenario, despite the results presented by the confusion matrices, a global analysis of the accuracy rates, presented by Fig. 9, shows that all methods achieved good classification results on the average. The accuracy rates were higher than 99.5% for all tested approaches for all analyzed days, except for the LR method on Day 6. CNN method presented better results in comparison to the other methods, despite the similarity of their outcomes (rates around 99.9% on the average), as the difference between CNN, MLP, and D-MLP accuracies are around 0.01%. The LR method achieved the lowest accuracy results, with rates around 0.03% lower than CNN on the average. This method achieved an accuracy rate of around

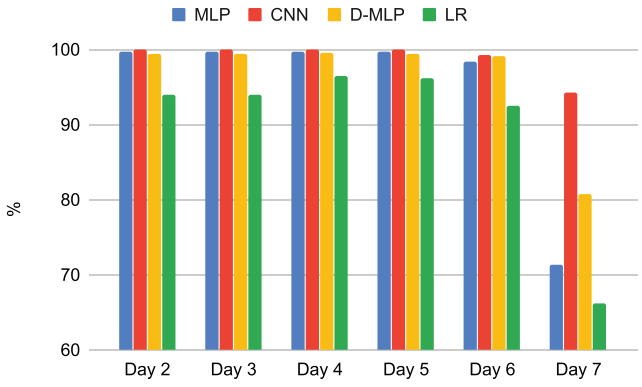


Fig. 10. Methods' precision outcomes for the first test scenario.

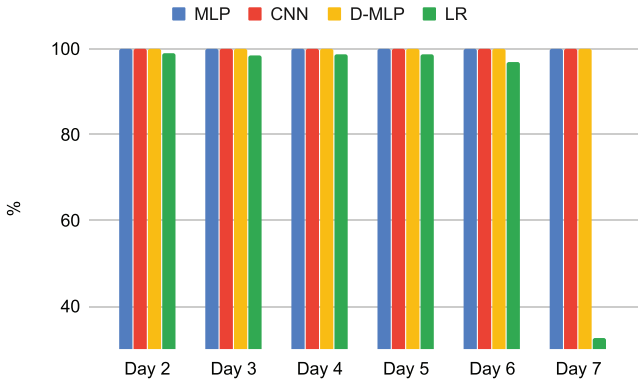


Fig. 11. Methods' recall outcomes for the first test scenario.

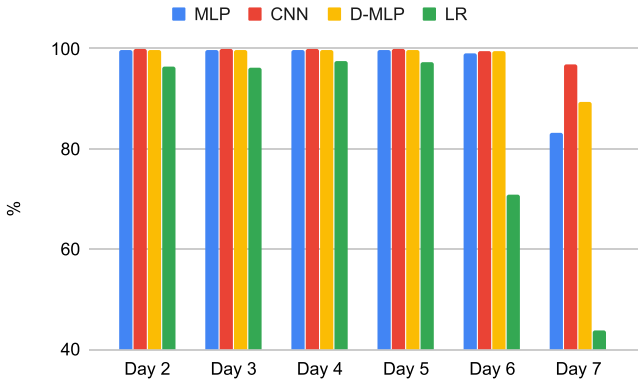


Fig. 12. Methods' f-measure outcomes for the first test scenario.

97% on Day 6, a day with a stealthier attack. However, on Day 7, the LR method achieved an improved accuracy outcome, even though both days present the same attack intensity. This occurrence is due to the attack duration since Day 7 has a shorter malicious time interval than Day 6.

As shown by Fig. 10, the CNN method achieved better outcomes for the precision metric, with a rate of 99.9% on the average. This result was expected, as the analysis of the confusion matrices pointed out that this method is more efficient in classifying benign intervals. Furthermore, MLP achieved precision rates of 99.7% on the average, faring slightly better than its deep-learning approach, which achieved precision rates of 99.4% on the average, possibly due to an overfitting occurrence. As pointed out by Chollet [15], when a small amount of data is available, models with fewer layers tend to be more efficient. Finally, the LR method fared worse, achieving better results on Days 4 and 5, when the number of hosts and attack intensity are nearer to the training set, with a precision rate of 95.1% on the average. However, most methods achieved low precision rates on Day 7, which is due to the day's characteristics (low DDoS intensity for a short period). On this day, CNN achieved a precision rate of 94.2%, which is lower than the results regarding the other analyzed days but significantly higher than the rates achieved by the different methods. Thus, it is possible to infer that the CNN method efficiently extracted the main characteristics of both benign and malicious traffic.

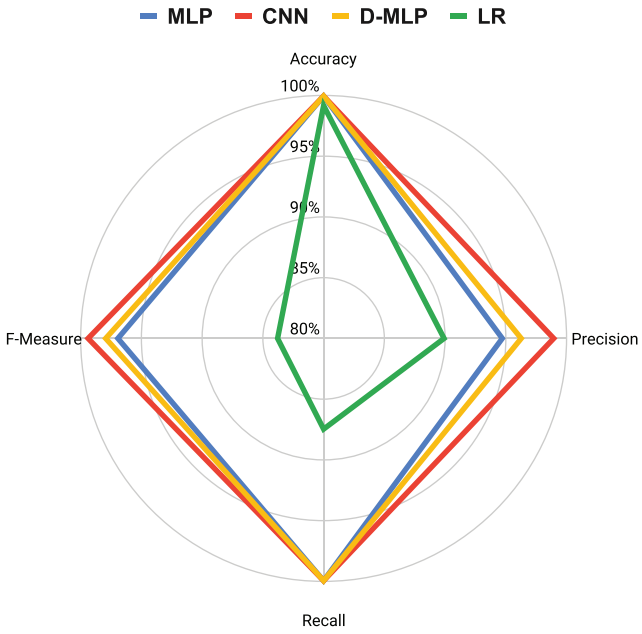


Fig. 13. Radar plot presenting the methods' average outcomes for the first test scenario.

As previously cited, the recall metric shows the efficiency in identifying DDoS intervals. The results presented by Fig. 11 show that CNN, MLP, and D-MLP achieved very similar results, with rates of around 99.9%, even though CNN fared slightly worse (with recall rates 0.04% lesser than MLP and D-MLP methods). Although LR fared worse than the other approaches, it achieved recall values higher than 96.7% on days 2 to 6, which is a good outcome. However, this method produced a low recall rate on Day 7, which points out that it could not efficiently identify the attack with low intensity performed over a short period in this scenario.

Finally, Fig. 12 presents the methods' outcomes relating to the f-measure metric. As it represents the harmonic mean between precision and recall metrics, CNN fared slightly better than the other techniques on Days 2 to 6, with rates of 99.9% on the average, followed by MLP, D-MLP, and LR methods, which achieved average rates of 99.7%, 99.6% and 91.7%, respectively. However, on Day 7, due to the test day's characteristics, the f-measure achieved by the tested approaches differs the most, with rates of 97%, 89.3%, 83.3%, and 43.8% for the methods CNN, D-MLP, MLP, and LR, respectively. Although all the tested approaches were able to detect DDoS attacks on the analyzed days efficiently, only the CNN method consistently performs this task on all evaluated days, which proves its efficiency in comparison to the other tested approaches.

Fig. 13 presents a radar plot that summarizes the results previously addressed in a single image, where the nearer to the outer circle, the closer to 100% the analyzed approach fared on the average for the four measurement techniques. While CNN, MLP, and D-MLP reached similar outcomes for accuracy and recall, CNN achieved better results for precision and f-measure rates.

Thus, in this test scenario, it is possible to conclude that the CNN method achieved the best classification results, operating as an efficient DDoS identifier at the Detection Module of the presented SDN defense system.

5.2. Scenario 2 - CICDDoS 2019 Data set

In this scenario, we applied simulated IP flows collected from a public data set called CICDDoS 2019 [16]. It generates realistic background traffic profiled through B-Profile System to abstract the behavior of human interactions for benign traffic. In this data set, the authors abstract the behavior of 25 users based on different protocols, such as HTTP, FTP, and SSH.

The CICDDoS 2019 data set separates the data into two days. The first one is a training day, containing 12 types of different DDoS attacks, including NTP, DNS, MSSQL, LDAP, NetBIOS, SNMP, UDP, UDP-Lag, SSDP, Syn, WebDDoS and TFTP. The second is a testing day, containing 6 different DDoS attacks, which are NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, and Syn.

This data set provides data with 87 extracted IP Flow features, such as source and destination IP addresses and ports, protocols, several flags, counters, and flow identification features. All the data was submitted to a data formatting process to convert qualitative features into quantitative ones, and to group data into one-second intervals like described in Section 3.

The parameters of the CNN were set with 64 and 32 filters and a kernel size of 32 and 16 for the first and second Conv1D layers, respectively. Both MaxPooling1D layers were set with a pool size of 2, and the Dropout with a rate of 0.5. The Fully-connected layer is composed of 10 neurons, and the output layer is formed of 1 neuron to generate a binary result. All methods were tested through 1000 epochs.

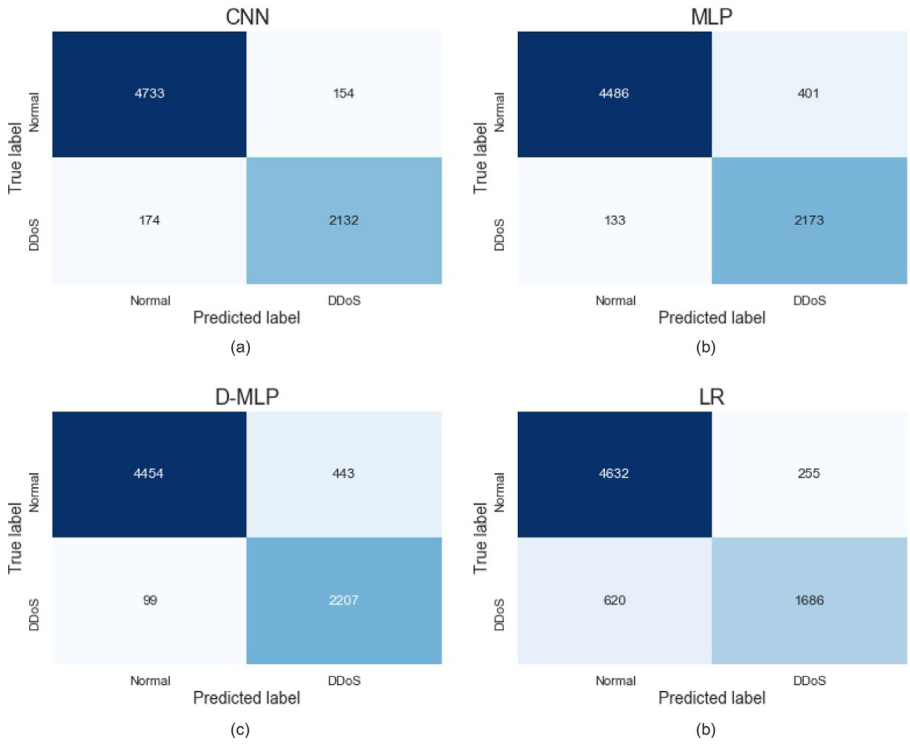


Fig. 14. Confusion Matrices of the tested methods relating the first test scenario.

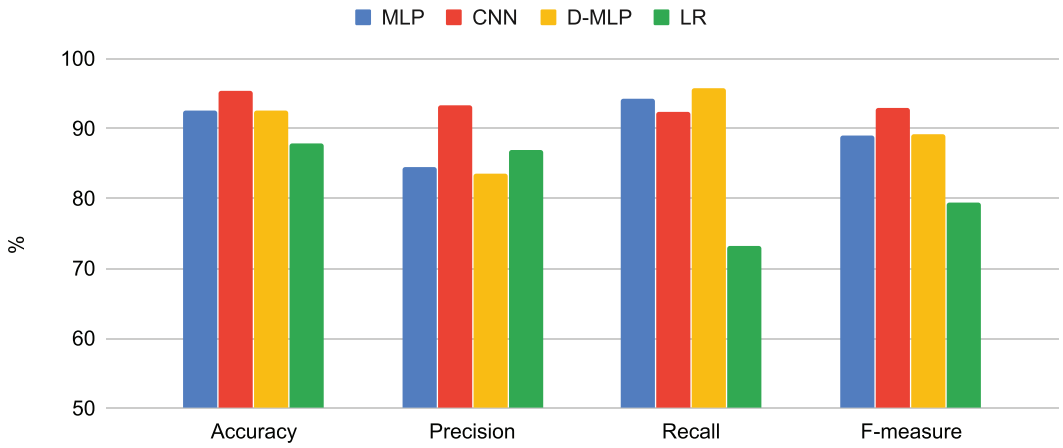


Fig. 15. Methods' outcomes for accuracy, precision, recall and f-measure metrics over the second test scenario.

The efficiency measurement was performed as described in the first test scenario, comparing CNN with the methods MLP, D-MLP, and LR over classical techniques. Fig. 14 shows the outcomes achieved by each one of the tested methods through confusion matrices on this scenario.

The second test scenario presents some interesting differences from the first one, which can be observed at the Confusion Matrices presented by Fig. 14. Besides having less normal intervals, the DDoS attacks present in this data are related to 12 different behaviors, which makes the binary classification process a complex task. As observed by the confusion matrices, the number of false-positive and false-negative intervals is higher for all the methods in comparison to the ones achieved in the first test scenario. With relation to the number of false-positive intervals, the CNN method fared better, followed by LR, MLP, and D-MLP, respectively. When considering the false-negative intervals, the D-MLP fared better, followed by MLP, CNN, and LR methods. While MLP and D-MLP presented a greater difficulty in classifying normal intervals, the LR method faired worse on classifying DDoS intervals. The CNN method, in turn, presented the most balanced outcome.

The results achieved by the tested methods using classical metrics in this scenario are shown in Fig. 15

As observed in Fig. 15, the CNN method obtained better accuracy measures, reaching a rate of 95.4%, followed by D-MLP, MLP, and LR approaches, which achieved rates of 92.6%, 92.5% and 87.8%, respectively.

For the precision metric, the CNN approach also fared better, with a rate of 93.3%, followed by LR, MLP, and D-MLP methods, with 86.8%, 84.4% and 83.4% precision rates, respectively.

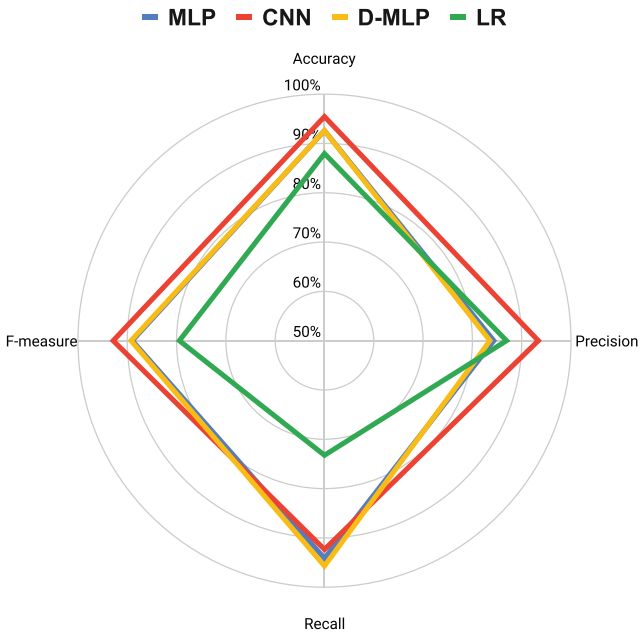


Fig. 16. Radar plot presenting the methods' average outcomes for the second test scenario.

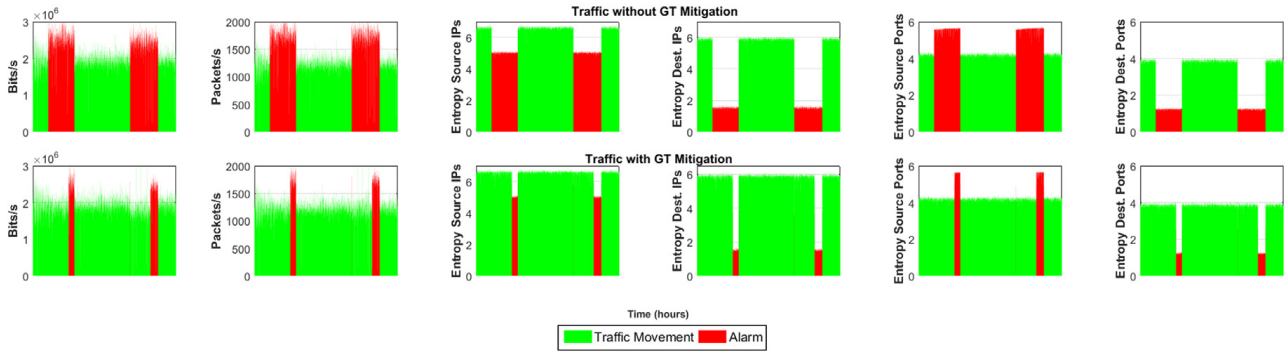


Fig. 17. Hexa-dimensional IP flow view of the analyzed SDN, with two DDoS attacks based on 15 hosts, before and afterwards the procedures of detection and mitigation utilizing CNN and GT.

For the recall metric, the D-MLP achieved the best results, with a rate of 95.7%, followed by MLP, with a percentage of 94.2%, CNN, which reached a 92.4% recall measure, and LR, with a rate of 77.1%.

Finally, for the f-measure metric, the CNN method achieved better outcomes with a rate of 92.8%, followed by D-MLP, MLP, and LR methods, which obtained an f-measure result of 89.2%, 89% and 79.4%, respectively.

Fig. 16 summarizes the previously addressed metrics' results in a single figure. In short, CNN methods achieved better accuracy, precision, and f-measure results than the other tested methods, reaching values around 95%. In turn, D-MLP presented the best recall outcome, followed by the MLP method, which produced similar results using fewer layers.

As the results achieved in the first scenario, the CNN method also fared better on the average than the other approaches on the second test environment, achieving promising test outcomes that make it an efficient technique on DDoS detecting.

5.3. Mitigation

As previously discussed in Section 3, the proposal of a novel mitigation approach is not in the scope of this paper. So, to demonstrate the efficiency of the presented SDN security system, we applied a game-theoretical (GT) approach, proposed in [29], on the mitigation module. This method is used in DDoS mitigation of internal attacks against external targets.

The Mitigation module is in charge of providing the SDN central controller with the optimal drop policy, aiming to mitigate or even interrupt the DDoS attack entirely. By preventing the distributed attack from reaching the Internet, it is possible to mitigate the attack on the destination-end network indirectly.

Every second, IP flow data are collected and submitted to the Detection module, where a classification process occurs based on the anomaly detection method. This classification outcome may be a "normal" label or a "DDoS" one, in which the Mitigation module is triggered. Thus, the GT-approach analyses the provided information to automatically determine the optimal drop rate as a DDoS countermeasure. Fig. 17 shows the traffic of the analyzed SDN before and after the mitigation

process, relating the first test day of the first scenario through the usage of the CNN anomaly detection method. For better understanding, the figure shows the traffic from 12:30 to 19:30, the interval in which the DDoS attacks occurred.

As observed, the GT-approach retrieved the optimal drop policy, which was incorporated by the SDN controller at the next time interval, *i.e.*, at the upcoming second from the CNN detection. It is also possible to observe that the final stage of the attacks was not mitigated. It occurs because the drop policy generated to minimize the DDoS has a lifespan of one hour, and the attacks lasted longer. When this happens, a new alarm will be generated by the detection module, and the GT-approach will be triggered once again, restarting the process.

As the results point out, the GT-approach was able to mitigate the DDoS attacks successfully. The mitigation module brings the SDN back to its regular operation, and GT represents a feasible approach against both internal and external DDoS attacks.

6. Conclusions

The proposed defense system was capable of inspecting the SDN traffic behavior in one-second time intervals. It effectively detects and mitigates the occurrence of DDoS attacks on the controller and, consequently, over the external targeted server.

We presented a Convolutional Neural Network (CNN) approach to acting within the Detection module, which was tested against three other anomaly detection approaches: the Logistic Regression (LR); the Multi-Layered Perceptron (MLP) network; and Dense MLP. The methods were submitted to two test scenarios. The first one uses simulated SDN data, generated using Mininet and Floodlight, over four different days containing DDoS attacks. The second one uses a public data set known as CicDDoS 2019, providing more than 12 types of DDoS attacks. As the LR method fared worse on most tests, the other three tested methods achieved relatively good results. The CNN method achieved low false-positive rates, higher accuracy, precision, and f-measure outcomes. The MLP and D-MLP methods fared better for the recall metric, achieving similar results.

Moreover, we described the Mitigation module and presented an example of an operation using a game-theoretical approach. To mitigate the attack, we applied a Game Theory (GT) based technique that optimizes the packet discard rate in a policy applied inside the central controller of the SDN. The outcomes reveal that the mitigation approach is efficient in restoring the SDN's regular operation.

For future works, we intend to increase the number of hosts on the simulated SDN environment to test the behavior of the proposed system against stealthier DDoS internal attacks. Furthermore, we want to study the impact of recurrent deep learning approaches, such as LSTM and GRU, on classification problems such as DDoS detection in SDN environments.

Declaration of Competing Interest

Authors declare that they do not have any conflicto of interest

CRediT authorship contribution statement

Marcos V.O. de Assis: Conceptualization, Data curation, Formal analysis, Writing - original draft. **Luiz F. Carvalho:** Conceptualization, Data curation, Formal analysis, Writing - original draft. **Joel J.P.C. Rodrigues:** Conceptualization, Data curation, Formal analysis, Writing - original draft. **Jaime Lloret:** Conceptualization, Data curation, Formal analysis, Writing - original draft. **Mario L. Proença Jr:** Conceptualization, Data curation, Formal analysis, Writing - original draft.

Acknowledgments

This study was financed in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grants 310668/2019-0 and 309335/2017-5; by the Ministerio de Economía y Competitividad in the "Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento" within the project under Grant TIN2017-84802-C2-1-P; by FCT/MCTES through national funds and when applicable co-funded EU funds under the Project UIDB/EEA/50008/2020; and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) by the granting of a scholarship through the "Programa de Doutorado Sanduche no Exterior (PDSE) 2019". Finally, this work was supported by Federal University of Paraná (UFPR) under Project Banpesq/2014016797.

References

[1] Tudosa I, Picariello F, Balestrieri E, De Vito L, Lamonaca F. Hardware security in IoT era: the role of measurements and instrumentation. In: 2019 II workshop on metrology for industry 4.0 and IoT (MetroInd4.0 IoT); 2019. p. 285–90. doi:10.1109/METROI4.2019.8792895.

[2] El-Mougy A, Ibnkahla M, Hegazy L. Software-defined wireless network architectures for the internet-of-things. In: 2015 IEEE 40th local computer networks conference workshops (LCN Workshops); 2015. p. 804–11. doi:10.1109/LCNW.2015.7365931.

[3] Chaabouni N, Mosbah M, Zemmari A, Sauvignac C, Faruki P. Network intrusion detection for IoT security based on learning techniques. IEEE Commun Surv Tut 2019;21(3):2671–701. doi:10.1109/COMST.2019.2896380.

[4] Kim H, Kim T, Jang D. An intelligent improvement of internet-wide scan engine for fast discovery of vulnerable IoT devices. Symmetry 2018;10(5). doi:10.3390/sym10050151.

[5] Bizanis N, Kuipers FA. Sdn and virtualization solutions for the internet of things: a survey. IEEE Access 2016;4:5591–606. doi:10.1109/ACCESS.2016.2607786.

[6] Caraguay A, Peral A, López L, Villalba L. SDN: evolution and opportunities in the development IoT applications. Int J Distrib SensNetw 2014;10(5):735142. doi:10.1155/2014/735142.
[7] Rego A, Garcia L, Sendra S, Lloret J. Software Defined Network-based control system for an efficient traffic management for emergency situations in smart cities. Fut Gener Comput Syst 2018;88:243–53. doi:10.1016/j.future.2018.05.054.
[8] CISCO. Annual cybersecurity report. 2018. Accessed 2 February 2019; https://www.cisco.com/c/en/us/products/security/security-reports.html.
[9] Kolias C, Kambourakis G, Stavrou A, Voas J. Ddos in the IoT: Mirai and other botnets. Computer 2017;50(7):80–4. doi:10.1109/MC.2017.201.
[10] Proenca ML, Zarpelao BB, Mendes LS. Anomaly detection for network servers using digital signature of network segment. In: Advanced industrial conference on telecommunications/service assurance with partial and intermittent resources conference/e-learning on telecommunications workshop (AICT/SAPIR/ELETE'05); 2005. p. 290–5. doi:10.1109/AICT.2005.26.
[11] Semerci M, Cemgil AT, Sankur B. An intelligent cyber security system against DDoSattacks in sip networks. Comput Netw 2018;136:137–54. doi:10.1016/j.comnet.2018.02.025.
[12] Mirkovic J, Prier G, Reiher P. Attacking DDoS at the source. In: 10th IEEE International conference on network protocols, 2002. Proceedings.; 2002. p. 312–21. doi:10.1109/ICNP.2002.1181418.
[13] Carvalho LF, Abrão T, de Souza Mendes L, Proença ML. An ecosystem for anomaly detection and mitigation in software-defined networking. Expert Syst Appl 2018;104:121–33. doi:10.1016/j.eswa.2018.03.027.
[14] Mladenov B. Studying the DDoS attack effect over SDN controller southbound channel. In: 2019 X National conference with international participation (ELECTRONICA); 2019. p. 1–4. doi:10.1109/ELECTRONICA.2019.8825601.
[15] Chollet F. Deep learning with python. 1st . Greenwich, CT, USA: Manning Publications Co.; 2017. ISBN 1617294438, 9781617294433
[16] Sharafaldin I, Lashkari A, Hakak S, Ghorbani A. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In: 2019 IEEE 53rd International Carnahan Conference on Security Technology; 2019.
[17] Fernandes Jr G, Rodrigues JJ, Carvalho LF, Al-Muhtadi JF, Proença Jr ML. A comprehensive survey on network anomaly detection. Telecommun Syst 2019;70(3):447–89. doi:10.1007/s11235-018-0475-8.
[18] Wang Z. An elastic and resiliency defense against DDoS attacks on the critical DNS authoritative infrastructure. J Comput Syst Sci 2019;99:1–26. doi:10.1016/j.jcss.2017.05.012.
[19] Johnson Singh K, Thongam K, De T. Entropy-based application layer DDoS attack detection using artificial neural networks. Entropy 2016;18(10). doi:10.3390/e18100350.
[20] Wang Y, An J, Huang W. Using CNN-based representation learning method for malicious traffic identification. In: 2018 IEEE/ACIS 17th international conference on computer and information science (ICIS); 2018. p. 400–4. doi:10.1109/ICIS.2018.8466404.
[21] Liu H, Lang B, Liu M, Yan H. CNN and RNN based payload classification methods for attack detection. Knowl-Based Syst 2019;163:332–41. doi:10.1016/j.knosys.2018.08.036.
[22] Jimenez JM, Romero O, Lloret J, Diaz JR. Energy savings consumption on public wireless networks by SDN management. Mob Netw Appl 2016. doi:10.1007/s11036-016-0784-7.
[23] Wang B, Zheng Y, Lou W, Hou YT. DDoS attack protection in the era of cloud computing and Software-Defined Networking. Comput Netw 2015;81:308–19. doi:10.1016/j.comnet.2015.02.026.
[24] Cui Y, Yan L, Li S, Xing H, Pan W, Zhu J, et al. SD-Anti-DDoS: fast and efficient DDoS defense in software-defined networks. J Netw Comput Appl 2016;68:65–79. doi:10.1016/j.jnca.2016.04.005.
[25] Joldzic O, Djuric Z, Vuletic P. A transparent and scalable anomaly-based DoS detection method. Comput Netw 2016;104:27–42. doi:10.1016/j.comnet.2016.05.004.
[26] Chen K, Junuthula AR, Siddhrau IK, Xu Y, Chao HJ. SDNShield: towards more comprehensive defense against DDoS attacks on SDN control plane. In: 2016 IEEE conference on communications and network security (CNS); 2016. p. 28–36. doi:10.1109/CNS.2016.7860467.
[27] Hameed S, Ahmed Khan H. SDN based collaborative scheme for mitigation of DDoS attacks. FutInternet 2018;10(3). doi:10.3390/fi10030023.
[28] Behal S, Kumar K, Sachdeva M. D-FACE: an anomaly based distributed approach for early detection of DDoS attacks and flash events. J Netw Comput Appl 2018;111:49–63. doi:10.1016/j.jnca.2018.03.024.
[29] Assis MVOD, Hamamoto AH, Abrão T, Proença ML. A game theoretical based system using holt-winters and genetic algorithm with fuzzy logic for DoS/DDoS mitigation on SDN networks. IEEE Access 2017;5:9485–96. doi:10.1109/ACCESS.2017.2702341.
[30] Abdulhammed R, Faezipour M, Abuzneid A, AbuMallouh A. Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic. IEEE Sens Lett 2019;3(1):1–4. doi:10.1109/LSENS.2018.2879990.

Marcos Vinicius Oliveira de Assis is a professor of the Federal University of Paraná, Brazil. He received a M.Sc. in Computer Science at the State University of Londrina Brazil and is a Ph.D. student in Electrical Engineering at the same institution. He worked as a visiting researcher at the Polytechnic University of Valencia Spain. He is part of the research group "Computer Networks and Data Communication."

Luiz Fernando Carvalho received the Ph.D. degree in Electrical Engineering and Telecommunications from State University of Campinas in 2018. He completed his masters degree in Computer Science at State University of Londrina in 2014. He has experience in Computer Science with emphasis in Computer Networks and is part of the research group Computer Networks and Data Communication. His main research interests are management and security of computer networks.

Joel J. P. C. Rodrigues is a professor at the Federal University of Piauí, Brazil; senior researcher at the *Instituto de Telecomunicações*, Portugal; and collaborator of the Post-Graduation Program on Teleinformatics Engineering at the Federal University of Ceará (UFC), Brazil. He is the leader of the Next Generation Networks and Applications (NetGNA) research group (CNPq), an IEEE Distinguished Lecturer, and Fellow of IEEE.

Jaime Lloret received his B.Sc.+M.Sc. in Physics in 1997, his B.Sc.+M.Sc. in electronic Engineering in 2003 and his Ph.D. in telecommunication engineering (Dr. Ing.) in 2006. He is currently Associate Professor in the Polytechnic University of Valencia. Since 2016 he is the Spanish researcher with highest h-index in the TELECOMMUNICATIONS journal list according to Clarivate Analytics Ranking. He is an IEEE Senior, ACM Senior and IARIA Fellow.

Mario Lemes Proença Jr. is an Associate Professor and leader of the research group that studies computer networks in the Computer Science Department at State University of Londrina (UEL), Brazil. He received the Ph.D. degree in Electrical Engineering and Telecommunications from State University of Campinas (UNICAMP) in 2005. He received a M.Sc degree in Computer Science from the Informatics Institute of Federal University of Rio Grande do Sul (UFRGS), in 1998.

APÊNDICE D - *A GRU Deep Learning System against Attacks in Software-defined Networks*

A GRU Deep Learning System against Attacks in Software-defined Networks

Marcos V. O. Assis[a], Luiz F. Carvalho[b], Jaime Lloret[d], Mario L. Proença Jr.[c]

[a]*Department of Engineering and Exacts, Federal University of Paraná, Brazil*
[b]*Federal University of Technology - Paraná, Brazil*
[c]*Computer Science Department, State University of Londrina, Paraná, Brazil*
[d]*Integrated Management Coastal Research Institute, Universitat Politecnica de Valencia, Valencia, Spain*

Abstract

Our communication technologies are continually evolving to an increasingly connected paradigm, where new devices and software solutions require a network connection to provide innovative services and experiences. Management of this new network environment is becoming more and more complex due to new requirements of devices' heterogeneity, regarding the popularization of Internet of Things (IoT), and dynamic traffic, required by next-generation applications and services. To address this problem, Software-defined Networking (SDN) emerges as a management paradigm able to handle these problems through a centralized high-level network approach. However, this centralized characteristic also creates a critical failure spot, since the central controller may be targeted by malicious users aiming to impair the network operation. In this paper, we propose an SDN defense system based on the analysis of single IP flow records, which uses Gated Recurrent Units (GRU) deep learning method on the detection of DDoS and intrusion attacks. This direct flow inspection enables faster mitigation responses, minimizing the attack's impact over the SDN. The proposed model is tested against several different machine learning approaches over two public datasets, the CICDDoS 2019 and the CICIDS 2018. The results point out promising detection rates, as well as an elevated amount of analyzed flows per second, which makes GRU a feasible approach for the proposed system.

Keywords:
Gated Recurrent Units, SDN, Deep Learning, DDoS, Intrusion detection.

Preprint submitted to Journal of Network and Computer Applications September 8, 2020

1. Introduction

The amount of data traveling on the Internet is rapidly increasing due to the growth in popularity and complexity of connected devices and software solutions. The usage of network resources by end users is rising through the popularization of social networks, web banking applications, and e-commerce, for instance. Thus, new cloud-based services are becoming essential to the operation of this new network environment, which brings specific requirements, such as dynamic traffic allocation (Maenhaut et al., 2017). Furthermore, the increasing popularity of Internet of Things (IoT) devices is gradually changing the Internet scenario by increasing the heterogeneously of communication, since each device (thing) has specific network requirements and processing capability (Yoon and Kim, 2017; Bera et al., 2018). In the face of these changes, management and security are becoming impracticable in traditional static network environments (da Costa et al., 2019; Hajiheidari et al., 2019).

A networking paradigm that is gaining space on several recent pieces of research and applications is the Software-defined Networking (SDN) (Zehra and Shah, 2017; Farris et al., 2019). This network paradigm operates by centralizing the network management into a single programmable controller, able to communicate and control network devices such as switches and routers regardless of their manufacturers, as "white-boxes". The SDN separates the control and data planes so that the central controller is responsible for sending, for instance, management and packet forwarding policies to the controlled devices in a scalable and coordinated manner. This characteristic is a valuable feature able to provide next-generation networks with the dynamic architecture they require (Zhang et al., 2019), in which changes can be performed in a fast, programmable, and on-demand way.

However, while bringing essential improvements to the current network architecture, the SDN, as any centralized service, has as a critical failure spot its central controller. Malicious users may target this controller aiming to impair the whole network operation through the usage of different approaches, such as intrusions (Lopez-Martin et al., 2017) and denial of service (DoS) attacks (Daneshgadeh Çakmakçı et al., 2020; Wang et al., 2020; Xu et al., 2020; Zhang et al., 2020). Thus, efficient protection mechanisms are needed in SDNs to guarantee the availability of the network and the quality of the

provided services (Correa Chica et al., 2020).

The occurrence of these attacks can be generically described as an anomaly, a situation when the network behavior differs from its normal state (Proença et al., 2005). The anomaly detection is a widely approached area, with several different methods proposed in the past years (Fernandes et al., 2019). However, it is still an open research field, since no consensus has been reached due to the enormous amount of different network scenarios and architectures available. In SDN environments, security is a central concern, arousing great interest from the scientific community due to the importance of this paradigm to present and future networks (Maziku et al., 2019).

Among all the anomaly detection methods, the IP flow-based ones are proving to be efficient approaches in SDN environments. It is mainly due to the amount of information these systems can provide, which can be used to characterize the regular network operation with high precision. However, most of the research in this area operates through sampling processes, analyzing the data in intervals of five minutes (Cortez et al., 2006; Bereziński et al., 2015; Shuying Chang et al., 2010), one minute (Pena et al., 2014; Sun et al., 2016), or even in smaller time intervals, such as thirty seconds (Carvalho et al., 2018), and five-seconds (De Assis et al., 2018). While the sampling process helps in scaling the defense system, this process may hide stealthier attacks, such as port scans. Thus, the data analysis performed on each flow separately may provide a more precise detection approach, in which the detection method can find anomalies in specific communications and even identifying who is participating in it.

In this paper, we propose a defense system against intrusions and denial-of-service attacks for SDNs based on the analysis of single IP flow records. This individual flow inspection enables faster mitigation responses, ensuring the quality of the services provided by the SDN. The system is divided into two main modules, Detection, and Mitigation.

The Detection Module is responsible for analyzing individual IP flows aiming to identify the occurrence of an anomaly. In this module, we used a recurrent deep learning algorithm called Gated Recurrent Units (GRU) (Cho et al., 2014) as a classifier. Deep learning approaches use multiple layers to learn data representation with various levels of abstraction and is increasingly gaining space among researchers for network applications (Lopez-Martin et al., 2018). GRU is widely applied in problems in which historical information is essential to the performance of classification tasks. In the proposed system, this method inspects individual IP flows through a mul-

tidimensional analysis, operating as a binary flow classifier, *i.e.*, classifying them as normal or abnormal.

The Mitigation Module generates efficient counter-measures against the detected attacks. Since the system proposed in this paper individually analyzes IP flows, it can directly identify the attacking node address. Thus, a directed mitigation approach is proposed, which aims to bring the SDN back to its regular operation through a light and straightforward process.

To evaluate the efficiency of GRU as a detection method, we tested it against seven other shallow and deep learning detection approaches over two different scenarios using public datasets. On the first one, named CICDDoS 2019 (Sharafaldin et al., 2019), we tested the methods over several different kinds of Distributed DoS (DDoS) attacks. The second scenario, called CICIDS 2018 (Sharafaldin et al., 2018), was used to test the efficiency of the detection methods against different intrusion techniques. Furthermore, these two datasets are used to measure the proposed mitigation approach's efficiency regarding each one of the evaluated detection methods. Finally, we tested the number of flows per second the tested anomaly detection approaches can process to prove the proposed system's feasibility.

We can highlight the following as main contributions of this paper:

- A system for SDN defense against intrusion and DDoS attacks;
- A precise anomaly detection scheme based on isolated IP flow analysis, enabling near real-time detection. This approach allows for faster mitigation responses, minimizing the impact suffered by the SDN;
- The efficiency evaluation and comparison of distinct shallow and deep learning anomaly detection techniques applied in public datasets and the efficiency measurement of the proposed mitigation process.

The remainder of this paper is organized as follows: Section 2 presents state of the art through related work; Section 3 describes the organization of the proposed system; Section 4 details the GRU method used for anomaly detection at the Detection Module; Section 5 discusses the performance outcomes achieved by GRU in comparison with seven other methods and the performance evaluation of the mitigation approach; Finally, section 6 presents the conclusions and future works.

2. Related Works

Software-defined Networking (SDN) is an emerging paradigm that significantly improves the management procedures, providing the network administrator with the flexibility of dynamic traffic allocation, as well as an online, softwarized, and centralized configuration. Several authors have been developing solutions through the usage of SDNs, such as Theodorou and Mamatas (2017) that demonstrated the operation of CORAL-SDN, an SDN based solution for the Internet of Things. The authors highlighted several benefits from the usage of this paradigm in Wireless Sensor Networks (WSN), such as the centralized control and the elasticity support regarding WSNs requirements.

Although centralized management is one of the main advantages of SDNs, it also represents a weak spot, since the operation impairment of the controller may lead to critical network issues. Thus, the security of this controller is an essential matter to SDN implementation. In Tatang et al. (2017) the authors proposed the SDN-GUARD, a system for detecting and mitigating rootkits in SDN controllers. Their system performs a dual-view comparison to detect malicious programming attempts, and the authors highlighted the achievement of reasonable detection rates with a relatively small performance overhead. In Nam and Kim (2018), the authors addressed the security enhancement of SDNs through the usage of open-source IDS software called Suricata. The authors also described the usage of OpenFlow to implement SDN security mechanisms. In Gkountis et al. (2017), the authors proposed a lightweight DDoS defense algorithm, based on a simple set of rules, in the protection of SDN environments. The authors highlighted that the proposed approach achieved better results in comparison to other legacy protection schemes regarding an SDN ecosystem of mobile users. In Sidki et al. (2016), the authors proposed fault protection for SDN controllers, enabling the network to maintain its regular operation through a redundancy-based approach using a synchronized slave controller.

One of the main approaches for detecting attacks in SDN environments is to characterize the network's normal behavior. Thus, when an abnormal situation is detected, the system may take countermeasures to mitigate the problem. This approach is called anomaly detection, and several different techniques may perform this task (Pena et al., 2014; Lei, 2017; Qin et al., 2018). On the past years, machine learning (ML) methods have been widely applied as classification systems on the detection of network anomalies. In

Nanda et al. (2016), the authors proposed the usage of machine learning algorithms trained over historical data to detect network attacks. They compared the efficiency of four different ML algorithms, the C4.5, Bayesian Network, Decision Table, and Naive-Bayes, achieving around 91% of prediction accuracy through the use of Bayesian Network. In Kornycky et al. (2017), the authors investigated the use of low-cost WLAN dongles to monitor a network, and passively perform traffic classification, improving service monitoring by focusing on enforcing network security policies. To reach this objective, the authors applied different machine learning methods, such as kNN, WkNN, GMM, GMM-UBM, BCT, PTSVQ, and TRAP-VQ. The results point out that TRAP-VQ, a technique proposed by the authors, achieved the highest f-measure among all tested approaches, proving to be efficient for WLAN traffic characterization regarding prior knowledge requirements and computational complexity. In Fukuda et al. (2017), the authors proposed the usage of the Domain Name System (DNS) backscatter as an additional source of information regarding network activity. The authors applied different machine-learning algorithms to classify originator activity of malicious traffic based on the retrieved dat a, which are classification and regression tree (CART), random forest (RF), and support vector machine (SVM). The proposed algorithm achieved reasonable accuracy and precision outcomes of around 75%, which demonstrates both that DNS backscatter is a good source of network information and that the tested machine learning approaches are efficient for malicious activity detection.

Moreover, several network anomaly detection approaches operate through sampling, analyzing data in time intervals, such as Cortez et al. (2006) and Bereziński et al. (2015), which operates in five-minute ranges. Smaller time intervals implicate on faster anomaly detection, as well as an increase in processing usage for data analysis. In Sun et al. (2016), the authors proposed an anomaly detection method based on Streaming Performance Metrics and Logs operating in one-minute intervals.

Furthermore, a subclass of machine learning algorithms named Deep Learning is increasingly gaining space among researchers in the area (Chowdhury et al., 2019). This subclass describes algorithms that use multiple layers to learn data representation with various levels of abstraction, usually when a large dataset is available for training. In Kao and Jiang (2019), the authors proposed an anomaly detection framework for univariate time series. To achieve this goal, they first divide data into three classes, which are stationary, periodic, and non-stationary time series. Then, different statistical

and Deep Learning methods, such as GRU, STL, SARIMA, LSTM, LSTM with STL, and ADSaS, are applied over these separated data for performing anomaly detection. The results pointed out that the proposed framework obtained better performance outcomes in comparison to related methods regarding precision, recall, and f-measures. Similarly, in Qin et al. (2018), the authors proposed the usage of Long Short Term Memory (LSTM) networks on the detection of anomalies in IP networks. The outcomes achieved shows promising precision and recall rates, demonstrating the efficiency of the method in classification problems.

In this paper, we propose an SDN defense system against DDoS and intrusion attacks. This system, unlike several different approaches described in this section, operates without sampling, acting directly into individual IP flow data, which provides faster detection and, consequently, decreases the impact caused by the attack through a mitigation process. The proposed system performs a multidimensional (multiple flow feature) analysis using GRU deep learning method for detecting attacks on the SDN controller.

3. SDN Defense system

In this section, we describe the operation of the proposed SDN defense system. The management centralization provides many advantages in this paradigm. Still, it is necessary to give the controller security resources to guarantee its operation and, consequently, the quality of the services provided. Thus, in this paper, we propose an SDN security defense system able to detect the occurrence of different intrusion and DDoS attacks on central controllers through an analysis of multi-dimensional IP flows.

Unlike other techniques that analyze traffic data through different time windows (from seconds to minutes), the proposed defense system aims to analyze and identify attacks in individual flows, providing a higher accuracy on the detection and improving the response time for mitigation actions. The main disadvantage of this approach is the amount of data analyzed by the system, which needs to be able to provide fast and accurate classification outcomes. However, it brings the advantages of rapid detection, decreasing the impact caused by the attacks over end users, and users' identification, since IP flow records stores qualitative information, such as source and destination IP addresses and ports used on the communication.

Two different parts compose our SDN defense system, the Detection, and Mitigation modules, which communicate with each other through the central

system logic, as observed in Fig. 1. Each module is composed of two other sub-modules, and the SDN system operates within the SDN controller. All data analysis is performed automatically, and the network administrator only receives notifications about detected attack events.

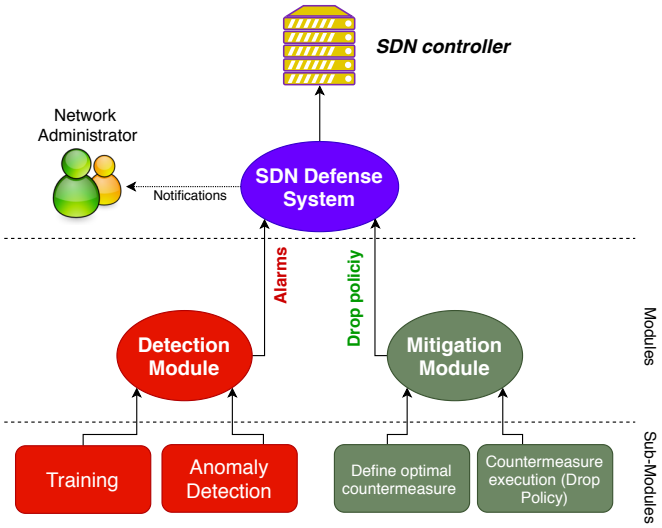


Figure 1: Modular organization of the proposed SDN security system.

The Detection module is responsible for detecting the occurrence of intrusion and DDoS attacks, as well as generating an alarm that invokes the operation of the mitigation module. In this module, we applied a recurrent Deep Learning approach called Gated Recurrent Units (GRU), which will be detailed in the next section. Since GRU is a supervised learning method, the sub-module called "Training" is responsible for calibrating the classification outcomes through the usage of historical labeled data. The second sub-module is accountable for analyzing flow records and generate a binary result, classifying them as normal or abnormal traffic.

It is essential to highlight that the investigation of different IP flow features (or dimensions) enriches traffic analysis by providing relevant information about the communication and who is participating in it (Shuying Chang et al., 2010). Several different approaches use this characteristic to improve the performance of anomaly detection (Shuying Chang et al., 2010; Bereziński et al., 2015; Carvalho et al., 2018). Still, most of them use a

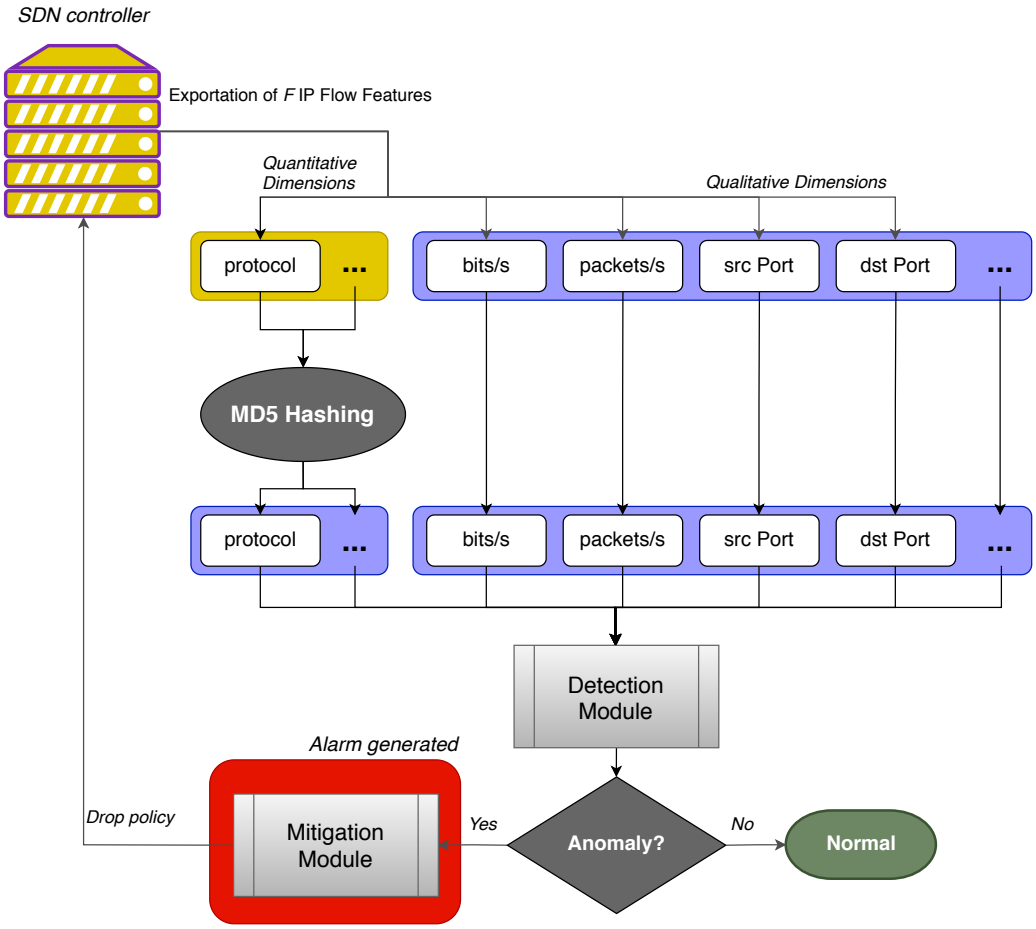


Figure 2: Overall operation of the SDN security system.

set of a few features manually selected, such as bits/s and packets/s rates. However, IP flows can provide a much more comprehensive range of information often unused by traditional methods. Unlike most conventional machine learning methods, Deep Learning approaches, such as GRU, can analyze several flow features and automatically give more weight (importance) to those dimensions that most impact the classification outcomes. This feature is indispensable since some patterns may not be obvious, which significantly improves the anomaly detection process.

The Mitigation module is responsible for defining and taking the optimal countermeasures to minimize the attack's impact over the SDN. As its name

suggests, the first sub-module determines the optimal countermeasure against the detected anomaly. In contrast, the second one sends the optimal drop policy to the SDN controller for implementation.

Different techniques may be used on the "Define optimal countermeasure" sub-module to define the better mitigation action against the detected attack. Since the proposed defense system analyzes single IP flows, it is possible to directly identify the attacker node, individually discarding related flows communication. Thus, there is no need for a probabilistic drop estimation, which could increase the system's operation's overall computational cost.

Therefore, we propose a directed mitigation approach, which represents a straightforward and light drop schema. Using the information provided by the detection module, such as source IP address, protocol, and destination IP address and port, the system generates an individual drop policy against the attacker IP. This policy is implemented by the SDN controller as soon as the first abnormal flow is detected. Hence, this mitigation schema's efficiency is directly related to the efficiency of the attack detection (method).

Fig. 2 summarizes the operation of the proposed SDN defense system. As observed, the SDN controller export single IP flow records composed of F features or dimensions. Most of these features are quantitative dimensions that can be directly analyzed by the detection method. However, if the record contains qualitative dimensions, such as the "protocol" element, this data is submitted to an MD5 hashing process to convert them into quantitative values. An essential part of this treatment is to not include the features source IP address and port, and destination IP address on the anomaly detection step. Their usage could compel the detection method to learn patterns that decrease its generalization capacity, stating that, for instance, DDoS attacks always aim at a specific IP address.

After this step, the IP flow record is submitted to the Detection module, in which it will be presented to a binary classification performed by the GRU method. If no anomaly is detected, then the analysis starts all over with the evaluation of the next flow record. Otherwise, an alarm is triggered, and the Mitigation module is invoked to generate the optimal drop policy against the detected attack. This policy is transmitted to the SDN controller for logic implementation, which forwards the mitigation messages to the network's routers and switches, securing the SDN environment.

It is essential to highlight that the defense system, unlike traditional approaches, operates autonomously, from the flow extraction to the attack's detection and mitigation. This characteristic is a crucial feature to guaran-

tee a fast response regarding the mitigation process, aiming to minimize the impact caused by the attack. When an attack is detected, an alarm is triggered and automatically invokes the mitigation module. This alarm is also received by the network administrator, but only as a warning notification, as described in Fig. 1.

4. Gated Recurrent Units to detect network attacks

Deep learning methods are becoming increasingly popular among researchers through its usage in various applications aiming to detect computer network attacks and anomalies. Deep learning is a class of machine learning that can retrieve patterns in complex data and, therefore, is widely applied to problems of image recognition, pattern classification, and time series prediction (McDermott et al., 2018). Deep learning stands for the concept of successive layers of representations, as its depth is known as the number of layers representing a model. Deep learning approaches typically use three or more layers of representation, whereas "shallow" learning models, such as Multi-Layered Perceptron (MLP), operates through only one or two layers of learning.

One of the most significant benefits of deep learning methods is the absence of manual feature engineering (McDermott et al., 2018). Therefore, there is no need for prior feature selection, as these methods can automatically find patterns among massive datasets during the training step. These patterns are identified, for instance, by weight matrices, which are used to give more "importance" to features that most impact on the classification process. Since IP flow protocols provide a wide range of different dimensions to describe network communications, some elaborate attack patterns, sometimes less evident for human eyes, can be extracted from the dataset, which significantly improves the classification outcomes.

Among the different deep learning approaches, the recurrent ones are promising on the detection of network anomalies and attacks. Unlike the so-called feedforward networks, such as densely connected and convolutional networks, Recurrent Neural Networks (RNN) have memory, *i.e.*, considers past information on prediction/classification process (McDermott et al., 2018). This memory is an essential feature to detecting anomalies, since the state of the network before the occurrence of an attack (previously analyzed IP flows) may be used as well as the current analyzed flow behavior itself to generate alarms. In short, RNNs handles sequences by iterating through

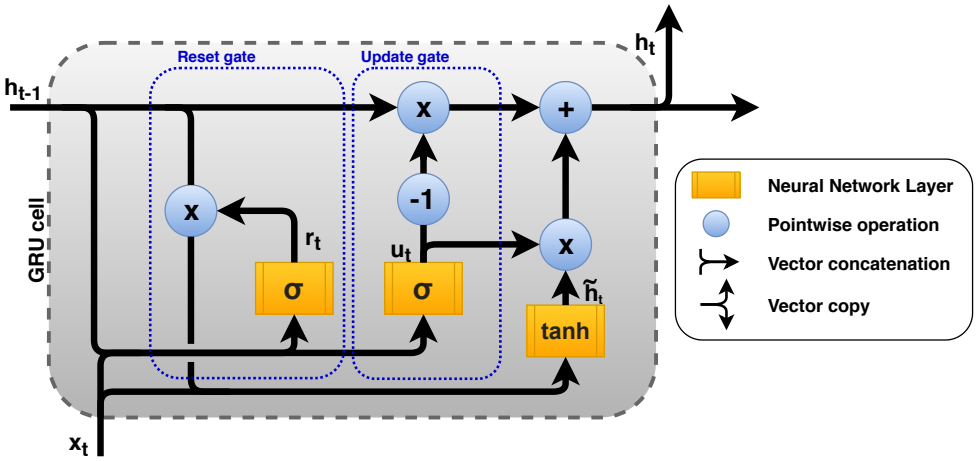


Figure 3: Representation of a GRU cell.

the elements of the series and maintaining a state that contains information about what it has seen so far Cho et al. (2014).

However, RNNs have an issue known as vanishing gradient problem. Although this network should theoretically be able to retain information about inputs seen many timesteps before, in practice, RNN is unable to learn long-term dependencies (He and Droppo, 2016). In short, this occurs because successive operations in long-term data gradually reduce their significance and, thus, the more in-depth the analysis, the less meaningful these data are to influence the network outcome (Bengio et al., 1994).

Different approaches have been proposed to address this problem, while the most commonly used in literature is the Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). This method implements a series of mechanisms called gates, which can regulate learning and forgetfulness rates, guaranteeing that long-term data maintains its influence over recent predictions. Recently, Cho et al. (2014) proposed a modified version of the LSTM called Gated Recurrent Units (GRU), which summarizes the operation of LSTMs by reducing the number of gates while maintaining the relevance of long-term memories (Cho et al., 2014). The efficiency of LSTM and GRU differs regarding the application they are being applied to, as both were not significantly different in classification accuracy (Zhang et al., 2018). However, GRU has fewer tensor operations in comparison to LSTM, which makes its training process faster.

In this paper, we propose the usage of the GRU method for detecting DDoS and intrusion attacks over SDN environments. This method, as well as LSTM, operates through the utilization of gates that are different neural networks that decide which information should be forgotten or retained. GRUs works using two gates, the Update and Reset ones. The first one is responsible for defining what information regarding a new entry will be forgotten and what new information will be added. In contrast, the second one describes how much long-term or past data will be forgotten. Fig. 3 represents the operation of a GRU cell and its gates.

As presented by Fig. 3, h_{t-1} stands for the hidden state of time interval $t-1$, while x_t and h_t represents input data and output hidden state on the current interval t, respectively.

As previously stated, gates are different neural networks used to define which information to forget or retain. As shown in Fig. 3, both gates operate through a sigmoidal activation function, which is applied to simplify this process since it normalizes its output in values between 0 and 1. Thus, any value multiplied by 0 will be forgotten, while values multiplied by 1 will be kept.

To compute the value of h_t, the cell starts by concatenating h_{t-1} and x_t, and then submitting the resulting vector to the Reset and Update gates, obtaining their output r_t and u_t through Eq. (1) and (2), respectively.

$$r_t = \sigma(W_r \ \cdot [h_{t-1}, x_t] + b_r) \tag{1}$$

$$u_t = \sigma(W_u \ \cdot [h_{t-1}, x_t] + b_u), \tag{2}$$

where W_r and W_u stand for the weight matrices of the neural networks, while b_r and b_u are the neural networks' bias vector. Then, a pointwise multiply is performed between r_t and h_{t-1}, and the result is concatenated with x_t and submitted to a third neural network, with a Hyperbolic Tangent ($tanh$) activation function this time. The use of $tanh$ normalize data between -1 and 1, regulating the output of the neural network and preventing data from being over or undersized between iterations. The output $\tilde{h_t}$ of this neural network is computed through Eq. (3).

$$\tilde{h_t} = tanh(W_o \cdot [r_t * h_{t-1}, x_t] + b_o), \tag{3}$$

where W_o and b_o stand for the weight matrix and bias vector of the neural network, respectively. The outcome of the Update gate u_t is used for two

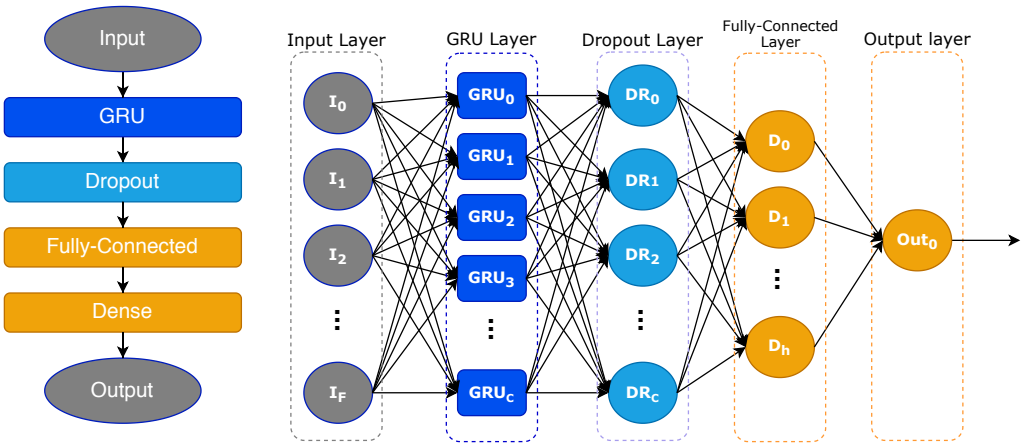


Figure 4: GRU architecture through a high-level representation (a), and through a traditional neural network representation (b).

situations. On the first one, it decides which part of the new information to add by multiplying its outcome with $\tilde{h}_t$. In the second one, it determines which data to throw away by pointwise multiplying h_{t-1} with $1-u_t$. Finally, a pointwise addition is performed between these two outputs, generating the GRU cell's result, the hidden state h_t. Eq. (4) describes this situation.

$$h_t = (1 - u_t) * h_{t-1} + u_t * \tilde{h}_t \tag{4}$$

As this process describes the operation of a single GRU cell, the depth of this network is represented by the number of cells C used to fit the classification regarding the stated problem.

The configuration of the GRU implemented in this paper is described in Fig. 4.

The architecture of the network is composed of a GRU layer ($C = 32$), followed by a Dropout layer (drop rate of 0.5), added to save the system from overfitting, and a Fully-Connected layer ($h = 10$ neurons) that performs a global model classification. The output is given by a single neuron (Dense layer) with a sigmoid activation function for binary classification, *i.e.*, classifying the flow as legitimate or malicious. The parameter estimation of the referred values was defined through extensive empirical testing.

Figs. 5 and 6 present the result of the tests that supported the choice of GRU network parameters, which are the number of GRU cells and the number of neurons on the fully-connected layer. Fig. 5 tests different quan-

tities of GRU cells regarding the accuracy and number of analyzed flows per second. These tests were applied through the use of the CICDDoS 2019 dataset. As observed, the more cells are added to the network, the better are the classification results. However, it is essential to consider the number of flows per second the model can classify since the proposed system aims to analyze individual records. As shown by this figure, GRU achieves a good tradeoff between accuracy and flow throughput with 32 cells.

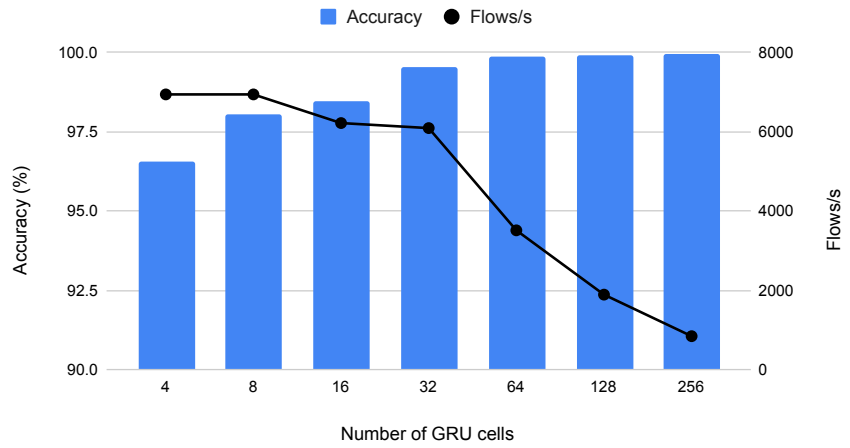


Figure 5: Comparison of different numbers of GRU cells regarding accuracy and amount of analyzed flows per second through the CICDDoS 2019 dataset.

Fig. 6 compares the usage of different amounts of neurons at the fully connected layer, also comparing the efficiency of GRU regarding accuracy and IP flow throughput. These tests were applied through the use of the CICIDS 2018 dataset, as the results achieved using the CICDDoS 2019 dataset are similar to each other. It shows that this layer does not considerably influence the number of analyzed flows/s. Furthermore, the lowest accuracy rate was achieved with 0 neurons, *i.e.*, without adding the fully-connected layer before the output one. Since the accuracy levels seem to achieve similar outcomes with 10 or more neurons, we set $h = 10$ neurons to reduce computational cost.

The drop rate at the Dropout layer was chosen based on Srivastava et al. (2014), in which the authors state that 0.5 appears to be near to optimal for most networks and tasks.

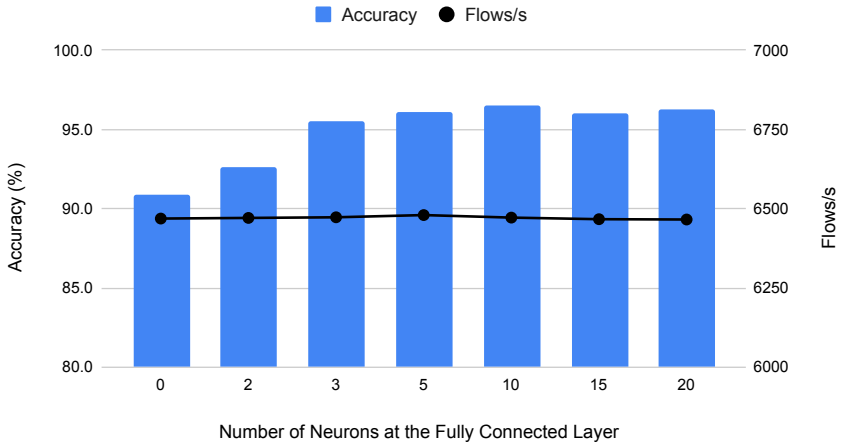


Figure 6: Comparison of different numbers of neurons at the fully-connected layer regarding accuracy and amount of analyzed flows per second through the CICIDS 2018 dataset.

5. Tests and results

This section analyzes the performance results relating to the detection and mitigation modules for the security system. We have applied the GRU method together with the directed mitigation approach. Aiming this objective, we compared the proposed method with the following detection approaches: Deep Neural Network (DNN) (Abdulhammed et al., 2019), Convolutional Neural Network (CNN) (Kwon et al., 2018), Long-Short Term Memory (LSTM) (Qin et al., 2018), Support Vector Machine (SVM) (Lei, 2017), Logistic Regression (LR) (Yadav and Selvakumar, 2015), k-Nearest Neighbors (kNN) (Divyatmika and Sreekesh, 2016) and Gradient Descent (GD) (Wijnhoven and de With, 2010). All methods were implemented using Python, Keras (GRU, LSTM, CNN, and DNN), and Sklearn (SVM, LR, kNN, and GD), on a computer using Windows 10 64bit, Intel Core i7 2.8GHz, and 8GB of RAM. DNN was implemented using 3 hidden layers, composed of 100, 40, and 10 neurons. CNN was implemented using 3 layers, with 64, 32, and 16 filters with kernel size of 16, 8, and 3, respectively. LSTM was configurated with 32 units, similarly to our GRU implementation. The SVM method was configured with a linear kernel. kNN method, in turn, was defined with the number of neighbors equals 3. Finally, LR and GD were configurated with Sklearn default values. All methods were tested through

100 epochs, as many methods converge with fewer iterations.

To compare the efficiency of the tested methods, we applied traditional classification metrics, such as accuracy, precision, recall, and f-measure. The accuracy presents the percentage of correctly classified flow records. Precision is used to measure the ratio of IP flows correctly recognized as abnormal among all the samples classified as unusual. The recall metric estimates the percentage of correctness for anomalous flows. F-measure represents the harmonic mean between true-positive rate (malicious flows classified as anomalous) and precision.

Two different test scenarios were used in this performance analysis, both of them using public datasets. Furthermore, we tested the number of flows per second the anomaly detection approaches can process since it is a vital feature for the system's operation. In the next sections, we describe each test scenario and present the results achieved by the tested methods on them.

5.1. Scenario 1 - CICDDoS 2019 Dataset

We applied simulated IP flows collected from the public dataset CICDDoS 2019 (Sharafaldin et al., 2019). The authors generate realistic background traffic profiled through B-Profile System (Sharafaldin et al., 2017) to abstract the behavior of human communications for legitimate traffic through different protocols, such as HTTP, FTP, and SSH. In this dataset, the authors simulate the behavior of 25 users.

The CICDDoS 2019 dataset separates the data into two days. The first one is a training day, containing 12 types of different DDoS attacks, including DNS, MSSQL, Syn, NetBIOS, LDAP, SNMP, NTP, UDP-Lag, SSDP, Web-DDoS, TFTP and UDP. The second is a testing day, containing 6 different types of DDoS attacks, which are Syn, UDP, NetBIOS, LDAP, UDP-Lag, and MSSQL.

This dataset provides data with $F = 87$ extracted IP Flow features, such as source and destination IP addresses and ports, protocols, several flags, counters, and flow identification features. However, we used only 83 features since the dimensions "source and destination IP address," "source port" and "Flow ID" was not used for training to avoid data bias. The "destination port" feature was maintained since several network applications operate through a default port, which could help the detection of attacks at specific servers. The remaining data was submitted to a formatting process to convert qualitative features into quantitative ones like described in section 3.

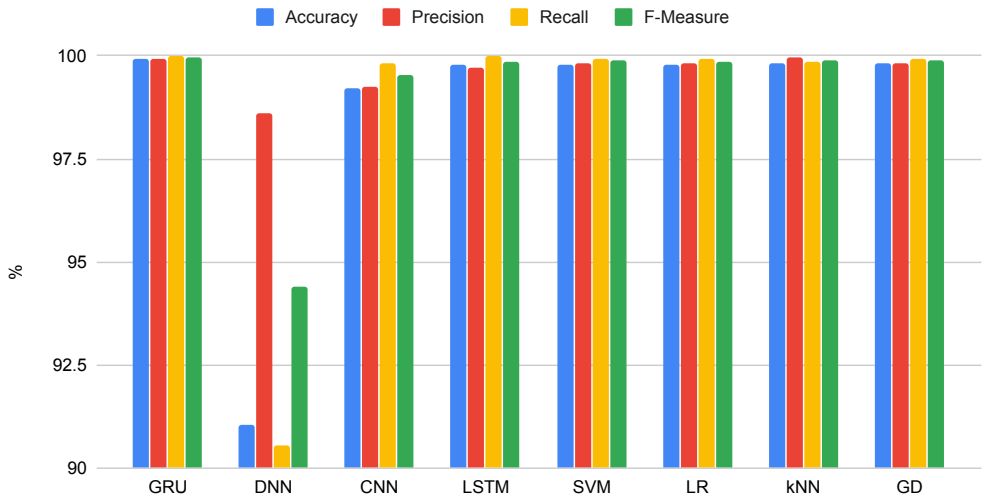


Figure 7: Individual results for each tested method regarding accuracy, precision, recall and f-measure rates - first test scenario.

Fig. 7 presents the results achieved by each method, in which, for each one, we analyze the accuracy, precision, recall, and f-measure separately.

As observed, most of the tested methods achieved excellent outcomes, with metrics' results near 100%. Even DNN, which fared worse than the other tested methods, achieved an accuracy rate of around 91%. CNN method produced better results in comparison to DNN but is also visually worse than the different tested approaches. The results' similarity achieved by the remaining approaches is mainly due to the characteristics of the attacks the dataset used in this scenario. AAlthough there are some differences regarding DDoS types, they are primarily flooding attacks, a behavioral pattern that the tested methods seem to identify efficiently. Fig. 8 shows the same results, but through a stacked-bar graph to ease the visualization of which approach achieved the best outcomes.

Through the analysis of this figure, it is possible to infer that the outcomes achieved by GRU, LSTM, SVM, LR, kNN, and GD are equally efficient in this test scenario since the difference between them is minimal. The GRU method fared slightly better than the other approaches, achieving one of the most balanced outcomes between the four tested metrics.

Aiming to measure the differences between the tested methods further,

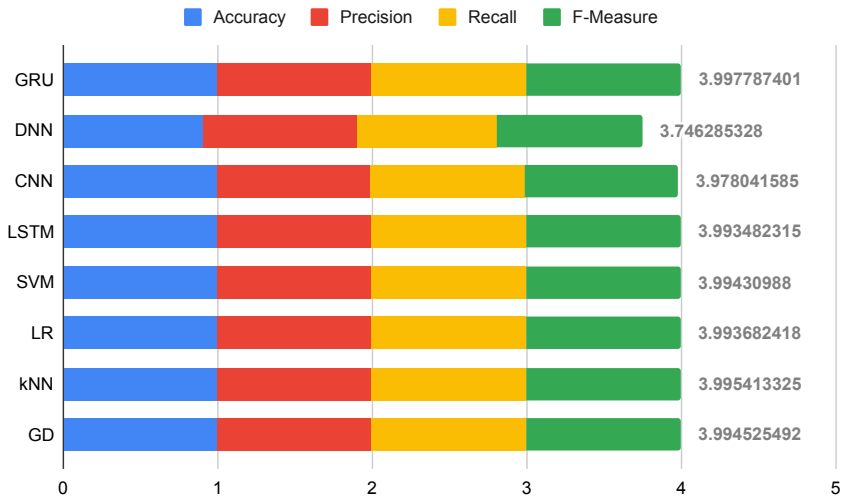


Figure 8: Overall results for each tested method regarding accuracy, precision, recall and f-measure rates - first test scenario.

we performed a separate analysis of the data, measuring their effectiveness on classifying normal (specificity) and attack flows (recall or sensitivity). The results are presented in Fig. 9.

As observed in Fig. 9, most of the methods achieved similar results on classifying attack flows, differing mostly on the classification of the normal ones. In this test scenario, kNN and GRU achieved the best outcomes on legitimate flow classification, with rates of 99.7% and 99.6%, respectively. They are followed by SVM, GD, LR, LSTM, CNN, and DNN, which achieved specificity rates of 99.1%, 99.1%, 99.0%, 98.6%, 96.1%, and 93.6%, respectively. Although these results appear to have a small difference between each other, this difference becomes more significant as the analyzed network size increases, which directly influences the mitigation efficiency.

5.2. Scenario 2 - CICIDS 2018 Dataset

In this test scenario, we applied simulated IP flows collected from the public dataset CICIDS 2018 Sharafaldin et al. (2018). As described in the previous test scenario, the authors also generate realistic background traffic profiled through B-Profile System (Sharafaldin et al., 2017) to describe human communications for legitimate traffic. In this dataset, the authors

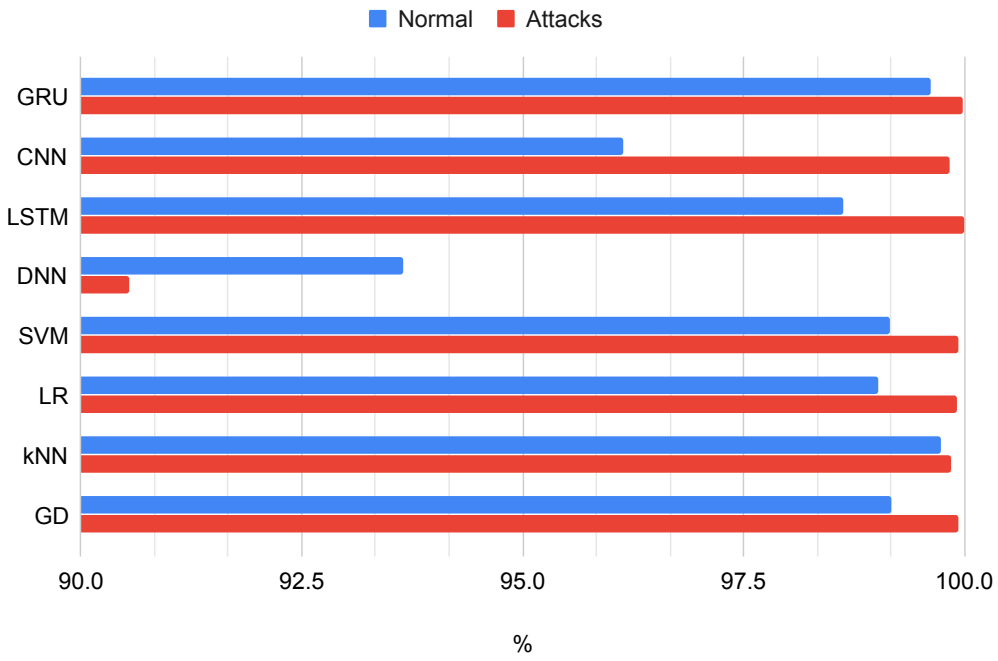


Figure 9: Proportion of correctly identified normal (specificity) and attack (recall) IP flows of each tested method - first test scenario.

simulate the behavior of 500 machines, 420 of legitimate users, 30 of server nodes, and 50 of attacking nodes. The authors used a system called M-Profile to generate different attack scenarios in which the devices operate specific tasks accordingly to the attack's type. They are:

- Infiltration of the network from inside;
- HTTP denial of service;
- Collection of web application attacks;
- Brute force attacks;
- Last updated attacks.

Unlike the previously addressed dataset, the CICIDS 2018 does not divide the data into training and test groups. Thus, we randomly selected 66% of the dataset for training and used the remaining 34% for testing.

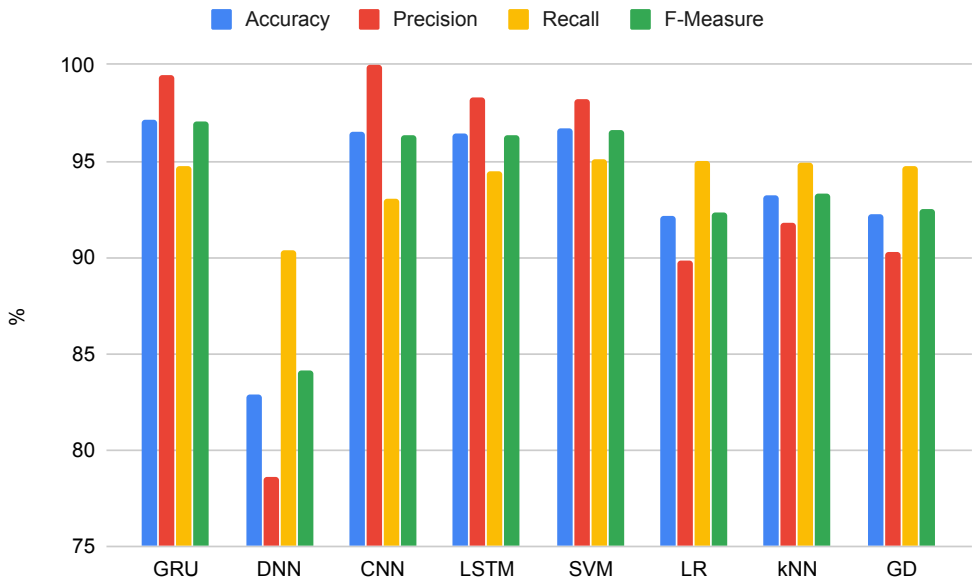


Figure 10: Individual results for each tested method regarding accuracy, precision, recall and f-measure rates - second test scenario.

This dataset provides data in two different extensions: ".pcap", the standard extension for flow export, and ".csv", in which the authors present data ready for training on machine learning methods, *i.e.*, without qualitative dimensions such as source and destination IP addresses and Flow ID. We used the ".csv" files and, thus, in this scenario, the methods operate through the usage of $F = 79$ features provided.

This test scenario has a more significant amount of computers performing legitimate communications. Furthermore, intrusion attacks need to cause minimal impact on the network's behavior to avoid detection, unlike flooding attacks that aim to impair network operations. These characteristics make this test scenario a challenge for the anomaly detection approaches.

Fig. 10 presents the individual results achieved by the tested methods.

As observed, for this scenario, the methods' results for the test metrics are more heterogeneous. However, except for DNN, all approaches achieved accuracy rates higher than 90%, with GRU (97.1%), CNN (96.5%), LSTM (96.4%) and SVM (96.6%) achieving similar results.

For the precision metric, the CNN method fared better than the other

methods, with a rate of 99.9%, followed by GRU, with a 99.4% rate. LSTM and SVM methods achieved the similar precision rates of 98.3% and 98.2%, respectively, followed by kNN (91.7%), GD (90.3%), LR (89.8%) and DNN (78.6%).

For the recall metric the SVM method achieved the best outcomes with a 95.1% rate, closely followed by LR and kNN, with rates of 95% and 94.9%, respectively. GD and GRU achieved similar results of 94.8% and 94.7%, respectively, and LSTM fared slightly worse than the already cited methods in this metric, with a 94.4% rate. CNN and DNN fared worse, reaching recall rates of 93% and 90.4%, respectively.

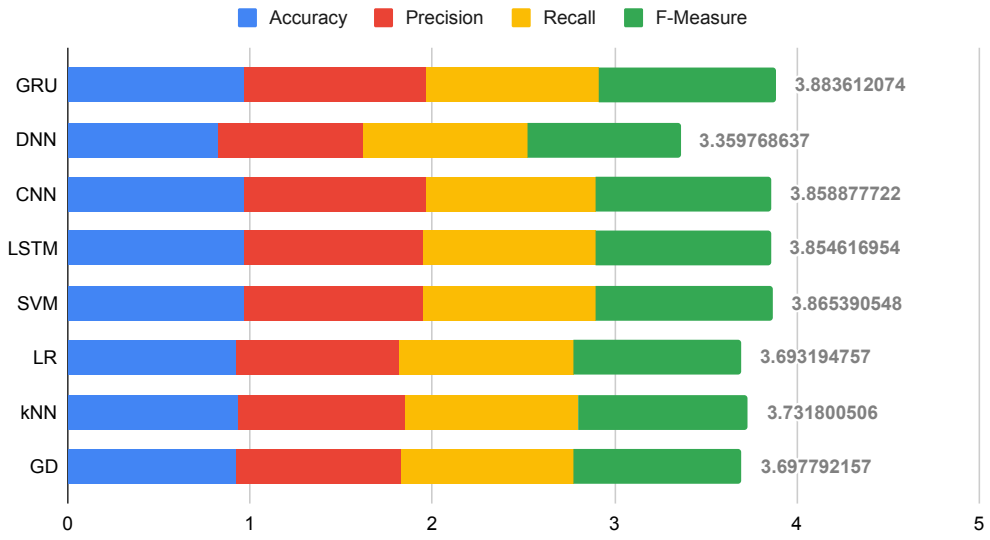


Figure 11: Overall results for each tested method regarding accuracy, precision, recall and f-measure rates - second test scenario.

Finally, for the f-measure, GRU achieved better outcomes, with a 97% rate, being closely followed by SVM (96.6%), CNN (96.4%), and LSTM (96.3%). kNN, GD, and LR methods achieved similar results for this metric, with rates of 93.3%, 92.5%, and 92.3%, respectively, while DNN fared worse with a value of 84%.

Similarly to the first scenario, Fig. 11 presents the results for the four tested metrics through a stacked-bar graph to perform an overall analysis on which method fared better.

As observed, GRU achieved better overall results, being the most balanced approach regarding the accuracy, precision, recall, and f-measure results. It is followed by SVM, CNN, LSTM, which achieved similarly reasonable overall rates. Finally, kNN, GD, LR, and DNN fared worse in comparison to the other tested approaches.

Similarly to the previously analyzed test scenario, we evaluate the effectiveness of the methods' classification regarding normal (specificity) and attack (recall or sensitivity) flows separately. The results are presented in Fig. 12.

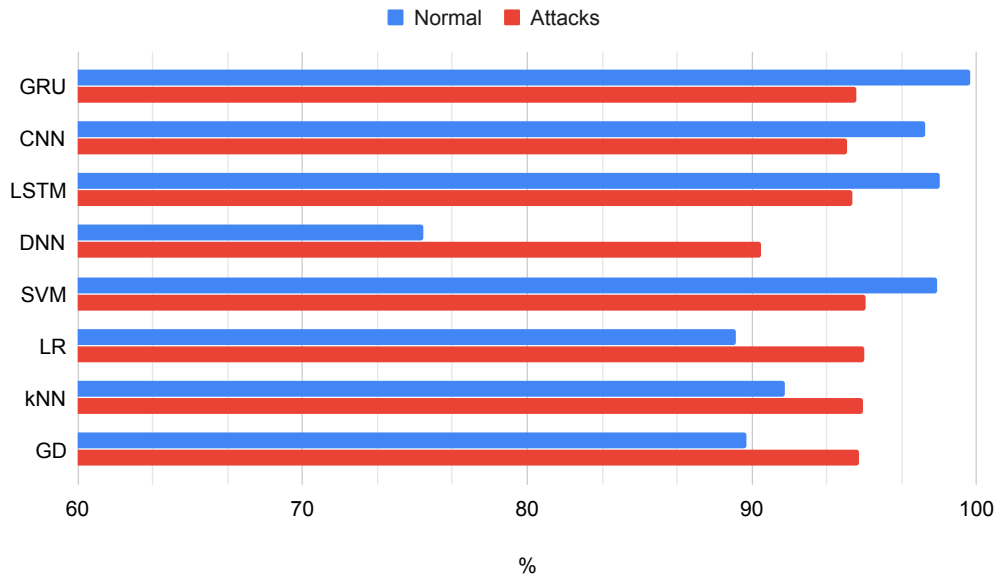


Figure 12: Proportion of correctly identified normal (specificity) and attack (recall) IP flows of each tested method - second test scenario.

As shown in Fig. 12, once again, most of the methods presented similar outcomes regarding the detection of attack flows, while the results differ regarding the correct classification of legitimate flows. In this test scenario, GRU and LSTM achieved the best classification results of normal flows, with rates of 99.7% and 98.3%, respectively. They are followed by SVM, CNN, kNN, GD, LR, and DNN, which achieved specificity rates of 98.2%, 97.7%, 91.4%, 89.8%, 89.2%, and 75.3%, respectively.

We believe GRU fared better than the other tested methods because of

its ability to learn long-term dependencies. This characteristic improves the classification of both normal and abnormal flows, generating more balanced outcomes regarding precision and recall rates, unlike the other tested approaches. However, this ability is also present on the LSTM method, which achieved inferior results in comparison to GRU in both scenarios. Accordingly to Jozefowicz et al. (2015), which performed empirical evaluations to compare both approaches, it is not clear which one present the best results. Although both methods tend to generate similar results, one of them can be more efficient than the other depending on the application scenario (which includes dataset size, number of analyzed features, and so on). GRU is a more straightforward method, which enables a faster training process in comparison to LSTM, which has a more complex structure to learn and describe traffic behaviors. Even though LSTM should achieve better results in more massive datasets due to this characteristic, we can conclude that for these test scenarios, the usage of GRU is more efficient in detecting DDoS and intrusion attacks.

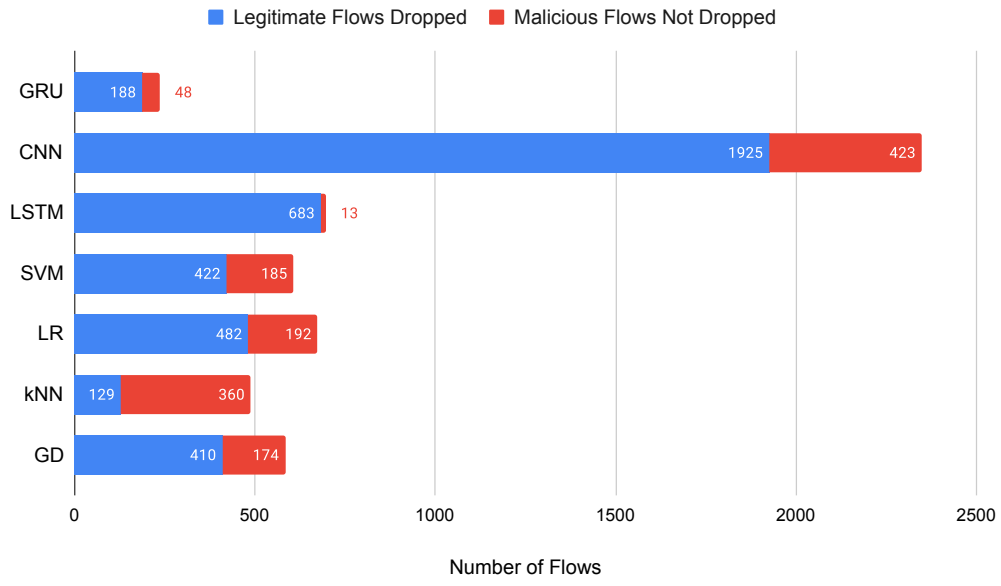


Figure 13: Number of legitimate flows dropped and malicious flows not dropped for each tested method, regarding the dataset CICDDoS 2019

5.3. Mitigation outcomes

As this paper proposes the analysis of individual IP flows for attack detection, it is possible to directly identify which hosts are participating in the communication process and determine the attacker node. Therefore, a simple, straightforward mitigation approach is presented, which drops the identified target packets as soon as the detection occurs.

Thus, the efficiency of the mitigation is directly influenced by the detection approach. To evaluate the mitigation outcomes for each detection method tested, we analyze two metrics: i) the absolute number of normal (legitimate) flows dropped, and ii) the absolute number of attack (malicious) flows not dropped. These values are stacked to summarize the mitigation approach's efficiency in each tested detection method. Fig. 13 and 14 present the mitigation results relating the datasets CICDDoS 2019 and CICIDS 2018, respectively.

As observed in Fig. 13, although the kNN method dropped a lesser amount of legitimate flows in this test scenario, it could not detect as many DDoS intervals than GRU. Thus, GRU achieved the most balanced outcome, guaranteeing both the detection of malicious flows and preserving legitimate ones. Although the LSTM method achieved an outstanding mitigation rate regarding attack flows, it was less efficient than GRU on detecting normal ones.

As shown in Fig. 14, GRU once more achieved the most balanced outcomes in this test scenario. Although the amount of malicious flows not dropped by this method is more significant than the ones conducted by SVM, LR, kNN, and GD, it correctly identified most of the legitimate flows, reducing the impact caused to regular users.

The difference between the values presented by Fig. 13 and 14 occurs due to the size of the test datasets, which are summarized by Table 1.

Table 1: Amount of flows available for testing regarding both datasets CICDDoS 2019 and CICIDS 2018.

	Legitimate flows	**Attack flows**	**Total**
CICDDoS 2019 (testing day)	49,763	248,815	298,578
CICIDS 2018 (33%)	906,094	907,742	1,813,836

As observed, this difference occurs due to the number of flows available for testing, since the CICIDS 2018 testing set was six times bigger than the CICDDoS 2019 one. Furthermore, intrusion attacks tend to be stealthier

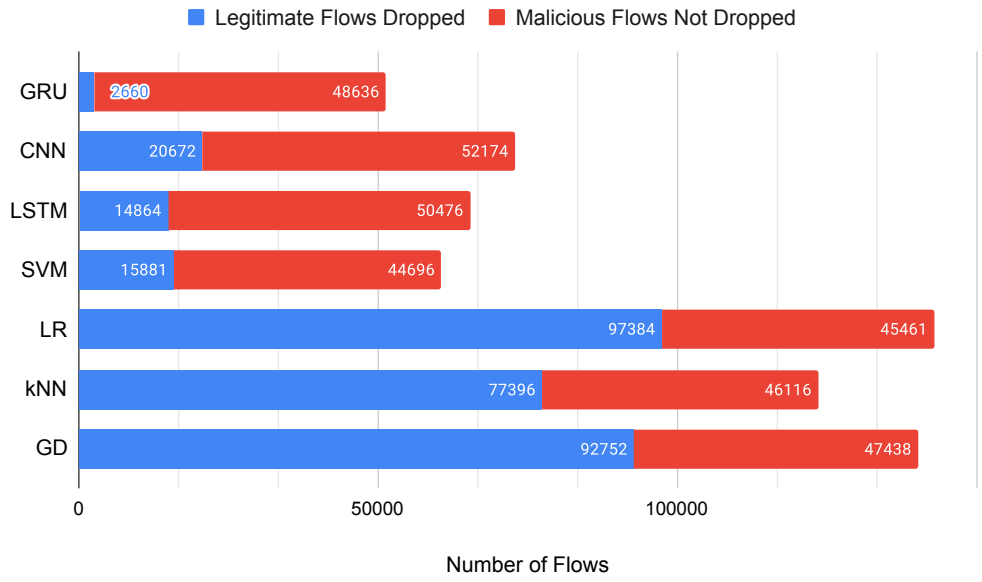


Figure 14: Number of legitimate flows dropped and malicious flows not dropped for each tested method, regarding the dataset CICIDS 2018.

than flooding ones, which elevates the complexity of detection. These values were presented in an absolute form to illustrate how the network's size influences the mitigation outcomes, especially regarding the impact on legitimate users. Therefore, the larger the SDN network (regarding the throughput of IP flows), the more evident is the difference in detection of the evaluated methods.

Since this mitigation approach directly depends on the detection method's efficiency, it represents a low cost, fast mitigation process. Although this approach proved to be efficient, it may replicate misclassifications through time. The generation of a drop time window may improve the mitigation outcomes, which should be implemented in future works.

5.4. Feasibility of implementation

Since the proposed SDN defense system aims to inspect the traffic data through individual flow analysis, the detection method should be both lightweight and efficient. These characteristics are essential, since delays on the detection may impair the operation of the mitigation approach, as well as cause

more damage to the SDN.

To measure the efficiency of the presented approach, we calculate the average number of flows per second the tested anomaly detection methods can analyze and classify. The rates were collected through 10 different executions, and the results are shown in Table 2.

Table 2: Average amount of flows/s each method is able to process.

	Flows/s	**Standard Deviation**
GRU	6,469	0.044
DNN	87,561	0.008
CNN	18,318	0.408
LSTM	5,298	0.272
SVM	1,470	0.661
LR	7,344,471	0.002
kNN	1,597	1.013
GD	7,307,621	0.002

As observed, the LR and GD methods are by far the fastest, in comparison to the other methods. However, their results regarding the classification performance were among the worst tested in this paper. They are followed by DNN, which can classify 87561 flow records per second, CNN, GRU, LSTM, kNN, and SVM.

To verify if the results achieved by the tested methods are feasible on real-world scenarios, we collected real IP flow data from the State University of Londrina (Brazil)[1], a large-scale network composed of about 7000 different active hosts. The data was collected through a whole week through intervals of one second, and, for each day, we measured the average flow/s rate. On regular days, around 500 flows/s pass through the collector, while in heavy traffic days, they achieved peaks of a maximum of 1780 flows/s. Fig. 15 presents the average flow/s rates reached by the tested methods in comparison to the average and worst-case rates observed through the real-world environment data. For ease of data visualization, we take the results of DNN, LR, and GD from the graph since they achieved much higher flow/s rates in comparison with the others.

[1]http://www.uel.br/grupos/orion/datasets.html

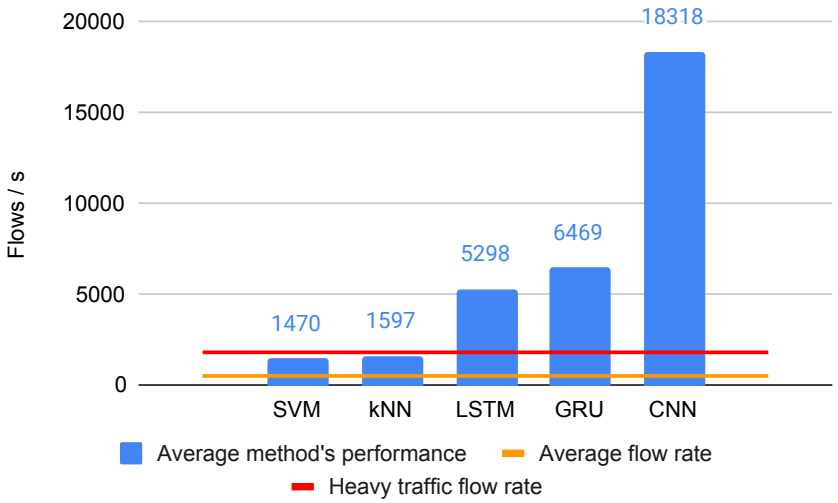


Figure 15: Flow/s rates of GRU, CNN, LSTM, SVM, and kNN methods in comparison to real data rates collected form a real-world, large-scale network.

By comparing the achieved results with the real collected data, we conclude that GRU is a feasible method for real-world anomaly detection. In this case study, it was able to analyze around two times the amount of flows/s required in dense traffic situations, operating within the limited CPU and memory resources of a personal computer. Furthermore, it presented the most balanced results regarding the accuracy, precision, recall, and f-measure for both test scenarios. Thus, it is possible to conclude that GRU is a promising method to operate within the Detection module of the proposed SDN defense system.

6. Conclusions and Future Work

In this paper, we proposed an SDN defense system against intrusion and DDoS attacks. This approach can protect the SDN central controller against situations that may compromise it, consequently impairing the network operation. The proposed system is composed of two main parts, the Detection and Mitigation modules. The Detection module is responsible for detecting the occurrence of attacks, while the Mitigation module takes the required countermeasures to reduce its impact over the network and, consequently, its users. The proposed approach individually analyzes and classifies IP flows

into normal or abnormal, which enables a faster detection process that contributes to minimizing the attack's influence over the SDN controller. Furthermore, this individualized analysis of IP flows can easily identify attackers and their targets through feature extraction.

We also propose the usage of the Gated Recurrent Units (GRU) method on the system's Detection module. GRU is a recurrent deep learning approach that simplifies the operation of the Long-Short Term Memory (LSTM) method while maintaining similar performance outcomes. To test the efficiency of the proposed method, we compare it with seven different approaches. They are Dense Neural Network (DNN), Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM), Support Vector Machine (SVM), Logistic Regression (LR), k-Nearest Neighbors (kNN), and Gradient Descent (GD).

All methods were tested over two different scenarios, both of them using public datasets. The first test scenario uses the CICDDoS 2019 dataset and measures the methods over various types of DDoS attacks, where most of the tested methods achieved similarly good results. The second one uses the CICIDS 2018 dataset, providing different kinds of intrusion attacks for the methods' testing. In this scenario, the results are more heterogeneous since the emulated network is composed of more devices, and these attacks tend to be stealthier than flooding ones. GRU achieved the best overall results in both scenarios, presenting the most balanced outcomes regarding the accuracy, precision, recall, and f-measure. We believe GRU fared better than the other tested methods because of its ability to learn long-term dependencies. Since GRU outperformed all other evaluated methods on classifying legitimate flows, we conclude that this characteristic represents a significant advantage of the GRU compared to them.

Furthermore, we measured the number of flows per second each method can analyze and classify since speed is an essential feature for the proposed system. We compared the outcomes with real data collected from a large-scale network and, added to the results obtained through both test scenarios, concluded that GRU is a promising and feasible approach for anomaly detection in real-world SDN environments.

Finally, we proposed a directed mitigation schema, a straightforward approach based on using the individualized flow information provided by the Detection Module. By identifying the attacker IP, the system can generate an individual drop policy against it. Therefore, although this mitigation schema proved to be efficient in the evaluated test scenarios, it is directly

influenced by the detection method's performance.

For future work, we intend to use the GRU method as a multi-label classifier, able not only to detect the occurrence of anomalies but also to identify them. Moreover, we want to estimate and evaluate a drop time window usage on the Mitigation Module, calculating the optimal time to minimize the computational cost and improve the mitigation outcomes. We believe that these modifications can significantly improve the performance of the proposed system.

Acknowledgements

This study has been partially supported by the National Council for Scientific and Technological Development (CNPq) of Brazil under Grant of Project 310668/2019-0; by the "Ministerio de Economía y Competitividad" in the "Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento" within the project under Grant TIN2017-84802-C2-1-P; and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) by the granting of a scholarship through the "Programa de Doutorado Sanduíche no Exterior (PDSE) 2019". Finally, this work was supported by Federal University of Paraná (UFPR) under Project Banpesq/2014016797.

References

Abdulhammed, R., Faezipour, M., Abuzneid, A., AbuMallouh, A., 2019. Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic. IEEE Sensors Letters 3, 1–4. doi:10.1109/LSENS.2018.2879990.

Bengio, Y., Simard, P., Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks 5, 157–166. doi:10.1109/72.279181.

Bera, S., Misra, S., Roy, S.K., Obaidat, M.S., 2018. Soft-wsn: Software-defined wsn management system for iot applications. IEEE Systems Journal 12, 2074–2081. doi:10.1109/JSYST.2016.2615761.

Bereziński, P., Jasiul, B., Szpyrka, M., 2015. An entropy-based network anomaly detection method. Entropy 17, 2367–2408. doi:10.3390/e17042367.

Carvalho, L.F., ao, T.A., de Souza Mendes, L., Proença, M.L., 2018. An ecosystem for anomaly detection and mitigation in software-defined networking. Expert Systems with Applications 104, 121 – 133. doi:https://doi.org/10.1016/j.eswa.2018.03.027.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar. pp. 1724–1734. doi:10.3115/v1/D14-1179.

Chowdhury, A., Raut, S.A., Narman, H.S., 2019. Da-drls: Drift adaptive deep reinforcement learning based scheduling for iot resource management. Journal of Network and Computer Applications 138, 51 – 65. doi:https://doi.org/10.1016/j.jnca.2019.04.010.

Correa Chica, J.C., Imbachi, J.C., Botero Vega, J.F., 2020. Security in sdn: A comprehensive survey. Journal of Network and Computer Applications 159, 102595. doi:https://doi.org/10.1016/j.jnca.2020.102595.

Cortez, P., Rio, M., Rocha, M., Sousa, P., 2006. Internet traffic forecasting using neural networks, in: The 2006 IEEE International Joint Conference on Neural Network Proceedings, pp. 2635–2642. doi:10.1109/IJCNN.2006.247142.

da Costa, K.A., Papa, J.P., Lisboa, C.O., Munoz, R., de Albuquerque, V.H.C., 2019. Internet of things: A survey on machine learning-based intrusion detection approaches. Computer Networks 151, 147 – 157. doi:https://doi.org/10.1016/j.comnet.2019.01.023.

Daneshgadeh Çakmakçı, S., Kemmerich, T., Ahmed, T., Baykal, N., 2020. Online ddos attack detection using mahalanobis distance and kernel-based learning algorithm. Journal of Network and Computer Applications 168, 102756. doi:https://doi.org/10.1016/j.jnca.2020.102756.

De Assis, M.V.O., Novaes, M.P., Zerbini, C.B., Carvalho, L.F., Abrão, T., Proença, M.L., 2018. Fast defense system against attacks in software defined networks. IEEE Access 6, 69620–69639. doi:10.1109/ACCESS.2018.2878576.

Divyatmika, Sreekesh, M., 2016. A two-tier network based intrusion detection system architecture using machine learning approach, in: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 42–47. doi:10.1109/ICEEOT.2016.7755404.

Farris, I., Taleb, T., Khettab, Y., Song, J., 2019. A survey on emerging sdn and nfv security mechanisms for iot systems. IEEE Communications Surveys Tutorials 21, 812–837. doi:10.1109/COMST.2018.2862350.

Fernandes, Jr., G., Rodrigues, J.J., Carvalho, L.F., Al-Muhtadi, J.F., Proença, Jr., M.L., 2019. A comprehensive survey on network anomaly detection. Telecommun. Syst. 70, 447–489. doi:10.1007/s11235-018-0475-8.

Fukuda, K., Heidemann, J., Qadeer, A., 2017. Detecting malicious activity with dns backscatter over time. IEEE/ACM Transactions on Networking 25, 3203–3218. doi:10.1109/TNET.2017.2724506.

Gkountis, C., Taha, M., Lloret, J., Kambourakis, G., 2017. Lightweight algorithm for protecting sdn controller against ddos attacks, in: 2017 10th IFIP Wireless and Mobile Networking Conference (WMNC), pp. 1–6. doi:10.1109/WMNC.2017.8248858.

Hajiheidari, S., Wakil, K., Badri, M., Navimipour, N.J., 2019. Intrusion detection systems in the internet of things: A comprehensive investigation. Computer Networks 160, 165 – 191. doi:https://doi.org/10.1016/j.comnet.2019.05.014.

He, T., Droppo, J., 2016. Exploiting lstm structure in deep neural networks for speech recognition, in: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5445–5449. doi:10.1109/ICASSP.2016.7472718.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9, 1735–1780. doi:10.1162/neco.1997.9.8.1735.

Jozefowicz, R., Zaremba, W., Sutskever, I., 2015. An empirical exploration of recurrent network architectures, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, JMLR.org. p. 2342–2350.

Kao, J., Jiang, J., 2019. Anomaly detection for univariate time series with statistics and deep learning, in: 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), pp. 404–407. doi:10.1109/ECICE47484.2019.8942727.

Kornycky, J., Abdul-Hameed, O., Kondoz, A., Barber, B.C., 2017. Radio frequency traffic classification over wlan. IEEE/ACM Transactions on Networking 25, 56–68. doi:10.1109/TNET.2016.2562259.

Kwon, D., Natarajan, K., Suh, S.C., Kim, H., Kim, J., 2018. An empirical study on network anomaly detection using convolutional neural networks, in: 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 1595–1598. doi:10.1109/ICDCS.2018.00178.

Lei, Y., 2017. Network anomaly traffic detection algorithm based on svm, in: 2017 International Conference on Robots Intelligent System (ICRIS), pp. 217–220. doi:10.1109/ICRIS.2017.61.

Lopez-Martin, M., Carro, B., Lloret, J., Egea, S., Sanchez-Esguevillas, A., 2018. Deep learning model for multimedia quality of experience prediction based on network flow packets. IEEE Communications Magazine 56, 110–117. doi:10.1109/MCOM.2018.1701156.

Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., Lloret, J., 2017. Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot. Sensors 17. doi:10.3390/s17091967.

Maenhaut, P., Moens, H., Volckaert, B., Ongenae, V., Turck, F.D., 2017. Resource allocation in the cloud: From simulation to experimental validation, in: 2017 IEEE 10th International Conference on Cloud Computing (CLOUD), pp. 701–704. doi:10.1109/CLOUD.2017.96.

Maziku, H., Shetty, S., Nicol, D.M., 2019. Security risk assessment for sdn-enabled smart grids. Computer Communications 133, 1 – 11. doi:https://doi.org/10.1016/j.comcom.2018.10.007.

McDermott, C.D., Majdani, F., Petrovski, A.V., 2018. Botnet detection in the internet of things using deep learning approaches, in: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. doi:10.1109/IJCNN.2018.8489489.

Nam, K., Kim, K., 2018. A study on sdn security enhancement using open source ids/ips suricata, in: 2018 International Conference on Information and Communication Technology Convergence (ICTC), pp. 1124–1126. doi:10.1109/ICTC.2018.8539455.

Nanda, S., Zafari, F., DeCusatis, C., Wedaa, E., Yang, B., 2016. Predicting network attack patterns in sdn using machine learning approach, in: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 167–172. doi:10.1109/NFV-SDN.2016.7919493.

Pena, E.H.M., Barbon, S., Rodrigues, J.J.P.C., Proença, M.L., 2014. Anomaly detection using digital signature of network segment with adaptive arima model and paraconsistent logic, in: 2014 IEEE Symposium on Computers and Communications (ISCC), pp. 1–6. doi:10.1109/ISCC.2014.6912503.

Proença, M.L., Zarpelao, B.B., Mendes, L.S., 2005. Anomaly detection for network servers using digital signature of network segment, in: Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/E-Learning on Telecommunications Workshop (AICT/SAPIR/ELETE'05), pp. 290–295. doi:10.1109/AICT.2005.26.

Qin, G., Chen, Y., Lin, Y., 2018. Anomaly detection using lstm in ip networks, in: 2018 Sixth International Conference on Advanced Cloud and Big Data (CBD), pp. 334–337. doi:10.1109/CBD.2018.00066.

Sharafaldin, I., Gharib, A., Habibi Lashkari, A., Ghorbani, A., 2017. Towards a reliable intrusion detection benchmark dataset. Software Networking 2017, 177–200. doi:10.13052/jsn2445-9739.2017.009.

Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization, in: Proceedings of the 4th International Conference on Information Systems Security and Privacy - Volume 1: ICISSP,, INSTICC. SciTePress. pp. 108–116. doi:10.5220/0006639801080116.

Sharafaldin, I., Lashkari, A.H., Hakak, S., Ghorbani, A.A., 2019. Developing realistic distributed denial of service (ddos) attack dataset and taxon-

omy, in: 2019 International Carnahan Conference on Security Technology (ICCST), pp. 1–8. doi:10.1109/CCST.2019.8888419.

Shuying Chang, Xuesong Qiu, Zhipeng Gao, Feng Qi, Ke Liu, 2010. A flow-based anomaly detection method using entropy and multiple traffic features, in: 2010 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT), pp. 223–227. doi:10.1109/ICBNMT.2010.5705084.

Sidki, L., Ben-Shimol, Y., Sadovski, A., 2016. Fault tolerant mechanisms for sdn controllers, in: 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 173–178. doi:10.1109/NFV-SDN.2016.7919494.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

Sun, D., Fu, M., Zhu, L., Li, G., Lu, Q., 2016. Non-intrusive anomaly detection with streaming performance metrics and logs for devops in public clouds: A case study in aws. IEEE Transactions on Emerging Topics in Computing 4, 278–289. doi:10.1109/TETC.2016.2520883.

Tatang, D., Quinkert, F., Frank, J., Röpke, C., Holz, T., 2017. Sdn-guard: Protecting sdn controllers against sdn rootkits, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 297–302. doi:10.1109/NFV-SDN.2017.8169856.

Theodorou, T., Mamatas, L., 2017. Coral-sdn: A software-defined networking solution for the internet of things, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), pp. 1–2. doi:10.1109/NFV-SDN.2017.8169870.

Wang, P., Yang, L.T., Nie, X., Ren, Z., Li, J., Kuang, L., 2020. Data-driven software defined network attack detection : State-of-the-art and perspectives. Information Sciences 513, 65 – 83. doi:https://doi.org/10.1016/j.ins.2019.08.047.

Wijnhoven, R.G.J., de With, P.H.N., 2010. Fast training of object detection using stochastic gradient descent, in: 2010 20th International Conference on Pattern Recognition, pp. 424–427. doi:10.1109/ICPR.2010.112.

Xu, J., Wang, L., Xu, Z., 2020. An enhanced saturation attack and its mitigation mechanism in software-defined networking. Computer Networks 169, 107092. doi:https://doi.org/10.1016/j.comnet.2019.107092.

Yadav, S., Selvakumar, S., 2015. Detection of application layer ddos attack by modeling user behavior using logistic regression, in: 2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), pp. 1–6. doi:10.1109/ICRITO.2015.7359289.

Yoon, S., Kim, J., 2017. Remote security management server for iot devices, in: 2017 International Conference on Information and Communication Technology Convergence (ICTC), pp. 1162–1164. doi:10.1109/ICTC.2017.8190885.

Zehra, U., Shah, M.A., 2017. A survey on resource allocation in software defined networks (sdn), in: 2017 23rd International Conference on Automation and Computing (ICAC), pp. 1–6. doi:10.23919/IConAC.2017.8082092.

Zhang, S., Wang, Y., Zhou, W., 2019. Towards secure 5g networks: A survey. Computer Networks 162, 106871. doi:https://doi.org/10.1016/j.comnet.2019.106871.

Zhang, X., Xie, L., Yao, W., 2020. Spatio-temporal heterogeneous bandwidth allocation mechanism against ddos attack. Journal of Network and Computer Applications 162, 102658. doi:https://doi.org/10.1016/j.jnca.2020.102658.

Zhang, X., Zhang, Y., Zhang, L., Wang, H., Tang, J., 2018. Ballistocardiogram based person identification and authentication using recurrent neural networks, in: 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1–5. doi:10.1109/CISP-BMEI.2018.8633102.